

FileMaker

Conventions de développement

© 2007 FileMaker, Inc. Tous droits réservés. FileMaker est une marque de FileMaker, Inc., déposée aux Etats-Unis et dans d'autres pays. Le logo du dossier et ScriptMaker sont des marques de FileMaker, Inc. Toutes les autres marques sont la propriété de leurs détenteurs respectifs. Les caractéristiques et la disponibilité des produits sont sujettes à modification sans préavis.

La documentation de FileMaker est protégée par la législation sur les droits d'auteur. En effectuant des téléchargements sur le site Web de FileMaker, vous vous engagez à ne pas créer de copies supplémentaires ni à distribuer cette documentation sans l'accord écrit de FileMaker.

LE PRESENT DOCUMENT EST FOURNI « TEL QUEL », SANS GARANTIE D'AUCUNE SORTE, ET FILEMAKER, INC. DECLINE TOUTE GARANTIE, EXPLICITE OU IMPLICITE, Y COMPRIS, MAIS SANS QUE CETTE LISTE SOIT EXHAUSTIVE, LES GARANTIES IMPLICITES DE QUALITE MARCHANDE OU D'ADEQUATION A UN USAGE PARTICULIER, OU LA GARANTIE DE NON-VIOLATION. EN AUCUN CAS FILEMAKER, INC. OU SES FOURNISSEURS NE SAURAIENT ETRE TENUS RESPONSABLES DE DOMMAGES, QUELLE QU'EN SOIT LA NATURE, Y COMPRIS DES DOMMAGES DIRECTS, INDIRECTS, FORTUITS, CONSECUTIFS A L'UTILISATION DU PRODUIT, AINSI QUE DE LA PERTE DE PROFITS, DE LA CESSATION D'ACTIVITE, OU DE DOMMAGES EN REPARATION OU SPECIAUX, MEME SI FILEMAKER .INC OU SES FOURNISSEURS ONT EU CONNAISSANCE DE LA POSSIBILITE DE TELS DOMMAGES. CERTAINS ETATS N'AUTORISENT PAS L'EXCLUSION OU LA LIMITATION DE RESPONSABILITES. FILEMAKER PEUT APPORTER DES MODIFICATIONS AU PRESENT DOCUMENT A TOUT MOMENT, SANS PREAVIS. LE PRESENT DOCUMENT N'EST PAS NECESSAIREMENT A JOUR, ET FILEMAKER NE S'ENGAGE PAS A METTRE A JOUR CES INFORMATIONS.

Cette recommandation technique est une traduction et adaptation en date du 20 mars 2007 du document en anglais "FileMaker Développement Conventions" (doc v1).

Introduction

La programmation dans FileMaker® Pro est bien différente de la programmation effectuée dans d'autres environnements de développement. Les développeurs FileMaker ont la liberté de créer et de modifier rapidement des applications sans avoir à se soucier de bien des contraintes propres aux autres environnements de développement. Quand vous choisissez une approche interactive du développement, un outil tel que FileMaker vous permet de modifier et d'étendre les fonctionnalités de votre solution sans réellement vous préoccuper des dépendances pouvant intervenir à d'autres endroits de la solution. Toutefois, plus une solution est complexe, plus sa maintenance est difficile.

Par exemple, sans quitter le contexte de développement et parcourir de nombreuses boîtes de dialogue, il est impossible de comprendre les informations essentielles de l'objet programmable principal de FileMaker Pro : la rubrique. Quel est son type de données ? Quels éléments dépendent d'elles ? Comment stocke-t-elle les données ? Où se situe-t-elle ou quel rôle joue-t-elle dans le graphique de liens ? Comment se comportera-t-elle dans un script, dans un modèle, dans un calcul ou dans une liste de valeurs ?

FileMaker Pro est un outil extrêmement souple qui permet d'appréhender le développement de différentes manières bien particulières. Aussi de nombreux standards, différents les uns des autres, ont-ils vu le jour. De nombreux développeurs utilisent des standards. Toutefois, ils ont chacun les leurs, dictés par une méthodologie de développement spécifique ou simplement par leurs préférences particulières. Certains développeurs ont publié leurs standards, mais leur adoption reste limitée. Il n'existe pas de convention de dénomination universelle, adaptée à l'ensemble des méthodologies de développement et des modèles de solutions. En outre, il serait contre-productif d'essayer d'établir des règles aussi rigides.

Le développeur a besoin de recommandations de base pour établir un ensemble de critères à prendre en considération pour toute convention de dénomination. Ces recommandations ne constituent qu'un point de départ, duquel vous pouvez partir pour établir une convention plus étoffée, adaptée à des solutions et méthodes de conception plus exigeantes.

En réponse à ce besoin, FileMaker Inc. a mis en place une commission rassemblant des experts du développement FileMaker pour susciter une prise de conscience et établir des guides pour les conventions de dénomination. Ce groupe spécialisé, appelé FileMaker Development Conventions Advisory Committee (FDCAC), rassemble des experts partenaires de la FileMaker Solutions Alliance (FSA). Ensemble, ils représentent des centaines d'années d'expérience, consacrées à des projets d'entreprises ou de particuliers et au développement de solutions intégrées. Cette commission propose un large éventail d'opinions et de pratiques, représentant un spectre étendu d'approches des conventions plutôt qu'une seule et même vision des choses.



L'objectif de la FDCAC était de créer le livre blanc « Conventions de développement FileMaker », dont le but est de promouvoir la cohérence et le professionnalisme au sein des solutions FileMaker.

Les développeurs qui utilisent FileMaker ont des profils extrêmement divers. De par sa convivialité, FileMaker est la solution idéale pour le développeur novice mais il est également très intéressant pour les développeurs expérimentés, car il permet également de développer rapidement des applications. Les efforts que le développeur doit faire pour appliquer les conventions de dénomination varient en fonction de l'approche choisie.

Le document « Conventions de développement FileMaker » s'adresse à la quasi-totalité des développeurs, mais il concerne plus particulièrement les développeurs de niveau intermédiaire. Il n'est pas prévu pour les développeurs qui débutent sous FileMaker Pro. Il n'est pas non plus adapté aux développeurs de solutions très complexes car les conventions qu'il préconise risquent de ne pas être suffisantes. De la même manière, les développeurs de solutions commerciales ou intégrées risquent de ne pas le trouver adapté à leurs besoins car la complexité et l'architecture de leurs applications nécessitent généralement des conventions plus rigides et plus spécifiques. Toutefois, ces développeurs peuvent s'inspirer de l'esprit de ce document pour redéfinir ensuite des conventions plus précises.

Les recommandations générales du document « Conventions de développement FileMaker » offrent aux développeurs une base de travail. Elles prennent en considération les standards mais n'ont pas pour autant vocation à être adaptées à toutes les architectures ou méthodologies envisageables. Toutefois, elles constituent un bon point de départ, qui peut donner lieu à des modifications par la suite.

L'approche de la FDCAC consistait à identifier et à préciser les problèmes spécifiques que les conventions de dénomination sont censées régler. Après avoir identifié les problèmes, la commission a recherché les points communs au sein de cette multitude de variantes. Souvent, il suffit de définir un ensemble de règles cohérentes pour régler le problème. Toutefois, dans certains cas, c'est le style ou la méthodologie de développement qui dicte la convention à appliquer. Aussi les rédacteurs du document « Conventions de développement FileMaker » ont-ils choisi de formuler des recommandations pour les standards généraux, et d'y ajouter des informations plus détaillées montrant comment certains développeurs appréhendent des aspects plus complexes.

Ce document inclut différentes sections, couvrant presque tous les aspects de l'environnement de développement FileMaker pouvant conduire le développeur à appliquer des conventions de dénomination. La liste ci-dessous répertorie ces sections :

- Dénomination des fichiers
- Dénomination des tables
- Dénomination des rubriques
- Dénomination des occurrences de tables
- Dénomination des modèles
- Documentation et format des calculs
- Dénomination des listes de valeurs
- Documentation et format des fonctions personnalisées
- Dénomination des comptes
- Documentation sur la conformité au document « Conventions de dénomination FileMaker ».

Chaque section est subdivisée en trois sous-sections, dont la première est intitulée « Définition du problème ». Cette section identifie les endroits clés où un développeur peut appliquer une convention. Il peut s'agir de :

- pallier un inconvénient de l'environnement de développement ;
- déjouer les pièges trop fréquents d'une convention ;
- renforcer la cohérence, apporter une meilleure compréhension ou documenter un aspect.



La deuxième section du document « Conventions de développement FileMaker » traite de recommandations visant à sensibiliser les développeurs aux standards. Elle constitue une sorte de modèle de base dont les développeurs peuvent se servir pour appliquer des conventions à leurs projets.

Enfin, ce document présente quelques exemples montrant comment des membres de la DFCAC ont appréhendé les conventions pour un aspect donné. L'objectif est de fournir des exemples de concepts plus avancés et d'aspects qui vont au-delà de la sensibilisation aux standards.

Bien que FileMaker soit persuadé que les informations contenues dans ce document peuvent profiter à tous les développeurs, quel que soit leur niveau, le meilleur moyen de progresser reste de rejoindre la communauté FileMaker. C'est la première étape à franchir pour avancer.

Il est évident que ce document ne s'adresse pas à tous les développeurs. Si vous commencez seulement à utiliser FileMaker, nous vous recommandons de poursuivre votre travail mais nous vous conseillons aussi de parcourir ce document afin de vous familiariser avec les concepts qu'il aborde. Si vous êtes en train de créer une application révolutionnaire et que vous avez élaboré vos propres conventions de dénomination pour un projet faisant intervenir une méthodologie plus avancée, ce document sera une sorte de révision. Toutefois, nous espérons que vous lirez ces informations et que vous essaieriez de les utiliser chaque fois que vous le pourrez.

A tous les développeurs qui se situent dans la moyenne, ce document permet d'accéder aux connaissances d'associés et de partenaires experts de FileMaker. Mieux vaut par conséquent le lire et l'utiliser. Ce document peut vous aider à accélérer le développement de solutions professionnelles conçues pour le long terme.

Préface

Le document « Conventions de développement FileMaker » ne saurait en aucun cas dicter la bonne manière ou la manière universelle de développer des solutions FileMaker. En outre, il serait contre-productif d'essayer d'établir des règles aussi rigides. Après tout, l'une des forces de FileMaker est justement sa souplesse. Ni FileMaker ni personne d'autre ne saurait vous imposer une méthodologie ou une convention de dénomination pour votre solution. Nous ne pensons pas non plus qu'une seule personne ni une seule entreprise dispose du recul suffisant pour établir une convention. L'objectif est ici d'établir une base de travail, qui sera certainement appelée à évoluer. Le succès ne dépend pas d'un individu isolé. C'est l'ensemble de la communauté FileMaker qui doit orienter ses efforts dans ce sens, et, collectivement, apporter de nouveaux éléments pour faire évoluer ce projet. Il est évident qu'il y a au sein de la communauté de développement FileMaker une motivation à utiliser un cadre reposant sur des standards, dans la mesure où nous connaissons tous les avantages des standards en termes de communication et de durabilité.

Cet effort collectif présente des avantages considérables. Les équipes de développement impliquées dans un même projet ont plus de facilité à travailler ensemble. Les développeurs issus d'entreprises différentes ont besoin de moins de temps pour comprendre le code des autres, ou même leur propre code plusieurs années plus tard. Il est plus facile de créer des modèles et des solutions ouvertes à partager, et ceux-ci sont en outre plus cohérents et plus faciles à comprendre. La documentation est plus claire et plus cohérente. En outre, certaines consignes très simples peuvent éviter des problèmes d'évolution et de compatibilité courants dans les solutions que vous créez.

Le document « Conventions de développement FileMaker » 1.0 a été écrit avec la gamme de produits FileMaker 7 comme support de départ. Certains aspects de FileMaker 8 y ont été abordés lorsque cela ne posait pas de difficulté, mais l'objectif était de ne pas traiter de manière trop détaillée des nouvelles fonctions disponibles dans la gamme de produits FileMaker 8. Cette décision s'inscrivait dans la droite lignée de l'un de nos objectifs principaux, qui était de formuler des recommandations reposant sur une expérience réelle. Or, cette expérience n'existait pas pour FileMaker 8 avant la rédaction de ce document. Les futures versions du document « Conventions de développement FileMaker » traiteront de ces aspects si nécessaire.

Ce document profitera aux développeurs débutants tout comme aux développeurs confirmés. Au fur et à mesure que le produit FileMaker évoluera, la multitude de manières que nous avons d'appréhender les problèmes évoluera elle aussi. La communauté de développeurs doit poursuivre ses efforts. Ainsi, comme la gamme de produits FileMaker elle-même, elle s'améliorera à chaque nouvelle version.



Remerciements

Nous tenons à remercier chacune des entreprises suivantes pour le temps qu'elle nous a consacré et pour les connaissances approfondies dont elle nous a fait profiter.

Beezwax	http://www.beezwax.net Vince Menanno Rick Aguire
Core Solutions	http://www.coresolutions.ca Steve Hearn
Dataworks	http://www.dataworks.ca James Hea
DataWaves International	http://www.data-waves.com Peter Makin Colleen Hammersley
FileMaker	http://www.filemaker.com
Geist Interactive	http://www.geistinteractive.com Todd Geist
InResonance	http://www.inresonance.com Corn Walker
iSolutions	http://www.isolutions.com Cris Ippolite
Management Counselling Services	http://www.fmp-power.com Steven Blackwell
The Moyer Group	http://www.moyergroup.com Chris Moyer
New Millenium Communications	http://www.nmci.com Danny Mack
Soliant Consulting	http://www.soliantconsulting.com Steve Lane Scott Love Rodger Jacques
The Support Group	http://www.supportgroup.com Chad Novotny



Table des matières

1	▷ FICHIERS	9
1.1	Objectifs :	9
1.2	Définition du problème :	9
1.3	Recommandations intégrant les standards (fichiers) :	10
1.4	Considérations annexes :	12
2	▷ DÉNOMINATION DES TABLES	13
2.1	Objectifs :	13
2.2	Définition du problème :	13
2.3	Recommandations intégrant les standards (tables) :	14
2.4	Considérations annexes :	15
3	▷ DÉNOMINATION DES RUBRIQUES	16
3.1	Objectifs :	16
3.2	Définition du problème :	17
3.3	Recommandations intégrant les standards (rubriques) :	17
3.3.1	Rubriques générales.....	18
3.3.2	Rubriques clés/sources	18
3.3.3	Rubriques utilitaires	20
3.3.4	Référence de notation de rubriques.....	22
3.4	Considérations annexes :	22
4	▷ OCCURRENCES DE TABLES	23
4.1	Objectifs :	23
4.2	Définition du problème :	24
4.3	Recommandations intégrant les standards (occurrences de tables) :	25
4.3.1	Méthode du groupement fonctionnel en araignée ou FSG (<i>Functional Spider Grouping</i>)	25
4.3.1.1	Avantages du groupement fonctionnel en araignée*	27
4.3.1.2	Inconvénients du groupement fonctionnel en araignée.....	27
4.3.1.3	Groupement fonctionnel en araignée : recommandations intégrant les standards.....	28
4.3.2	Groupement fonctionnel d'occurrences de tables ou FTOG (<i>Functional Table Occurrence Groups</i>)	28
4.3.2.1	Avantages du groupement fonctionnel d'occurrences de tables (FTOG).....	29
4.3.2.2	Inconvénients du groupement fonctionnel d'occurrences de tables (FTOG).....	29
4.3.2.3	Groupement fonctionnel d'occurrences de tables : recommandations intégrant les standards.....	29
4.3.3	Groupement hiérarchique d'occurrences de tables HTOG (<i>Hierarchical Table Occurrence Grouping</i>) ou modèle de l'ancre et de la bouée (<i>Anchor Buoy</i>)	32
4.3.3.1	Avantages du groupement hiérarchique d'occurrences de tables (HTOG)	33
4.3.3.2	Inconvénients du groupement hiérarchique d'occurrences de tables (HTOG)	33
4.3.3.3	Groupement hiérarchique d'occurrences de tables (HTOG) : recommandations intégrant les standards	33
4.4	Considérations annexes :	35



5	▷ MODÈLES	37
5.1	Objectifs :	37
5.2	Définition du problème :	38
5.3	Recommandations intégrant les standards (modèles) :	39
6	▷ FONCTIONS PERSONNALISÉES	40
6.1	Objectifs :	40
6.2	Définition du problème :	40
6.3	Recommandations intégrant les standards (fonctions personnalisées) :	41
6.3.1	Fonctions personnalisées publiques	41
6.3.2	Fonctions personnalisées privées	42
6.3.3	Paramètres de fonctions personnalisées	42
6.3.4	Exemples de dénomination de fonctions personnalisées	43
6.3.5	Documentation des fonctions personnalisées	43
6.3.6	Mise en forme des fonctions personnalisées	44
6.4	Considérations annexes	44
7	▷ SCRIPTS	45
7.1	Objectifs :	45
7.2	Définition du problème :	45
7.3	Recommandations intégrant les standards (scripts) :	45
7.3.1	Noms de scripts	45
7.3.2	Variables	46
7.3.3	Documentation des scripts	46
8	▷ CALCULS	48
8.1	Objectifs	48
8.2	Définition du problème	48
8.3	Recommandations intégrant les standards (calculs) :	48
8.3.1	Variables	48
8.3.2	En-tête de commentaires dans les calculs	49
8.3.3	Commentaires entre les lignes de calculs	49
8.3.4	Mise en forme des calculs	50
8.3.4.1	Exemple de mise en forme n° 1	50
8.3.4.2	Exemple de mise en forme n° 2	51
8.4	Considérations annexes	53
9	▷ LISTES DE VALEURS	54
9.1	Objectifs :	54
9.2	Définition du problème :	54
9.3	Recommandations intégrant les standards (listes de valeurs) :	55



10	▷ COMPTES ET SÉCURITÉ	57
10.1	Objectifs :	58
10.2	Définition du problème.....	58
10.3	Recommandations intégrant les standards (sécurité) :	59
10.3.1	Jeux de privilèges.....	59
10.3.2	Noms de comptes authentifiés en interne	59
10.3.3	Noms de comptes authentifiés en externe (noms de groupes)	60
10.4	Considérations annexes :	61
11	▷ CONFORMITÉ.....	63
11.1	Objectifs de la convention.....	63
11.2	Définition du problème.....	63
11.3	Recommandations pour une convention intégrant les standards.....	64
ANNEXE A	▷ MOTS RÉSERVÉS SQL.....	65
ANNEXE B	▷ TABLEAU SUR L'UTILISATION DES CARACTÈRES.....	74
ANNEXE C	▷ TERMINOLOGIE ET DÉFINITIONS.....	76
ANNEXE D	▷ LÉGENDE DE LA SYNTAXE.....	77



1 ▷ Fichiers

Une solution peut être constituée d'un ou de plusieurs fichiers suivant sa conception et son architecture. Si elle est constituée d'un seul fichier, ce fichier contiendra l'ensemble des tables. Si elle est constituée de plusieurs fichiers, les données et l'interface seront contenues dans des fichiers différents. Une solution peut aussi contenir plusieurs fichiers utilitaires, employés pour la création de rapports ou pour les modules de solutions. Dans tous les cas, une solution est constituée soit d'un seul fichier, soit de plusieurs. Pour chacun de ces fichiers, vous devez prendre en considération un certain nombre de facteurs pendant le processus de dénomination.

Le jeu de caractères utilisé ne doit pas nuire à la connectivité entre la solution et des technologies telles que ODBC, JDBC et XML. Choisissez une casse et tenez-vous y. Outre le choix de la casse, demandez-vous comment les mots qui composent un nom seront séparés. Vous devez également déterminer comment le nom du fichier lui-même sera séparé syntaxiquement des métadonnées, ajoutées sous forme de préfixe ou de suffixe. Dans ce document, nous parlerons de « séparation syntaxique » pour désigner ce point. Vous devez aussi indiquer si les noms doivent être au singulier ou au pluriel pour créer une présentation professionnelle. Enfin, vous devez déterminer comment regrouper la série de fichiers afin de disposer d'une représentation visuelle de la manière dont ils sont liés les uns aux autres. Depuis la version 7 de FileMaker, les développeurs peuvent créer des solutions tout-en-un, dans lesquelles chaque table est contenue dans le même fichier. Il faut savoir que ce n'est pas toujours le meilleur choix, pour des raisons qui dépassent le cadre de ce document. Avec les solutions constituées de plusieurs fichiers, il est important, d'un point de vue administratif et pour l'évolutivité du projet, de suivre une convention qui fasse apparaître clairement cette multiplicité de fichiers. Enfin, vous devez intégrer un certain nombre de considérations multi-plates-formes et de recommandations générales liées aux versions.

1.1 Objectifs :

- ✓ Jeu de caractères recommandé
- ✓ Convention recommandée pour la casse
- ✓ Séparation syntaxique recommandée
- ✓ Longueur recommandée pour les noms de fichiers
- ✓ Méthode de groupement recommandée pour les solutions composées de plusieurs fichiers
- ✓ Recommandations liées à l'usage du singulier et du pluriel
- ✓ Conseils concernant les versions
- ✓ Consignes applicables aux extensions
- ✓ Problèmes liés aux technologies de connectivité (XML, ODBC, JDBC) et aux mots réservés SQL

1.2 Définition du problème :

Jeu de caractères recommandé : les caractères autorisés par FileMaker sont soumis à un certain nombre de restrictions de base. En outre, certains caractères peuvent poser des problèmes de connectivité externe ou d'échange de données dans certains systèmes de gestion de base de données relationnels (SGBDR). En tant que développeur, vous devez avoir conscience des contraintes de FileMaker et de tous les systèmes externes avec lesquels vos solutions doivent interagir.

Convention recommandée pour la casse : cette recommandation n'a pas pour but de résoudre un problème particulier mais d'offrir une méthode de dénomination uniforme pour les fichiers. L'objectif est de sélectionner une méthode et de s'y tenir.

Séparation syntaxique recommandée : dans le domaine de la dénomination de fichiers, la séparation syntaxique désigne la manière de différencier les préfixes et suffixes éventuels du nom du fichier. Les développeurs utilisent souvent des préfixes ou des suffixes pour indiquer qu'un fichier fait partie d'un groupe ou pour signaler des caractéristiques d'identification. L'objectif est d'offrir une méthode cohérente et reconnue universellement pour distinguer ces métadonnées (préfixes ou suffixes) du nom du fichier lui-même.



Instructions concernant les versions : la maintenance des références de fichiers est basée sur les noms de fichiers. Afin d'éviter des problèmes de références de fichiers au fur et à mesure que les versions changent, mieux vaut éviter d'utiliser les noms de fichiers pour représenter la version d'une solution.

Consignes applicables aux extensions : sous Windows, le fichier porte toujours une extension « fpX ». En environnement Macintosh, par défaut, les fichiers portent eux aussi une extension mais ce n'est pas obligatoire. Cela peut poser un problème quand il s'agit d'héberger un fichier sans extension sur un serveur Windows qui, au démarrage du service, recherche l'extension .fpX pour monter le fichier. La convention doit suivre une approche cohérente et prévoir une extension qui fonctionne avec toutes les plates-formes prises en charge.

Recommandations liées à l'usage du singulier et du pluriel : la distinction entre le singulier et le pluriel n'est faite ici que pour signaler qu'en tant que développeur, vous devez faire un choix et vous y tenir.

Problèmes de longueur de nom de fichier : c'est le système d'exploitation qui fixe la longueur maximale des noms de fichiers. Toutefois, il n'est pas pratique d'avoir un nom de fichier de 1 000 caractères. Aussi est-il préférable de choisir une longueur raisonnable pour les noms de fichiers, et de ne pas oublier que certaines applications requises pour le projet peuvent imposer diverses limites. Par exemple, il peut être difficile de joindre à un message électronique un fichier dont le nom est très long. Si vous avez besoin d'interagir avec d'autres applications, sachez que la longueur des noms de fichiers peut poser problème.

Méthode de groupement recommandée pour les solutions composées de plusieurs fichiers : dans une solution composée de plusieurs fichiers, il y a un ou plusieurs fichiers principaux et des fichiers secondaires. Il y a toujours au moins un fichier principal. Les fichiers principaux sont visibles de l'utilisateur final. Dans une solution hébergée, il s'agit des fichiers que l'utilisateur doit sélectionner à partir de la boîte de dialogue hôte. Dans une solution installée en local, il s'agit des fichiers que l'utilisateur doit sélectionner directement à partir du système d'exploitation ou par l'intermédiaire d'un raccourci ou d'un alias. Les fichiers de support sont les fichiers secondaires, qui ne sont pas visibles de l'utilisateur quand il accède directement à la solution. Dans une solution hébergée, ce sont les fichiers utilisés pour prendre en charge la fonction du fichier principal. En outre, certaines architectures séparent l'interface utilisateur de la couche de données, auquel cas il y a généralement deux fichiers séparés. Certains modèles de développement placent l'ensemble des fichiers sur le serveur. D'autres les répartissent entre le client et le serveur. D'autres encore les répartissent entre plusieurs serveurs. L'objectif est de prendre conscience que les fichiers, où qu'ils soient stockés, font partie de la même solution ou sont connectés d'une manière ou d'une autre. Cet aspect est important pour diverses raisons. Du point de vue de l'administration, il devient beaucoup plus facile de gérer et d'administrer les solutions pour les déplacements, la sécurité et les sauvegardes. Du point de vue du développement, cela permet de disposer d'une méthode cohérente représentant visuellement le groupement d'une solution.

Connectivité externe : pour interagir avec un système de gestion de bases de données relationnel, vous devez vous méfier des conflits de noms de fichiers dûs à des caractères spéciaux ou des mots réservés. En outre, quand l'accès aux fichiers FileMaker se fait à partir d'autres technologies telles que ODBC, JDBC ou XML, vous devez spécifier le nom du fichier.

1.3 Recommandations intégrant les standards (fichiers) :

1. Le nom d'un fichier ne doit utiliser que les caractères
 - alphabétiques majuscules et minuscules aA - zZ ;
 - numériques 0, 1, 2, 3, 4, 5, 6, 7, 8 et 9 ;
 - de tiret bas simple et double « _ » « __ ».
2. Il ne doit pas PAS contenir d'espaces.
3. Il ne doit PAS commencer par une valeur numérique.
4. Le nom d'un fichier ne doit PAS servir à indiquer la version.
5. Il ne doit PAS contenir de points, hormis celui de l'extension.
6. Les noms de fichiers doivent être systématiquement au singulier ou au pluriel.
7. Les différents mots composant le nom d'un fichier doivent être séparés de manière cohérente, à l'aide de l'une des méthodes suivantes.



Méthodes recommandées pour la séparation des mots (noms de fichiers) :

casseChameauMinuscule	monFichier (myFileName)
CasseChameauMajuscule	MonFichier (MyFileName)
Tiret bas simple (minuscules)	mon_fichier (my_file_name)
Tiret bas simple (majuscule à chaque initiale)	Mon_Fichier (My_File_Name)
Tiret bas simple (MAJUSCULES)	MON_FICHER (MY_FILE_NAME)

- Le nom d'un fichier doit systématiquement porter l'extension .fp7 (en minuscules), quelle que soit la plateforme. Les solutions d'exécution, en revanche, peuvent utiliser n'importe quelle extension, mais veillez à ne pas utiliser une extension enregistrée par ailleurs.
- Dans une solution constituée d'un seul fichier, toutes les tables sont contenues dans le fichier. Il n'y a pas d'autres recommandations.
- Dans une solution constituée de plusieurs fichiers, les tables sont contenues dans un groupe de fichiers. Dans une solution composée de plusieurs fichiers, il y a un ou plusieurs fichiers principaux et des fichiers secondaires. Si le groupement est intéressant dans une solution constituée de plusieurs fichiers, c'est principalement parce qu'il permet d'identifier les fichiers qui composent la solution. Souvent, il est possible d'utiliser des noms naturels. Par exemple, imaginons une solution constituée de 4 fichiers nommés comme suit :

- Guichet.fp7
- BILLETS.fp7
- Rapports.fp7
- Stock.fp7

Les noms de ces fichiers n'indiquent pas clairement qu'ils font partie du même groupe. Il suffit de quelques modifications pour que le groupement apparaisse clairement avec ces noms naturels. Toutefois, il n'est pas toujours possible d'utiliser des noms naturels. Vous pouvez aussi décider d'adopter une approche de dénomination plus cohérente, même s'il est possible d'utiliser des noms naturels dans certains cas. Prenons le même exemple avec la recommandation intégrant les standards. Cette recommandation facilite les choses à plusieurs égards. Tout d'abord, elle identifie clairement le fichier principal. Ensuite, elle identifie clairement les fichiers de support. Enfin, elle indique que cette série de fichiers constitue un groupe.

- Guichet_hd.fp7
- hd__BILLETS.fp7
- hd__Rapports.fp7
- hd__Stock.fp7

10.1. Fichier(s) de support :

Chaque fichier de support doit contenir un préfixe, défini ici comment l'identificateur LSI (Logical Solution Identifier ou identificateur de solution logique). Chaque identificateur de solution logique doit être utilisé pour chaque fichier de support de la solution. Chaque indicateur LSI doit suivre les conventions générales et être séparé du nom par deux caractères de tiret bas « __ ». Ces deux caractères de tiret bas permettent de distinguer plus facilement l'identificateur de solution logique du nom de fichier, qui peut par ailleurs utiliser des caractères de tiret bas simples comme séparateurs de mots. L'identificateur de solution logique (LSI) peut être de n'importe quelle longueur mais il doit être identique dans l'ensemble de la solution.

Syntaxe (fichiers de support)

<<LSI>>[_]monFichier[.fpX]

Voir la [légende de la syntaxe](#) pour une description de la syntaxe.

- LSI (Logical Solution Identifier, *identificateur de solution logique*) – requis, défini par le développeur. Indique que les fichiers sont connectés de manière logique.
- « __ » – requis
Sépare l'indicateur LSI du nom du fichier



- monFichier (myFileName) –
Nom de fichier descriptif
- .fpX – requis
Tout fichier doit se terminer par « . », (point), suivi de l'extension fpX extension, où X est la version du format de fichier FileMaker. Exemple : .fp7

10.2. Fichiers primaires

Vous pouvez choisir d'inclure l'indicateur LSI dans le nom des fichiers primaires sous forme de préfixe ou de suffixe. Dans un cas comme dans l'autre, l'indicateur LSI doit toujours être séparé du nom par deux caractères de tiret bas « __ ».

Syntaxe (fichiers primaires) :

monFichier[__] { LSI } [. fpX]

{ LSI } [__] monFichier [. fpX]

Voir la [légende de la syntaxe](#) (annexe D) pour obtenir une description de la syntaxe.

- Indicateur LSI sous forme de préfixe – requis, défini par le développeur
Indique que les fichiers sont connectés de manière logique
- « __ » – requis
Sépare l'indicateur LSI du nom du fichier
- monFichier – nom de fichier descriptif
- Indicateur LSI sous forme de suffixe – requis, défini par le développeur
Indique que les fichiers sont connectés de manière logique
- .fpX – requis
Tout fichier doit se terminer par « . » (point), suivi de l'extension fpX extension, où X est la version du format de fichier FileMaker.
Exemple : .fp7

10.3. Exemples

- MaSolution__xxx.fp7 – fichier primaire
- xxx__MonFichierSupportA.fp7 – fichier de support
- xxx__MonFichierSupportB.fp7 – fichier de support
ou
- xxxx__maSolution.fp7 – fichier primaire
- xxxx__fichierSupportA.fp7 – fichier de support
- xxxx__fichierSupportB.fp7 – fichier de support
ou
- Ma_Solution__ABC.fp7 – fichier primaire
- ABC__clients.fp7 – fichier de support
- ABC__factures.fp7 – fichier de support

1.4 Considérations annexes :

- **Fichiers empruntés** : dans certains cas, une solution peut inclure une référence à un autre fichier qui n'est pas associé individuellement à une solution précise. Par exemple, un annuaire d'employés peut être utilisé dans un grand nombre de solutions. Dans pareil cas, il est conseillé de traiter le fichier « emprunté » comme sa propre solution, et non de l'incorporer dans une solution donnée.
- **Stockage des fichiers clients** : pour des raisons de clarté et d'organisation, il est conseillé de placer les fichiers de support résidant sur un disque dur local dans un sous-dossier/répertoire du dossier/répertoire contenant les fichiers primaires. Quand les fichiers d'une solution sont répartis entre le serveur et le client, il est conseillé de créer un dossier de programmes et de stocker les fichiers primaires côté client à la racine de ce dossier, tous les fichiers de support étant stockés dans un sous-dossier.



2 ▶ Dénomination des tables

Les noms de tables ne sont guère exposés au sein de l'environnement de développement.

L'interaction directe avec les tables se limite généralement à associer une table à une occurrence de table se trouvant dans le graphique de liens. Toutes les interactions avec les données se font par l'intermédiaire de l'occurrence de table. Il est important d'opérer cette distinction. Alors que de nombreuses boîtes de dialogue utilisent le terme « Table », les opérations portent en fait sur l'occurrence de table.

Pour sélectionner des noms de tables, vous devez prendre en considération un certain nombre de facteurs. Comme pour n'importe quel autre type de nom, vous devez choisir un jeu de caractères qui ne nuise pas à la connectivité entre la solution et des technologies telles que ODBC, JDBC et XML. Pour les noms de tables eux-mêmes, ce n'est pas directement important car vous ne ferez pas référence aux tables directement. Toutefois, si vous choisissez d'utiliser le nom de la table source dans le nom de l'occurrence de table, qui est utilisé pour les appels de connectivité et référencé à travers toutes les interactions, vous devrez être plus sélectif. Bien entendu, toutes les solutions n'utilisent pas une connectivité externe mais au fur et à mesure que les solutions évoluent, il est préférable de les préparer et de prévoir cette possibilité.

Pour des raisons de cohérence et de professionnalisme, vous devez sélectionner une casse et vous y tenir. Vous devez également choisir une manière de séparer les différents mots composant le nom de votre table. Vous devez par ailleurs déterminer comment le nom de la table lui-même sera séparé syntaxiquement des métadonnées ajoutées sous forme de préfixe ou de suffixe. Vous pouvez utiliser un système de séparation syntaxique pour le nom des tables pour distinguer les noms par fonction ou par catégorie. Par exemple, vous pouvez signaler une table de « lien » ou de « session » à l'aide d'un préfixe, sous la forme « j__NomTable » ou « ses__nom_table ». Dans les deux cas, le séparateur syntaxique est un double tiret bas « __ ». Les noms de tables doivent être systématiquement au singulier ou au pluriel. Demandez-vous s'il peut être intéressant d'utiliser des espaces pour organiser le contenu de l'onglet Table. Cela peut être intéressant pour gérer un grand nombre de tables. Enfin, le nom d'une table peut comporter jusqu'à 100 caractères mais une telle longueur n'est guère pratique pour diverses raisons.

2.1 Objectifs :

- ✓ Jeu de caractères recommandé
- ✓ Convention recommandée pour la casse
- ✓ Recommandations d'usage concernant le singulier et le pluriel
- ✓ Recommandations concernant la longueur des noms de tables
- ✓ Utilisation d'espaces pour pallier les problèmes d'organisation
- ✓ Conseils pour nommer les tables suivant leur type (lien, utilitaire, système, référence, etc.)

2.2 Définition du problème :

Jeu de caractères recommandé : les noms de tables ne sont guère concernés par les problèmes de caractères dans la mesure où ils sont peu utilisés dans une application. Toutefois, vous devez réfléchir aux caractères utilisés dans la mesure où les noms de tables peuvent être utilisés ailleurs, comme dans les noms d'occurrences de tables, où ils sont référencés.

Convention recommandée pour la casse : la casse choisie pour les noms de tables peut varier suivant la manière dont ceux-ci seront utilisés dans les occurrences de tables et dans les modèles. En pratique, le choix entre casseChameauMinuscule, CasseChameauMajuscule et MAJUSCULES est simplement une question de préférence. Vous devez toutefois prendre en considération quelques facteurs. Tout d'abord, comptez-vous inclure le nom de la table dans le nom de l'occurrence de table ? Si tel est le cas, vous devez prendre en considération la convention utilisée pour les noms d'occurrences de tables. Si vous utilisez la casse MAJUSCULES, il vous faut déterminer comment séparer les mots (à l'aide d'un espace, d'un tiret bas simple ou double, etc.). En outre, si vous utilisez des caractères de tiret bas pour séparer les mots, qu'utiliserez-vous pour séparer la syntaxe dans les occurrences de tables, les modèles et les autres endroits où vous devrez inclure des noms de tables ? Pour faire vos choix, réfléchissez à l'endroit où les noms seront utilisés et à la manière dont ils seront utilisés.



Problèmes de longueur de nom de table : le nom d'une table peut comporter jusqu'à 100 caractères. Toutefois, souvent, une telle longueur n'est guère pratique. L'approche à suivre a déjà été évoquée dans la section « Convention recommandée pour la casse ». Le nom de la table peut être inclus dans le nom de l'occurrence de table. Il allonge donc d'autant le nom de l'occurrence de table. Le nom obtenu peut être plus long que ce que certaines boîtes de dialogue permettent d'afficher avec certaines opérations, suivant la convention suivie pour les occurrences de tables. Pour déterminer la longueur de vos noms de tables, vous avez donc intérêt à réfléchir aux endroits où les conventions seront appliquées.

Recommandations liées à l'usage du singulier et du pluriel : la distinction entre le singulier et le pluriel n'est faite ici que pour signaler qu'en tant que développeur, vous devez faire un choix et vous y tenir.

Subdivision des tables en catégories : certains développeurs aiment inclure des métadonnées dans les noms de tables. Souvent, cela permet de subdiviser les tables en catégories, et d'indiquer que telle table contient tel type de données ou joue tel rôle au sein de la solution. Cela peut être le cas des tables de session, de référence ou de lien, ou encore des tables utilitaires. Dans FileMaker, il n'y a pas de moyen de signaler ce type de catégorie. Aussi certains développeurs choisissent-ils d'inclure ces métadonnées dans le nom de la table.

Tables d'espacement : certains développeurs utilisent des tables spéciales appelées « tables d'espacement » qui ne contiennent pas de rubriques mais font simplement office de séparateurs entre les catégories de tables. Cette technique n'a pas d'intérêt pour les solutions de petite taille mais elle peut en avoir quand une solution se développe et intègre de plus en plus de tables.

2.3 Recommandations intégrant les standards (tables) :

1. Le nom d'une table ne doit utiliser que les caractères
 - alphabétiques majuscules et minuscules aA - zZ ;
 - numériques 0, 1, 2, 3, 4, 5, 6, 7, 8 et 9 ;
 - de tiret bas simple et double « _ » « __ ».
2. Il ne doit pas PAS contenir d'espaces.
3. Les noms de tables doivent être systématiquement au singulier ou au pluriel.
4. Les différents mots composant le nom d'une table doivent être séparés de manière cohérente, à l'aide de l'une des méthodes suivantes.

Méthodes recommandées pour la séparation des mots (noms de tables) :

casseChameauMinuscule	maTable (myTableName)
CasseChameauMajuscule	MaTable (MyTableName)
Tiret bas simple (minuscules)	ma_table (my_table_name)
Tiret bas simple (majuscule à chaque initiale)	Ma_Table (My_Table_Name)
Tiret bas simple (MAJUSCULES)	MA_TABLE (MY_TABLE_NAME)

5. Le nom d'une table ne doit pas utiliser de mots réservés FileMaker.
6. Il ne doit PAS non plus utiliser de mots réservés SQL. Pour plus d'informations, reportez-vous à l'annexe A. Consultez également la documentation du SGBDR.
7. Subdivision des tables en catégories : comme indiqué dans la définition du problème, certains développeurs aiment indiquer le type ou la fonction de la table dans le nom lui-même. Si votre projet le nécessite, nous vous recommandons d'utiliser la syntaxe suivante :
8. Syntaxe : {fonction}[__]NomTable
Voir la [légende de la syntaxe](#) (annexe D) pour obtenir une description de la syntaxe.
 - fonction – facultatif, défini par le développeur
Fournit une subdivision en catégories définie par le développeur et permet une extension si nécessaire. En tant que développeur, vous pouvez définir les différentes catégories vous-même. Veillez à prendre en considération la casse, la longueur et la cohérence. Documentez vos choix en suivant les instructions de la section [Conformité](#) de ce document.



- « __ » – requis
Utilisez un double tiret bas pour séparer la fonction du nom lui-même.
- NomTable – nom descriptif de votre table.

2.4 Considérations annexes :

Utilisation des tables d'espace : quand une solution contient un grand nombre de tables, il peut être utile de subdiviser celles-ci en catégories. Comme l'environnement de développement n'offre pas de mécanisme à cet effet, c'est au développeur d'employer une technique pour y parvenir. L'exemple ci-dessous ne contient qu'un petit nombre de tables mais c'est suffisant pour illustrer notre propos. Si la liste était plus longue, l'avantage d'un tel système serait encore plus évident. Cet exemple montre des tables, dont certaines indiquent qu'elles font office de titre de catégorie. La figure ci-dessous permet de voir les entrées Comptabilité, Communication et « - Enregistrement - ». Il s'agit en fait de tables qui ne contiennent pas de rubriques. Vous pouvez les supprimer du graphique de liens puisqu'elles ne seront jamais utilisées dans notre solution.

Toutefois, elles facilitent l'organisation des tables, surtout dans une solution de grande taille. Dans notre exemple, le nom de la table est placé entre deux tirets (« - ») mais vous pouvez utiliser autre chose. L'essentiel est que les noms de tables ressortent plus. FileMaker peut faire apparaître un message d'avertissement si vous employez certains caractères mais de toute manière, vous n'utiliserez pas ces tables.

Nom
◆ -----
◆ - Comptabilite -
◆ Clients
◆ Inscriptions
◆ Envois
◆ Factures
◆ ElementsFacturations
◆ Produits
◆ - Communication -
◆ Evenements
◆ Parents
◆ Fournitures
◆ - Enregistrement -
◆ AssidueCours
◆ Employes
◆ Etudiants

Figure 1



3 ▷ Dénomination des rubriques

Les conventions suivies pour les noms de rubriques sont extrêmement variées. D'une manière générale, il y a deux modes de pensée : celui qui préconise l'emploi de noms « naturels », et celui qui utilise une certaine forme de notation pour faire apparaître des métadonnées sur la rubrique. Ces métadonnées permettent d'ajouter des informations sur la rubrique qui ne sont pas disponibles hors de la boîte de dialogue Définir la base de données. Les détails de la notation peuvent être extrêmement variés. Toutefois, l'idée sous-jacente est toujours la même.

Pour sélectionner des noms de rubriques, vous devez prendre en considération un certain nombre de facteurs. Comme pour n'importe quel autre type de nom; vous devez choisir un jeu de caractères qui ne nuise pas à la connectivité entre la solution et des technologies telles que ODBC, JDBC et XML. Pour des raisons de cohérence et de professionnalisme, vous devez sélectionner une casse et vous y tenir. Vous devez également choisir une manière de séparer les différents mots composant le nom de la rubrique. Là encore, la cohérence et le professionnalisme sont l'objectif.

Vous devez par ailleurs déterminer comment le nom de la rubrique lui-même sera séparé syntaxiquement des métadonnées ajoutées sous forme de préfixe ou de suffixe. La séparation syntaxique est très importante pour les noms de rubriques en raison de la diversité des problèmes à éviter. Il faut prendre en considération la subdivision en catégories et indiquer les rubriques clés/sources, les rubriques utilitaires ou les rubriques du développeur. L'objectif est de pouvoir distinguer clairement les éléments de notation sans avoir à déchiffrer le code. Cela étant, si vous choisissez d'utiliser une notation, il faut qu'elle soit compréhensible universellement. Le document « Conventions de développement FileMaker » offre deux éléments pour vous aider. Premièrement, il contient une syntaxe commune indiquant à quel endroit placer les différentes parties. Les composantes les plus communes devraient y figurer.

Deuxièmement, au sein de cette syntaxe, ce document permet au développeur d'ajouter ses propres extensions ou personnalisations afin de répondre à ses besoins propres.

Il est important de vous demander à quel endroit les composants de la syntaxe sont placés et pourquoi il est recommandé de les organiser de la sorte. Vous devez par ailleurs choisir le pluriel ou le singulier pour les noms et vous tenir à ce choix. Pour les noms de rubriques, il est important de penser aux moyens d'effectuer des groupements, et d'utiliser par exemple des rubriques `Nom_Prenom` et `Nom_Nom` plutôt que `Prenom_Nom`, `Nom_Nom`. Certains développeurs utilisent le tri par nom de rubrique plutôt qu'un ordre de tri personnalisé. Le choix du tri n'est sans doute pas un critère déterminant pour la dénomination des fichiers mais cette fonction intégrée permet parfois de gagner du temps pour l'organisation. Il peut être intéressant d'utiliser des espaces pour organiser le contenu de votre liste de rubriques, surtout quand les rubriques sont très nombreuses et quand un tri manuel est plus pratique pour en afficher la liste.

Il est évident que vos pratiques de développement ne s'intègrent pas nécessairement au cadre défini par ce document. C'est donc à vous de choisir dans ce domaine. Si vous devez utiliser un système de notation, examinez attentivement cette recommandation.

3.1 Objectifs :

- ✓ Jeu de caractères recommandé
- ✓ Convention recommandée pour la casse
- ✓ Problèmes de nom de rubrique
- ✓ Options de dénomination possibles en cas d'absence de métadonnées dans l'environnement de développement
- ✓ Limitations empêchant de masquer les rubriques utilitaires dans une interface utilisateur conçue pour pallier les insuffisances de l'environnement de développement
- ✓ Séparation du nom de rubrique et des métadonnées
- ✓ Options possibles avec les espaces
- ✓ Options possibles pour le groupement et la lisibilité des rubriques



3.2 Définition du problème :

Jeu de caractères recommandé : si des caractères inappropriés sont utilisés, différents problèmes peuvent se poser. Il est particulièrement important que les caractères soient valides dans les calculs.

Le développeur doit choisir des caractères qui n'entrent pas en conflit avec les solutions FileMaker, éviter les problèmes de variables et se méfier des caractères qui peuvent poser problème lors des échanges avec des systèmes de gestion de bases de données relationnels.

Convention recommandée pour la casse : le mode de séparation des mots au sein des noms de rubriques a un impact sur la cohérence générale des conventions. Par exemple, si vous choisissez de séparer les mots par un « _ » (tiret bas), il sera difficile de distinguer le nom de la rubrique du suffixe ou préfixe éventuellement utilisés pour représenter les métadonnées de la rubrique. On utilise généralement « g_ » devant le nom de la rubrique.

Longueur recommandée pour les noms de rubriques : le nom d'une rubrique FileMaker peut contenir jusqu'à 100 caractères. Toutefois, souvent, une telle longueur n'est guère pratique. A partir de la version 8, la plupart des boîtes de dialogue peuvent afficher le nom complet sans problème. Toutefois, les boîtes de dialogue Trier, Exporter et Modifier les requêtes de recherche ne prennent en charge qu'un nombre limité de caractères. La boîte de dialogue « Trier » est limitée à 20 caractères sous OS X et à 26 caractères sous Windows. La boîte de dialogue « Exporter » est limitée à 30 caractères sous OS X et à 28 caractères sous Windows. La boîte de dialogue « Modifier » les requêtes de recherche est limitée à 26 caractères sous OS X et à 27 caractères sous Windows. Ces limites dépendent de la largeur des caractères. Par exemple, « 8 » est plus large que « 1 ». Les limites sont indiquées pour une chaîne de caractères de pleine largeur. Le document « Conventions de développement FileMaker » ne fait pas de recommandations en matière de longueur mais il mentionne ces limites à titre de référence.

Absence de métadonnées au sein de l'environnement de développement : la rubrique est l'un des principaux objets programmables sur lesquels le développeur travaille. A moins d'avoir une mémoire extraordinaire, il est difficile, voire impossible, de connaître les métadonnées d'une rubrique sans ouvrir la boîte de dialogue Définir la base de données. Quel est son type de données ? Quels éléments dépendent d'elles ? Comment stocke-t-elle les données ? Comment se comportera-t-elle dans un script ? Ces informations ne sont pas disponibles dès le départ. Vous pouvez pallier cet inconvénient à travers votre convention de dénomination. Si vous souhaitez faire apparaître ces informations dès le nom de la rubrique, vous devez indiquer comment procéder dans votre convention.

Impossibilité de masquer les rubriques utilitaires : les rubriques « utilitaires » sont des rubriques définies pour stocker des données temporaires ou des informations auxquelles l'utilisateur final n'a généralement pas accès. Dans ce cas, le développeur doit faire en sorte que l'utilisateur final ne voie pas ces rubriques dans les boîtes de dialogue qui présentent des rubriques. Alors que FileMaker 8 permet de ne faire apparaître que les rubriques d'un modèle pour les opérations d'exportation et de tri, pour un certain nombre d'autres opérations, toutes les rubriques sont affichées. Dans ce cas, comme il n'est pas possible de masquer des rubriques, le développeur peut les déplacer au sein de la liste. Généralement, il les décale tout au bas de la liste de rubriques. Depuis la version 7, FileMaker utilise l'ordre de tri Unicode pour les noms de rubriques. Hormis le tri manuel des noms de rubriques, qui peut être fastidieux et prendre beaucoup de temps, la seule solution consiste à établir une convention de dénomination ayant pour effet de déplacer les rubriques utilitaires au bas de la liste. Comme les tris reposent désormais sur l'ordre Unicode, on considère généralement le caractère « z », « zz » ou un caractère Unicode permettant de déplacer ces rubriques au bas de la liste comme le meilleur moyen de faire apparaître une rubrique en dernière position. En tant que développeur, vous devez déterminer si c'est important pour vous ou non.

Séparation du nom de rubrique et des métadonnées : il est important que les développeurs actuels et futurs et l'utilisateur final sachent clairement ce qui est utilisé pour séparer le nom de la rubrique des métadonnées/de la notation. En outre, la notation et le nom de la rubrique doivent être identifiés clairement.

3.3 Recommandations intégrant les standards (rubriques) :

Le document « Convention de développement FileMaker » distingue trois (3) grandes catégories de rubriques : les rubriques générales, les rubriques clés et les rubriques utilitaires.



3.3.1 Rubriques générales

Une rubrique générale sert au stockage de données générales.

1. Son nom ne doit utiliser que les caractères
 - alphabétiques majuscules et minuscules : a-z ou A-Z ;
 - numériques : 0,1,2,3,4,5,6,7,8 ou 9 ;
 - de tiret bas simple et double « _ » « __ ».
2. Il ne doit pas PAS contenir d'espaces.
3. Il ne doit PAS commencer par une valeur numérique.
4. Les noms de rubriques générales doivent être systématiquement au singulier ou au pluriel.
5. Les différents mots composant le nom d'une rubrique générale doivent être séparés de manière cohérente, à l'aide de l'une des méthodes suivantes.

Méthodes recommandées pour la séparation des mots (noms de rubriques) :

casseChameauMinuscule	maRubrique
CasseChameauMajuscule	MaRubrique
Tiret bas simple (minuscules)	ma_rubrique
Tiret bas simple (majuscule à chaque initiale)	Ma_Rubrique
Tiret bas simple (MAJUSCULES)	MA_RUBRIQUE

6. La liste des noms de rubriques doit être triée par nom.
7. Veillez à ne pas faire commencer le nom d'une rubrique par les caractères servant d'indicateur de rubrique utilitaire, qui doivent être réservés. Pour les rubriques utilitaires, les caractères « z », « zz » et tous les autres caractères Unicode placent les rubriques utilitaires au bas de la liste si les rubriques sont triées par nom.
8. Les noms de rubriques générales doivent utiliser le groupement de rubriques quand c'est possible.
Exemples :
 - Nom_Prenom
 - Nom_Nom
 - AdresseVille
 - AdresseRegion
 - AdresseCodePostal
 - passeportNumero
 - passeportPaysDelivrance
 - passeportDateExpiration
 - passeportNom
9. Les rubriques dérivées ou calculées dont le développeur ne veut pas faire des rubriques utilitaires mais pour lesquelles il souhaite inclure des éléments de notation doivent suivre les recommandations faites pour les suffixes des rubriques utilitaires.

Par exemple, `FactureTotal` est une rubrique calculée mais vous ne souhaitez pas la rendre invisible de l'utilisateur. Vous souhaitez en revanche inclure des éléments de notation dans son nom. Dans ce cas, vous pouvez nommer la rubrique `FactureTotal__lcn`, ce qui indique que la rubrique est stockée localement, qu'elle contient des informations calculées et que le résultat est de type numérique. Pour plus d'informations, reportez-vous à la section dénomination des rubriques utilitaires.

3.3.2 Rubriques clés/sources

Contrairement à bien des systèmes de gestion de bases de données relationnels reposant sur SQL, FileMaker n'impose pas un certain nombre de contraintes d'intégrité relationnelles. Par exemple, il n'impose pas de créer une clé primaire pour une table. Au sein de l'application, rien n'indique, visuellement ou par le contexte, qu'une



rubrique est une rubrique clé/source. Généralement, il est souhaitable d'utiliser des rubriques clés ou sources dans vos solutions. Vous pouvez utiliser des éléments de notation pour identifier précisément les rubriques clés.

Si vous choisissez d'utiliser des éléments de notation pour les rubriques clés, la syntaxe suivante est recommandée si vous souhaitez respecter les standards.

Directives sur la syntaxe des rubriques clés/sources :

- Prévoyez une manière universellement accessible, compréhensible et cohérente de représenter les métadonnées de la rubrique source.
 - Identifiez la rubrique clé/source
 - Identifiez la fonction de la rubrique clé/source
 - Identifiez le type de stockage de la rubrique clé/source
 - Identifiez le type de données de la rubrique clé/source
 - Triez les rubriques clés/sources en haut de la liste des rubriques tout en maintenant un tri des rubriques par nom (ordre alphabétique). Cette approche limite les opérations de tri manuel pour le développeur et présente l'ordre de tri attendu à l'utilisateur de la solution.
 - Utilisez les caractères de notation généralement considérés comme acceptables, tels que « p » pour la clé primaire, et « f » (abréviation de « foreign ») pour la clé étrangère, ou alignez-vous sur l'environnement de développement FileMaker.
1. Le nom d'une rubrique clé/source ne doit utiliser que les caractères
 - alphabétiques majuscules et minuscules : a-z ou A-Z ;
 - numériques : 0, 1, 2, 3, 4, 5, 6, 7, 8 et 9 ;
 - de tiret bas simple et double « _ » « __ ».
 2. Il ne doit pas PAS contenir d'espaces.
 3. Il ne doit PAS commencer par une valeur numérique.
 4. Les noms de rubriques clés/sources doivent être systématiquement au singulier ou au pluriel.
 5. Les différents mots composant le nom d'une rubrique clé/source doivent être séparés de manière cohérente, à l'aide de l'une des méthodes suivantes.

Méthodes recommandées pour la séparation des mots (noms de rubriques) :

casseChameauMinuscule	maRubrique
CasseChameauMajuscule	MaRubrique
Tiret bas simple (minuscules)	ma_rubrique
Tiret bas simple (majuscule à chaque initiale)	Ma_Rubrique
Tiret bas simple (MAJUSCULES)	MA_RUBRIQUE

6. Syntaxe : [k]<fonction>(stockage)(type)[__]NomDescriptif
Voir la [légende de la syntaxe](#) pour une description de la syntaxe.
 - « _ » – requis
Un tiret bas simple fait apparaître les rubriques clés en haut de toutes les boîtes de dialogue dans lesquelles une liste des rubriques est affichée et dans le graphique des liens.
 - k – minuscule ; requis
Indique qu'il s'agit d'une rubrique clé/source
 - fonction – minuscules ; requis ; utilise les valeurs fournies le cas échéant
Indique la catégorie de la rubrique clé et/ou sa fonction.
 - ▶ p – clé primaire (Primary Key)
 - ▶ f – clé étrangère (Foreign Key)
 - ▶ a – autre clé (Alternate Key)



- ▶ c – clé composée / concaténée / calculée (Compound / Concatenated / Calculated Key)
- ▶ m – clé multiligne (Multi-Line Key)
- stockage – minuscules ; facultatif ; utilise les valeurs fournies s’il y a lieu
Indique le type de stockage de la rubrique
 - ▶ l = stockage local (« l » minuscule)
 - ▶ g = stockage global
- type – minuscules ; facultatif ; utilise les valeurs fournies s’il y a lieu
Indique le type de rubrique
 - ▶ t = texte (Text)
 - ▶ n = valeur numérique (Number)
 - ▶ d = date (Date)
 - ▶ i = heure (Time)
 - ▶ m = horodatage (Time Stamp)
- « __ » – requis
Utilisez un tiret bas double pour séparer la syntaxe de la rubrique clé du nom descriptif de la rubrique
- NomDescriptif- CasseSelectionnee
Nom descriptif de la rubrique clé
- Exemples :

Format intégral utilisant l’ensemble des métadonnées en option

Format minimal sans métadonnées optionnelles

<code>_kplt__FactureID</code>	<code>_kp__FactureID</code>
<code>_kcgt__Participants_Selectionnes</code>	<code>_kc__Participants_Selectionnes</code>
<code>_kflt__ClientId</code>	<code>_kf__ClientId</code>
<code>_kmgmt__VueJoursSelectionnes</code>	<code>_km__VueJoursSelectionnes</code>
<code>_kflt__Facture_ligne_Article</code>	<code>_kf__Facture_ligne_Article</code>

3.3.3 Rubriques utilitaires

La convention de dénomination des rubriques utilitaires vise à répondre aux différents besoins des développeurs et à résoudre certains problèmes de dénomination particulièrement fréquents avec les rubriques qui ne doivent généralement pas être accessibles de l'utilisateur final.

Directives sur les rubriques utilitaires :

- Méthode pour offrir une manière universellement accessible, compréhensible et cohérente, de représenter les métadonnées de la rubrique source.
 - Identification des rubriques utilitaires
 - Identification des rubriques utilitaires personnalisables
 - Identification du type de stockage des rubriques utilitaires
 - Identification du type de données des rubriques utilitaires
 - Identification d’une rubrique utilitaire comme une rubrique multivaluée
 - Méthode pour faire apparaître les rubriques utilitaires au bas de la liste des rubriques tout en effectuant un tri des rubriques par nom (ordre alphabétique). Cette approche limite les opérations de tri manuel pour le développeur et met les rubriques hors de portée de l'utilisateur.
1. Le nom d’une rubrique utilitaire ne doit utiliser que les caractères
 - alphabétiques majuscules et minuscules : a-z ou A-Z ;
 - numériques : 0, 1, 2, 3, 4, 5, 6, 7, 8 et 9 ;
 - de tiret bas simple et double « _ » « __ ».



2. Il ne doit pas PAS contenir d'espaces.
3. Il ne doit PAS commencer par une valeur numérique.
4. Les noms de rubriques utilitaires doivent être systématiquement au singulier ou au pluriel.
5. Les différents mots composant le nom d'une rubrique utilitaire doivent être séparés de manière cohérente, à l'aide de l'une des méthodes suivantes.

Méthodes recommandées pour la séparation des mots (rubriques utilitaires) :

casseChameauMinuscule	maRubrique
CasseChameauMajuscule	MaRubrique
Tiret bas simple (minuscules)	ma_rubrique
Tiret bas simple (majuscule à chaque initiale)	Ma_Rubrique
Tiret bas simple (MAJUSCULES)	MA_RUBRIQUE

6. Syntaxe : <zz>{fonction}[__]NomDescriptif[__(stockage)(type)(multivaluées)]
Voir la [légende de la syntaxe](#) pour une description de la syntaxe.
 - « zz » ou caractère Unicode permettant d'afficher les rubriques utilitaires en bas – requis
Indique que la rubrique est une rubrique clé pour le développeur et la fait apparaître en bas dans toutes les boîtes de dialogue qui contiennent des noms de rubriques.
 - fonction – minuscules ; facultatif , défini par le développeur
Fournit une subdivision en catégories définie par le développeur et permet une extension si nécessaire. En tant que développeur, vous pouvez définir les différentes catégories vous-même. Veillez à prendre en considération la casse, la longueur et la cohérence. Documentez vos choix en suivant les instructions de la section [Conformité](#) de ce document.
 - « __ » – requis
Le tiret bas double matérialise la séparation entre le préfixe et le nom descriptif
 - NomDescriptif
Nom choisi par le développeur. Suit les mêmes recommandations que les rubriques générales.
 - « __ » – requis
Le tiret bas double sert à indiquer la fin du nom descriptif et le début d'une notation de suffixe.
 - stockage – minuscules ; requis ; utilise les valeurs fournies s'il y a lieu
Indique le type de stockage de la rubrique
 - ▶ l = stockage local (l minuscule)
 - ▶ g = stockage global
 - type – minuscules ; requis ; utilise les valeurs fournies s'il y a lieu
Indique le type des données stockées ou retournées.

Types de résultats non calculés

- ▶ xt = texte (Text)
- ▶ xn = valeur numérique (Number)
- ▶ xd = date (Date)
- ▶ xi = heure (Time)
- ▶ xm = horodatage (Time Stamp)
- ▶ xr = multimédia (Container)

Types de résultats calculés

- ▶ ct = texte (Text)
- ▶ cn = valeur numérique (Number)
- ▶ cd = date (Date)
- ▶ ci = heure (Time)



- cm = horodatage (Time Stamp)
- cr = multimédia (Container)

Type de résultat statistique

- xs = statistique (Summary)

multivaluées – minuscules ; requis ; utilise les valeurs fournies s'il y a lieu
Indique que les données sont stockées avec des répétitions.

- p = multivaluées

3.3.4 Référence de notation de rubriques

Référence de notation de rubriques

	Texte	Valeur numérique	Date	Heure	Horodatage	Multimédia
--	-------	------------------	------	-------	------------	------------

Rubriques clés (forme longue : avec toutes les métadonnées)

Clé : primaire	_kplt	_kpln	_kpld	_kpli	_kplm	N/A
Clé : étrangère	_kflt	_kfln	_kfld	_kfli	_kflm	N/A
Clé : autre	_kalt	_kaln	_kald	_kali	_kalm	N/A
Clé : composée	_kc1t	_kc1n	_kc1d	_kc1i	_kc1m	N/A
Clé : multilingue	_kmlt	_kmln	_kml d	_kml i	_kml m	N/A
Clés multivaluées (ajoutez un « p » minuscule pour mettre fin à la notation)	p	p	p	p	p	N/A

Rubriques utilitaires / rubriques générales et calculées (forme complète : avec toutes les métadonnées)

Générale (stockage local)	__lxt	__lxn	__lxd	__lxi	__lxm	__lxr
Générale (stockage global)	__gxt	__gxn	__gxd	__gxi	__gxm	__gxr
Calculée (stockage local)	__lct	__lcn	__lcd	__lci	__lcm	__lcr
Calculée (stockage global)	__gct	__gcn	__gcd	__gci	__gcm	__gcr
Multivaluée (ajoutez un « p » minuscule pour mettre fin à la notation)	p	p	p	p	p	p
Statistique (stockage local)	N/A	__lxs	N/A	N/A	N/A	N/A

3.4 Considérations annexes :

Séparateurs de rubriques : de nombreux développeurs utilisent une rubrique « non fonctionnelle » comme séparateur dans les listes de rubriques. Ainsi, les éléments espacés sont plus faciles à gérer dans les listes particulièrement longues. Cette approche permet une certaine forme de division dans un environnement qui n'offre pas de mécanisme à cet effet. Si vous utilisez cette technique, nous vous recommandons d'employer une rubrique de type numérique configurée pour le stockage global pour réduire la taille de stockage.



4 ▷ Occurrences de tables

Depuis FileMaker 7 et l'apparition du graphique des liens, le produit offre de nouvelles fonctionnalités mieux adaptées aux systèmes complexes et de grande taille. Toutefois, le graphique des liens peut devenir fastidieux et difficile à gérer quand il représente des applications complexes. Par exemple, dans un système complexe, la même table source peut être représentée plusieurs fois dans le graphique des liens.

En outre, il ne peut pas y avoir plus d'un chemin d'accès relationnel entre deux occurrences de tables. Le graphique peut par conséquent rapidement devenir très grand et la navigation risque d'être malaisée. Alors que le graphique des liens offre au développeur une interface utilisateur pour travailler sur les relations, il ne permet pas de grouper ou nommer de manière logique des groupes d'occurrences de tables. Aussi est-il difficile de savoir avec quelle occurrence de table travailler en effectuant une sélection dans une liste déroulante. Toutes les occurrences de tables sont classées en ordre alphabétique dans la liste, et il n'y a pas de moyen de les classer par ordre logique. Vous pouvez avoir à sélectionner une occurrence de table à différents endroits, par exemple, pour insérer des rubriques dans un modèle ou pour sélectionner une rubrique dans la boîte de dialogue de calculs. Dans pareil cas, il n'y a pas de moyen de voir la table source de l'occurrence, ni même son contexte dans le graphique des liens. Cette limite nous conduit à rechercher une convention de dénomination qui intègre la source et une description du contexte dans le nom de l'occurrence de table. Les conventions de dénomination proposées visent à indiquer à la fois la source et le contexte.

Il est évident que toute tentative de définition d'une convention de dénomination est intimement liée à la méthodologie utilisée par le développeur pour représenter les occurrences de tables. Aussi les rédacteurs de ce document ont-ils choisi d'illustrer plusieurs méthodes pour la construction des graphiques et les conventions de dénomination qui en découlent, plutôt que de proposer une seule convention qui privilégierait une méthode de développement particulière. Vous pourrez prendre ces éléments comme point de départ et les adapter pour développer vos propres solutions.

Enfin, quelle que soit la méthode utilisée, il est important que les spécificités de votre solution soient clairement identifiables sans trop d'efforts, même par un autre développeur. En pratique, il est souvent intéressant de partir d'une approche générale et de l'affiner ensuite. Les méthodes présentées ici ont un certain nombre de points communs. Par exemple, elles emploient toutes un format similaire. Elles sont groupées à l'aide d'un préfixe, suivi du nom de la table source. Le dernier élément, qui est facultatif, est un nom descriptif, dans lequel il est intéressant de préciser le contexte.

4.1 Objectifs :

- ✓ Terminologie
 - Définition du concept de table
 - Définition du concept d'occurrence de table (TO)
 - Définition du concept de table source/table de base
 - Définition du concept de groupe d'occurrences de tables fonctionnelles (TOG)
 - Définition du concept d'occurrence de table primaire (PTO)
- ✓ Jeu de caractères recommandé
- ✓ Convention recommandée pour la casse
- ✓ Problèmes de longueur d'occurrences de tables
- ✓ Mécanismes pour faciliter l'identification du contexte dans le graphique des liens, lors de la sélection d'une occurrence de table pour un calcul, de l'ajout de rubriques à un modèle ou de la définition du contexte d'un modèle



- ✓ Mécanismes pour résoudre le problème de groupement d'occurrences de tables à des fins de publication dans l'environnement de développement
- ✓ Mécanisme pour gérer la dépendance des calculs autonomes vis-à-vis des occurrences de tables primaires
- ✓ Mécanismes pour organiser les occurrences de tables
- ✓ Problèmes liés aux technologies de connectivité (XML, ODBC, JDBC) et aux mots réservés SQL

4.2 Définition du problème :

Définitions : Afin de clarifier ce document et d'être certains de bien comprendre les termes, nous allons définir un certain nombre de termes.

- **Table :** ensemble de données appartenant à une entité, par exemple des clients ou des prix unitaires. Un fichier de base de données contient une ou plusieurs tables, consistant en des rubriques et des enregistrements. Lors de la création d'une nouvelle table, une représentation visuelle, ou occurrence, de la table s'affiche dans le graphique des liens. Vous pouvez indiquer plusieurs occurrences (avec des noms uniques) de la même table de façon à travailler avec des liens complexes dans le graphique.
- **Occurrence de table ou TO (*Table Occurrence*) :** une occurrence de table désigne une instance d'une table dans le graphique des liens. N'oubliez pas que toutes les interactions avec une table au sein de l'environnement de développement interagissent avec les occurrences de tables. C'est le seul moyen de communiquer avec une table et son contenu.
- **Table source :** une occurrence de table est associée à une table. La table source est la table à laquelle l'occurrence de table est associée. Dans l'exemple ci-dessous, une occurrence de table appelée « AdmissionInterface_Classes_ListeClasse » est associée à une table source appelée Classes.

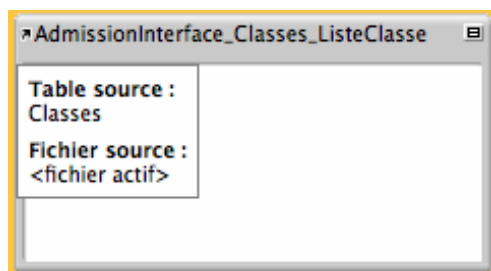


Figure 2

- **Occurrence de table primaire ou PTO (*Primary Table Occurrence*) :** l'occurrence de table primaire est une occurrence de table spéciale. C'est l'occurrence de table désignée pour être utilisée lors de la création de calculs « référencés en interne », c'est-à-dire les calculs dérivés de données contenues exclusivement dans la même table, et non issues de tables liées.
- **Table d'espacement :** système permettant de créer des tables étiquettes, ou tables de séparation, au sein de la boîte de dialogue « Définir la base de données ». Cette technique utilise des tables sans aucune rubrique pour regrouper et classer les tables et les occurrences de tables.

Jeu de caractères recommandé : les caractères autorisés par FileMaker sont soumis à un certain nombre de restrictions de base. En outre, certains caractères peuvent poser des problèmes de connectivité externe ou d'échange de données dans certains systèmes de gestion de base de données relationnels (SGBDR). En tant que développeur, vous devez avoir conscience des contraintes de FileMaker et de tous les systèmes externes avec lesquels vos solutions doivent interagir.

Convention recommandée pour la casse : cette recommandation n'a pas pour but de résoudre un problème particulier mais d'encourager l'utilisation d'un mode de dénomination uniforme pour les occurrences de tables. L'objectif est de sélectionner une méthode et de s'y tenir.

Séparation syntaxique recommandée : dans le domaine de la dénomination des occurrences de tables, la séparation syntaxique désigne la manière de différencier les préfixes et suffixes (métadonnées) éventuels du reste du nom de l'occurrence de table. Les développeurs utilisent fréquemment des préfixes ou suffixes pour



indiquer certains composants du nom ou pour signaler des caractéristiques d'identification. L'objectif est d'offrir une manière cohérente et reconnue universellement de différencier ces métadonnées sous forme de préfixes ou de suffixes du nom de l'occurrence de table.

Absence de contexte dans le graphique des liens lors de la sélection d'une occurrence de table pour un calcul, l'ajout de rubriques à un modèle ou de la définition du contexte d'un modèle. En dehors du graphique des liens, par exemple quand vous sélectionnez l'occurrence de table dans une liste, vous avez face à vous une longue liste d'occurrences de tables dans laquelle vous devez faire votre choix. Cette liste n'offre pas de système de groupement ni de tri à moins que votre convention de dénomination inclue le contexte.

Dépendance des calculs autonomes vis-à-vis des occurrences de tables primaires : quand vous créez des calculs, vous devez sélectionner le contexte à partir duquel le calcul sera évalué.

Pour les rubriques dont le résultat est dérivé sans référencer des données liées, on considère en général que ces calculs doivent toujours reposer sur le même contexte.

Absence de fonctions d'organisation pour les occurrences de tables : quelle que soit la méthode utilisée pour créer le graphique des liens, on considère généralement qu'il faut combiner un certain nombre d'occurrences de tables pour que la solution remplisse sa fonction, quelle que soit la connectivité vers d'autres occurrences de tables sur le graphique. Aussi est-il logique de chercher à les regrouper d'une manière ou d'une autre. Il n'existe pas de fonctionnalité à cet effet dans FileMaker, à moins de prévoir une convention de dénomination qui s'en charge.

4.3 Recommandations intégrant les standards (occurrences de tables) :

1. Le nom d'une occurrence de table ne doit utiliser que les caractères
 - alphabétiques majuscules et minuscules : aA ou zZ ;
 - numériques : 0, 1, 2, 3, 4, 5, 6, 7, 8 et 9 ;
 - de tiret bas simple et double « _ » « __ ».
2. Il ne doit pas PAS contenir d'espaces.
3. Il ne doit PAS commencer par une valeur numérique.
4. Il ne doit pas PAS contenir de points.
5. Les différents mots composant le nom d'une occurrence de table doivent être séparés de manière cohérente, à l'aide de l'une des méthodes suivantes.

Méthodes recommandées pour la séparation des mots (occurrences de tables)

casseChameauMinuscule	monNomOccurrence
CasseChameauMajuscule	MonNomOccurrence
Tiret bas simple (minuscules)	mon_nom_occurrence
Tiret bas simple (majuscule à chaque initiale)	Mon_Nom_Occurrence
Tiret bas simple (MAJUSCULES)	MON_NOM_OCCURRENCE

4.3.1 Méthode du groupement fonctionnel en araignée ou FSG (Functional Spider Grouping)

La première approche à suivre pour organiser les occurrences de tables consiste à faire commencer chaque nom par un préfixe indiquant le rôle de l'occurrence. Une fois cette méthode en place, l'image du graphique ressemble à une araignée, d'où le nom utilisé pour désigner cette approche. Notez que ce n'est pas la seule manière de désigner des groupes fonctionnels (reportez-vous notamment à l'exemple des groupes fonctionnels d'occurrences de tables).

L'idée de base du groupement fonctionnel est que les occurrences de tables ne sont pas représentées par nom ni par lien mais par groupe fonctionnel. Les types de fonctions possibles sont infinis, et ils dépendent des exigences de votre solution et de votre mise en œuvre (scripts, listes de valeurs, tables externes, etc.)



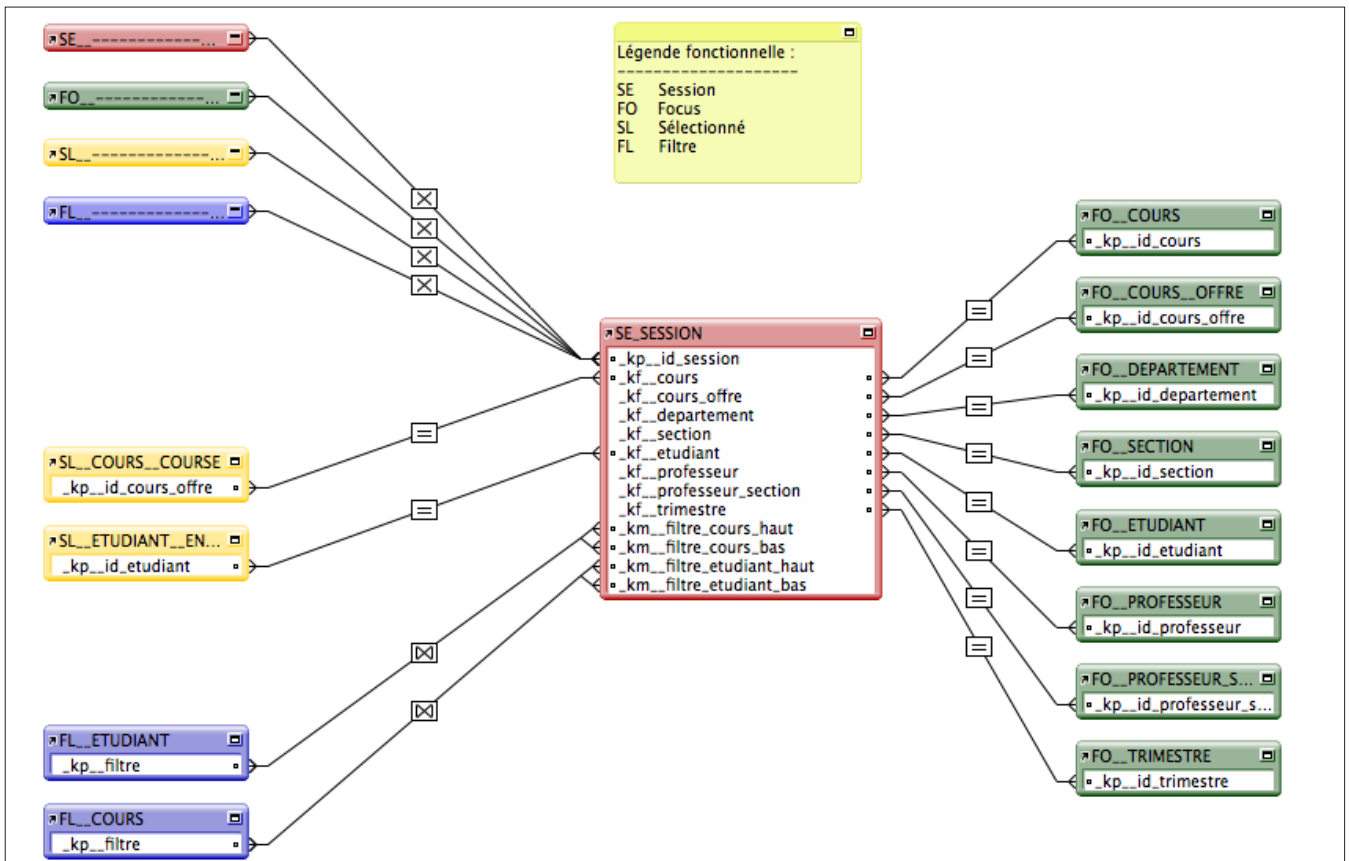


Figure 3

Dans cette méthode, les groupes fonctionnels sont identifiés par un « préfixe fonctionnel ». Ici, il s'agit d'un préfixe de deux lettres. Il est également possible d'utiliser des couleurs pour repérer plus facilement les différents groupes fonctionnels. Enfin, depuis FileMaker 8, il est également possible d'associer des notes au graphique. Ici, nous avons ajouté une note pour décrire ce que chaque groupe fonctionnel de deux lettres représente.

Le but n'est pas de décrire de manière trop détaillée le fonctionnement de la méthode du groupe fonctionnel en araignée mais de présenter les concepts principaux.

L'un des groupes fonctionnels du graphique est représenté par un préfixe de deux lettres intitulé FO, que nous utiliserons pour « Focus ». Toutes les tables de ce graphique qui commencent par ce préfixe de deux lettres seront groupées quand vous devrez entrer des informations liées à l'enregistrement qui a le focus.

Toutefois, comme il y aurait beaucoup trop de tables dans le graphique avec cette approche, une table de séparation est également utilisée pour permettre à ces groupes fonctionnels de ressortir. Ces tables d'espacement utilisent elles aussi le même préfixe de deux lettres et, associées à la méthode du groupement fonctionnel en araignée, elles facilitent la séparation et le groupement. Vous retrouvez ainsi plus facilement la fonction avec laquelle vous souhaitez travailler.



Table active ("SE_SESSION")
Tables liées
FL_-----
FL__COURS
FL__ETUDIANT
FO_-----
FO__COURS
FO__COURS__OFFRE
FO__DEPARTEMENT
FO__ETUDIANT
FO__PROFESSEUR
FO__PROFESSEUR_SECTION
FO__SECTION
FO__TRIMESTRE
SE_-----
SL_-----
SL__COURS__COURSE
SL__ETUDIANT__ENREGISTREMENT
Tables non liées
ER_-----
ER__COURS
ER__COURS__OFFRE
ER__DEPARTEMENT
ER__ETUDIANT
ER__INSCRIPTION
ER__PROFESSEUR
ER__PROFESSEUR_SECTION
ER__SECTION
✓ ER__TRIMESTRE
Définir la base de données...

Figure 4

L'exemple ci-dessus montre comment sélectionner une rubrique dans un modèle de session. Les séparateurs fonctionnels permettent de retrouver facilement un groupe fonctionnel donné dans la liste d'occurrences de tables.

4.3.1.1 AVANTAGES DU GROUPEMENT FONCTIONNEL EN ARAIGNÉE*

- Réduit le nombre d'occurrences de tables
- Offre un système d'organisation par fonction
- Fonctionne bien avec les solutions de taille moyenne ou petite (ce qui est le cas de la plupart des solutions)
- Peut également fonctionner avec les solutions de grande taille, suivant l'étendue des besoins fonctionnels. Plus les besoins fonctionnels auxquels le système doit répondre sont nombreux, moins cette méthode organisationnelle est intéressante.
- Prend en charge le modèle relationnel bidirectionnel en permettant aux modèles d'utiliser n'importe quel groupe d'occurrences de tables au sein du groupement fonctionnel en araignée. Cela permet de réduire le nombre d'occurrences de tables en ne limitant pas la visibilité du modèle à une occurrence de table du groupement fonctionnel en araignée.
- Offre un groupement fonctionnel au sein des menus d'occurrences de tables extérieurs au graphique des liens, suivant un ordre relativement bien groupé.

4.3.1.2 INCONVÉNIENTS DU GROUPEMENT FONCTIONNEL EN ARAIGNÉE

- Plus adapté aux méthodes reposant sur des tables externes. Ne fonctionne pas bien avec les solutions plus « ouvertes ».
- Le système de dénomination n'essaie pas d'exposer la structure hors du graphique des liens.
- Comme un modèle peut reposer sur n'importe quelle occurrence de table, les différentes occurrences de tables sont plus difficiles à repérer dans le graphique des liens.



4.3.1.3 GROUPEMENT FONCTIONNEL EN ARAIGNÉE : RECOMMANDATIONS INTÉGRANT LES STANDARDS

1. Syntaxe : <<PrefixeFonctionnel>>[_]NomDescriptif

Voir la [légende de la syntaxe](#) pour une description de la syntaxe.

- PrefixeFonctionnel – requis, défini par le développeur permet de représenter une séparation logique entre des occurrences de tables liées entre elles. Cela peut être une abréviation ou un nom fonctionnel.
- NomDescriptif – permet de fournir des informations sur le rôle de l'occurrence de table.
- " _ " – requis

Il est recommandé d'utiliser un tiret bas double « _ » comme séparateur.

Ainsi, il reste possible d'utiliser des caractères de tiret bas au sein des noms de tables sources et des noms d'occurrences de tables tout en offrant une meilleure lisibilité et des possibilités d'analyse.

4.3.2 Groupement fonctionnel d'occurrences de tables ou FTOG (Functional Table Occurrence Groups)

Le groupement fonctionnel d'occurrences de tables est une autre approche, selon laquelle le graphique des liens contient plusieurs « mini » graphes fonctionnels constitués uniquement des occurrences de tables représentant la fonctionnalité de ce groupement. L'une des différences clés de cette méthode est qu'elle subdivise les différents composants ou les différentes fonctions d'une solution en sous-ensembles plus restreints. Cela permet parfois d'obtenir un graphique plus facile à gérer. Cela réduit aussi le nombre d'occurrences de tables à rechercher au sein du graphique des liens. Pour une fonction donnée, vous n'avez qu'à consulter le groupement fonctionnel d'occurrences de tables concerné. Chaque groupement fonctionnel d'occurrences de tables ne contient que des occurrences de tables liées à la fonctionnalité correspondante.

Quelle que soit la variante utilisée, l'un des objectifs du groupement fonctionnel d'occurrences de tables est d'offrir un nom reflétant la fonction du groupe ou du sous-ensemble d'occurrences de tables. Cette fonction peut varier d'un groupe à l'autre, mais l'important est que le groupe ou sous-ensemble d'occurrences de tables remplisse une fonction et que son nom reflète cette fonction, afin qu'il soit facile à identifier hors du graphique des liens. La Figure 2 montre un graphique des liens organisé à l'aide de la méthode du groupement fonctionnel d'occurrences de tables. Chacun de ces groupements fonctionnels d'occurrences de tables remplit une certaine fonction indépendante des autres groupes. A l'aide de l'outil Texte de FileMaker 8, il est possible d'ajouter des notes à chaque groupe fonctionnel afin de mieux préciser son rôle.

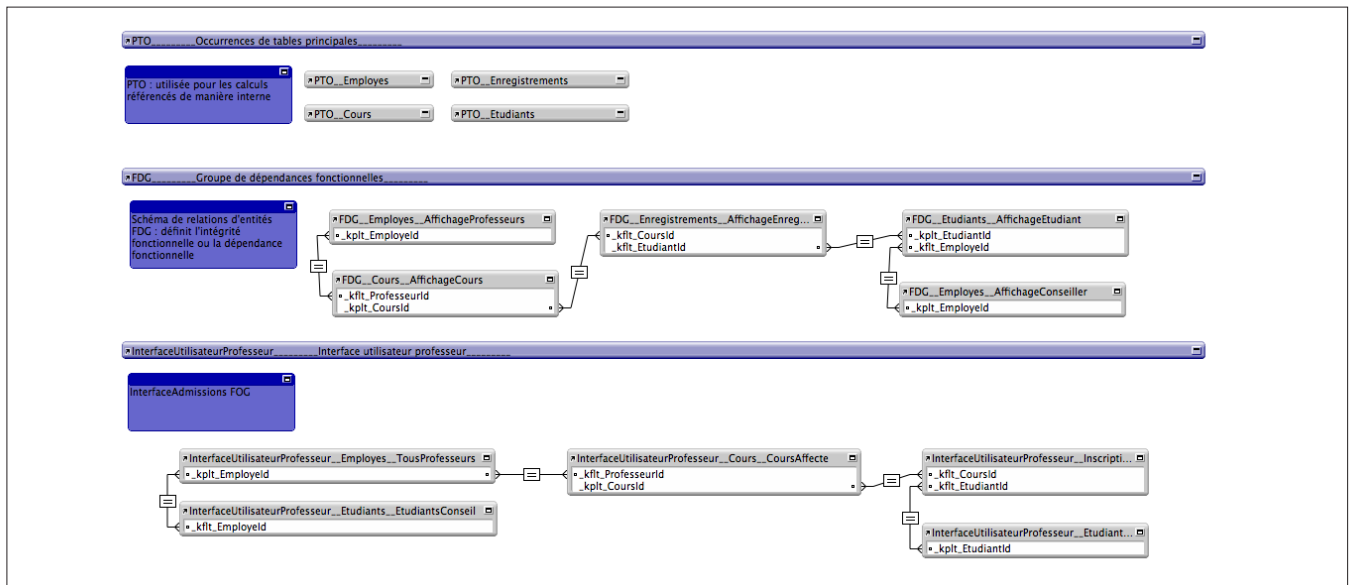


Figure 5



4.3.2.1 AVANTAGES DU GROUPEMENT FONCTIONNEL D'OCCURRENCES DE TABLES (FTOG)

- Réduit la complexité du graphique des liens
- Réduit le nombre d'occurrences de tables
- Offre une organisation visuelle sous forme de liste par fonctionnalité
- Fonctionne bien avec les solutions de taille moyenne ou petite (ce qui est le cas de la plupart des solutions)
- Peut également fonctionner avec les solutions de grande taille, suivant l'étendue des besoins fonctionnels.
- Plus les besoins fonctionnels sont nombreux, moins cette méthode organisationnelle est intéressante.
- Les tables liées au sein d'un groupement fonctionnel d'occurrences de tables sont relativement limitées, et elles sont groupées par ordre alphabétique.
- Prend en charge le modèle relationnel bidirectionnel en permettant aux modèles d'utiliser n'importe quel groupe d'occurrences de tables au sein du groupement fonctionnel d'occurrences de tables. Cela permet de réduire le nombre d'occurrences de tables en ne limitant pas la visibilité du modèle à une occurrence de table du groupement fonctionnel d'occurrences de tables.
- Prend en charge un large éventail de méthodologies de développement, utilisées pour les solutions reposant sur des tables externes, qui utilisent beaucoup de scripts, ainsi que pour les solutions plus ouvertes reposant sur ces contrôles natifs FileMaker.
- Offre un groupement fonctionnel au sein des menus d'occurrences de tables extérieurs au graphique des liens, suivant un ordre relativement bien groupé.

4.3.2.2 INCONVÉNIENTS DU GROUPEMENT FONCTIONNEL D'OCCURRENCES DE TABLES (FTOG)

- Les règles de dénomination sont considérées comme plus strictes.
- Le système de dénomination n'essaie pas d'exposer la structure ou la signification hors du graphique des liens.
- Comme un modèle peut reposer sur n'importe quelle occurrence de table, les différentes occurrences de tables sont plus difficiles à repérer sur le graphique des liens.

4.3.2.3 GROUPEMENT FONCTIONNEL D'OCCURRENCES DE TABLES : RECOMMANDATIONS INTÉGRANT LES STANDARDS

1. Syntaxe : <<FTOG>>[__]<NomTableSource>[__]NomDescriptif

Voir la [légende de la syntaxe](#) pour une description de la syntaxe.

- FTOG – requis ; défini par le développeur

Permet de nommer un ensemble d'occurrences de tables. Le nom peut être une abréviation ou un nom fonctionnel. Par exemple, il est possible d'utiliser le préfixe DF devant le nom du groupe d'occurrences de tables représentant la dépendance fonctionnelle. De la même manière, le préfixe « InterfaceUtilisateur » peut servir à indiquer le groupe d'occurrences de tables représentant l'interface utilisateur.

- « __ » – requis

Il est recommandé d'utiliser un tiret bas double « __ » comme séparateur. Ainsi, il reste possible d'utiliser des caractères de tiret bas au sein des noms de tables sources et des noms d'occurrences de tables tout en offrant une meilleure lisibilité et des possibilités d'analyse.

- NomTableSource – requis ; utilise le nom de la table source

Inclut le nom de la table source dans le nom de l'occurrence de table. Ce système offre un indicateur visuel pour la programmation en dehors du graphique des liens pour la table sous-jacente.

En effet, la fonction Obtenir(NomTableModèle) renvoie le nom de l'occurrence de table, et non de la table source, et il n'y a pas d'autre moyen d'obtenir le nom de la table source. Pour obtenir le nom de la table source par la programmation, il faut l'inclure dans le nom de l'occurrence de table.



- « __ » – requis
Il est recommandé d'utiliser un tiret bas double « __ » comme séparateur. Ainsi, il reste possible d'utiliser des caractères de tiret bas au sein des noms de tables sources et des noms d'occurrences de tables tout en offrant une meilleure lisibilité et des possibilités d'analyse.

- NomDescriptif
Permet de fournir des informations sur le rôle de l'occurrence de table.

Exemples :

- ▶ FD__Client__TousClients
- ▶ Interface__Facture__FacturesRetard
- ▶ Interface__Facture__Courantes_Factures
- ▶ Synchronisation__Version__Versions_Stockees_Serveur

2. Séparation du groupement fonctionnel d'occurrences de tables : il peut être difficile de localiser une occurrence de table au sein de la boîte de dialogue Calcul (pour placer une rubrique sur un modèle ou l'affecter à un modèle, par exemple). Il faut en effet la sélectionner dans une liste déroulante dépourvue de système de classement. Par exemple, imaginez que votre solution comporte 200 occurrences de tables. Pour effectuer l'une des tâches mentionnées ci-dessus, il faut naviguer au sein de cette longue liste. Avec une convention de groupement d'occurrences de tables, la tâche devient plus simple puisque les groupes sont classés par ordre alphabétique. Dans l'exemple suivant, nous n'avons que 13 occurrences de tables. Toutefois, la liste pourrait être beaucoup plus longue.

FDG__Cours__AffichageCours
FDG__Employes__AffichageConseiller
FDG__Employes__AffichageProfesseurs
FDG__Enregistrements__AffichageEnregistrements
FDG__Etudiants__AffichageEtudiant
InterfaceUtilisateurProfesseur__Cours__CoursAffecte
InterfaceUtilisateurProfesseur__Employes__TousProfesseurs
InterfaceUtilisateurProfesseur__Etudiants__EtudiantsConseil
InterfaceUtilisateurProfesseur__Etudiants__EtudiantsEnCours
InterfaceUtilisateurProfesseur__Inscriptions__EnregistrementsClasse
PTO__Cours
PTO__Employes
PTO__Enregistrements
PTO__Etudiants

Figure 6

Dans l'exemple suivant, chaque groupement fonctionnel d'occurrences de tables possède un en-tête permettant de le distinguer aisément. Quand la liste de groupes d'occurrences de tables est longue, cela peut être très utile.



```

FDG_____Groupe de dépendances fonctionnelles_____
FDG__Cours__AffichageCours
FDG__Employes__AffichageConseiller
FDG__Employes__AffichageProfesseurs
FDG__Enregistrements__AffichageEnregistrements
FDG__Etudiants__AffichageEtudiant
InterfaceUtilisateurProfesseur_____Interface utilisateur professeur_____
InterfaceUtilisateurProfesseur__Cours__CoursAffecte
✓ InterfaceUtilisateurProfesseur__Employes__TousProfesseurs
InterfaceUtilisateurProfesseur__Etudiants__EtudiantsEnCours
InterfaceUtilisateurProfesseur__Inscriptions__EnregistrementsClasse
PTO_____Occurrences de tables principales_____
PTO__Cours
PTO__Employes
PTO__Enregistrements
PTO__Etudiants

```

Figure 7

Cette méthode nécessite de créer une table d'espacement ou d'utiliser des occurrences de tables existantes comme en-têtes pour chaque groupe fonctionnel. Dans notre exemple, le nom de la table est constitué de 10 traits d'union (-----). Le nom n'est pas important, ni la création d'une table spécifique.

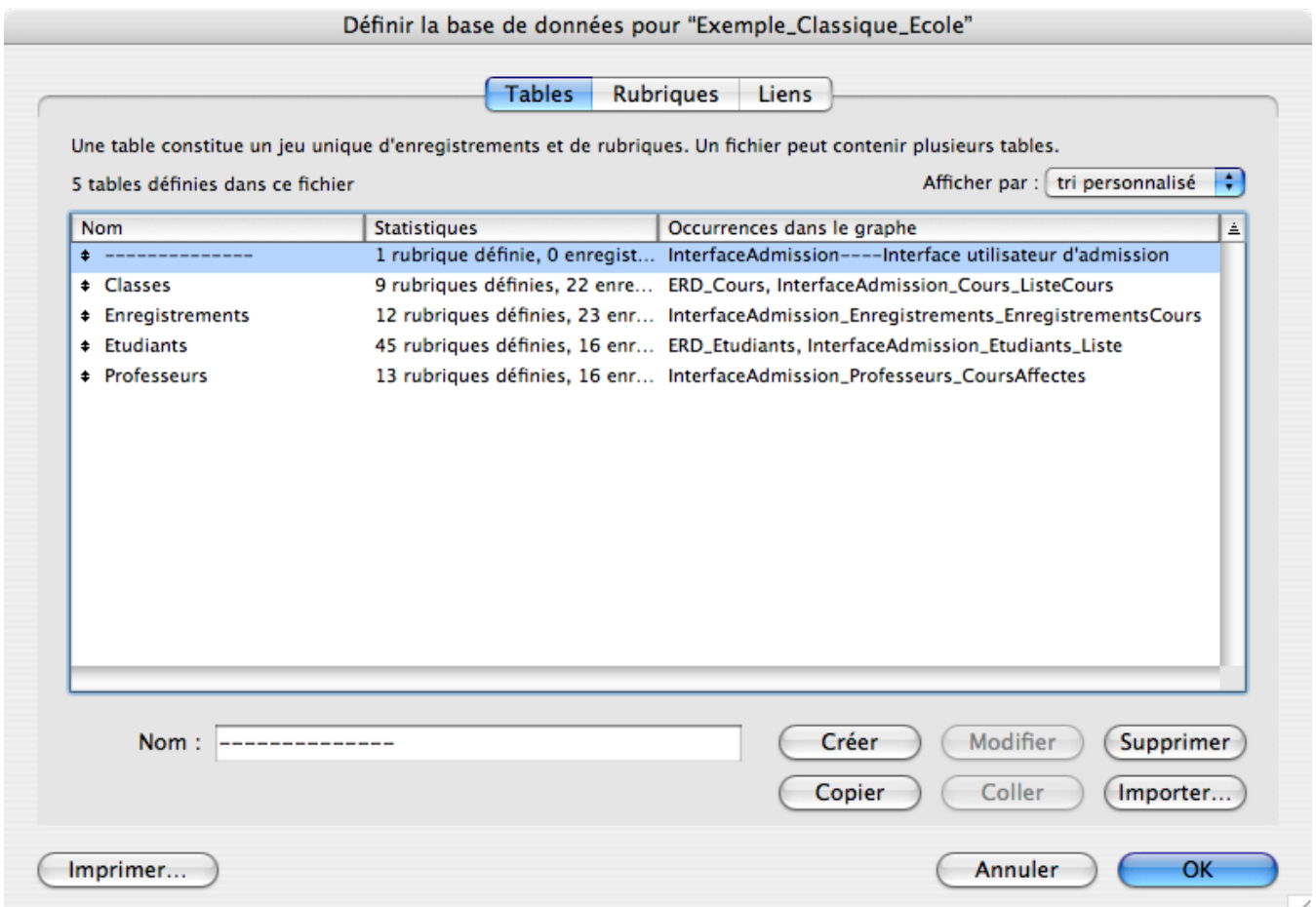


Figure 8



Lors de la création de la table d'espacement, l'occurrence de table créée est insérée automatiquement dans le graphique des liens. Ajoutez des occurrences de tables en utilisant cette table comme table source, puis indiquez le nom qui servira à séparer chacun des groupes fonctionnels dans la liste déroulante. L'exemple de Figure 7 montre trois structures de ce type.

- FDG _____ Dépendance fonctionnelle _____
- PTO _____ Occurrences de tables primaires _____
- InterfaceUtilisateurProfesseur _____ Interface utilisateur des professeurs _____

Dans une liste d'occurrences de tables très longue, cette technique vous aidera à trouver l'occurrence de table que vous recherchez en insérant des en-têtes pour chaque groupement fonctionnel d'occurrences de tables.

4.3.3 Groupement hiérarchique d'occurrences de tables HTOG (Hierarchical Table Occurrence Grouping) ou modèle de l'ancre et de la bouée (Anchor Buoy)

Cette approche du graphique de liens prend une série d'occurrences de tables reliées les unes aux autres par des lignes de liens et y ajoute une structure et des règles.

Elle est décrite par Roger Jacques, de Soliant Consulting, dans le numéro de septembre 2005 de *FileMaker Advisor* dans un article intitulé *Managing the FileMaker Pro 7 Relationship Graph*. Il n'est pas possible d'en reprendre tous les détails dans ce document. Toutefois, nous utiliserons sa description comme base pour un certain nombre d'exemples. Roger Jacques décrit le modèle hiérarchique de l'ancre et de la bouée de la manière suivante :

« La nature hiérarchique de la méthode de l'ancre et de la bouée repose sur une structure bien définie. Chaque groupement d'occurrences de tables comporte une occurrence de table ancre qui est servie par des données provenant d'une ou de plusieurs tables bouées. Ce modèle hiérarchique n'admet qu'un chemin d'accès entre deux occurrences de tables données. Une fois qu'un chemin d'accès a été créé, il est possible d'ajouter autant d'occurrences de tables que nécessaire pour récupérer les données liées. Voici une règle que je respecte à la lettre : ne reliez jamais deux groupes d'occurrences de table par une ligne de lien, et FileMaker ne vous laissera pas créer de références circulaires. »

« L'occurrence de table ancre est toujours située complètement à gauche, et les occurrences de tables bouées partent à droite sous forme de ramifications. Vous pouvez ajouter des occurrences de tables au fur et à mesure que vous en avez besoin afin d'offrir à la table ancre un accès aux données liées. En pratique, les chemins d'accès dépassent rarement une profondeur de quatre ou cinq niveaux. »

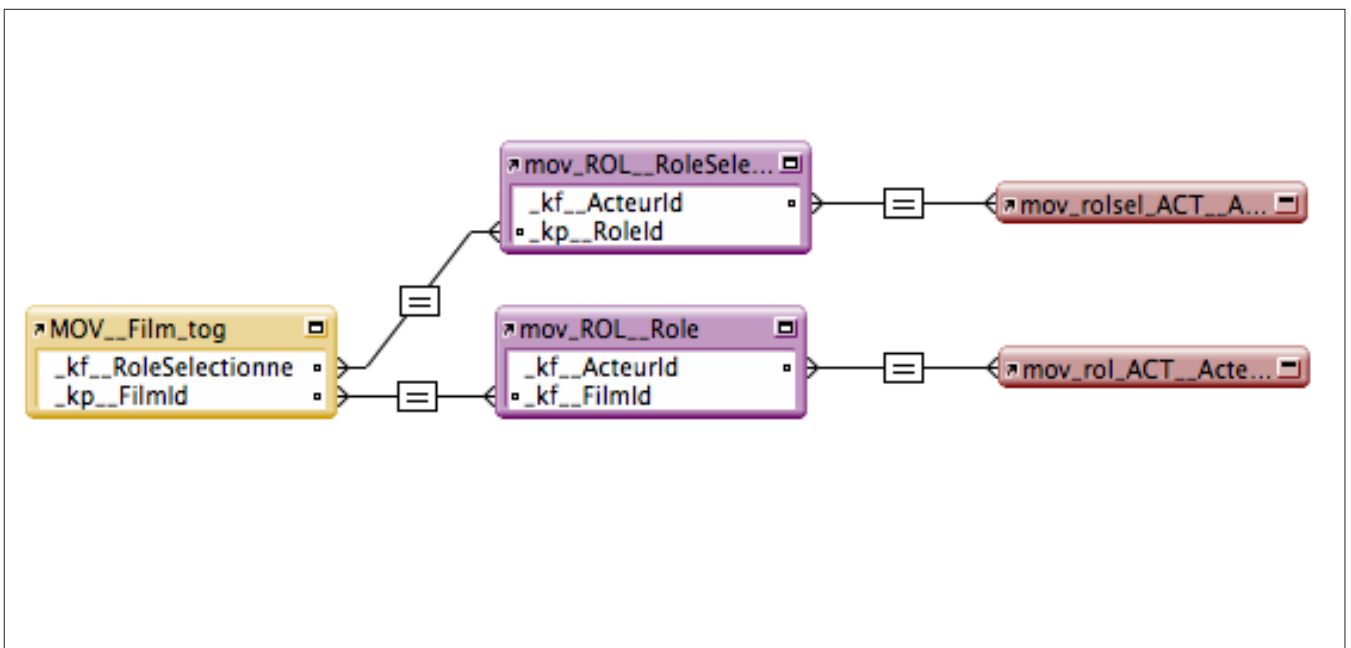


Figure 8a



« Notez que les deux chemins d'accès utilisent les mêmes tables mais à des fins différentes. Dans cet exemple, le chemin d'accès inférieur repose sur les liens des clés primaire et étrangères pour les trois entités, et il dresse la liste de tous les rôles et acteurs du film en cours. Le chemin d'accès supérieur permet aux utilisateurs de sélectionner une rangée de table externe, puis de voir les détails du rôle et de l'acteur correspondants dans les rubriques liées. »

Avec cette méthode, les modèles ne peuvent reposer que sur l'occurrence de table ancre (située complètement à gauche). Les occurrences de tables « bouées » servent uniquement à apporter des données aux modèles éventuels ou au système logique à partir de l'ancre.

4.3.3.1 AVANTAGES DU GROUPEMENT HIÉRARCHIQUE D'OCCURRENCES DE TABLES (HTOG)

- Réduit considérablement la complexité du graphique
- Permet de mieux comprendre la structure relationnelle du système en dehors du graphique.
- Le système de dénomination est automatique et il ne nécessite pas de réfléchir beaucoup, hormis pour les métadonnées facultatives de la relation (nom descriptif).
- Les groupements d'occurrences de tables reposent principalement sur des modèles, ce qui permet de localiser facilement les occurrences de tables au sein du graphique des liens.
- Fonctionne bien avec les solutions de grande taille
- Offre un groupement fonctionnel au sein des menus d'occurrences de tables extérieurs au graphique des liens, suivant un ordre hiérarchique.
- Prend en charge la règle du chemin unique du graphique des liens et n'est pas affecté par cette règle.
- Les tables liées au sein d'un groupement fonctionnel d'occurrences de tables sont relativement limitées, et elles sont groupées par ordre alphabétique.
- Prend en charge un large éventail de méthodologies de développement : utilisées pour les solutions reposant sur des tables externes, qui font intervenir beaucoup de scripts, ainsi que pour les solutions plus ouvertes reposant sur ces contrôles natifs FileMaker.
- Accessible à un large éventail de développeurs, qui doivent cependant être relativement chevronnés.

4.3.3.2 INCONVÉNIENTS DU GROUPEMENT HIÉRARCHIQUE D'OCCURRENCES DE TABLES (HTOG)

- Augmente considérablement le nombre d'occurrences de tables.
- Par essence, ne prend pas en charge le modèle relationnel bidirectionnel. Compte tenu du fait que les modèles sont réservés à l'ancre, il faut créer un autre groupement ancre-bouée pour exprimer les données de droite à gauche pour le modèle.

4.3.3.3 GROUPEMENT HIÉRARCHIQUE D'OCCURRENCES DE TABLES (HTOG) : RECOMMANDATIONS INTÉGRANT LES STANDARDS

1. Les différents mots doivent être séparés de manière cohérente à l'aide de la méthode indiquée pour chaque partie de la syntaxe.



Méthodes recommandées pour la séparation des mots (occurrences de tables : modèle de l'ancre et de la bouée)

Ancres		
NomEnteteTOGAncre (AnchorTOGHeaderName)	MAJUSCULES	
NomTableSource casse	ChameauMinuscule	*Doit correspondre à la casse du nom de la table
Bouées		
NomEnteteTOGAncre (AnchorTOGHeaderName)	casseChameauMinuscule	
NomTableSourceAbreviation (SourceTableNameAbbreviation)	MAJUSCULES	Abréviation conseillée afin de limiter la longueur
NomTableSource	casseChameauMinuscule	*Doit correspondre à la casse du nom de la table
NomDescriptif	casseChameauMinuscule ou CasseChameauMajuscule	

2. Syntaxe de l'ancre

<<NomEnteteTOGAncre>>[_]<NomTableSource>

Voir la [légende de la syntaxe](#) pour une description de la syntaxe.

- NomEnteteTOGAncre – requis, défini par le développeur
Identifie le groupe d'occurrences de tables. Chaque occurrence de table « connectée » aura le même nom d'en-tête de groupement d'occurrences de tables ancre.
- « _ » – requis
Un tiret bas double sépare le nom d'en-tête de groupement d'occurrences de tables ancre du nom de la table source.
- NomTableSource –
indique la table associée à l'occurrence de table.

3. Syntaxe de la bouée

<<NomEnteteTOGAncre>>[_]<<NomTableSourceAbreviation>>[_]<NomTableSource>[_]NomDescriptif

Voir la [légende de la syntaxe](#) pour une description de la syntaxe.

- NomEnteteTOGAncre – requis ; utilise la valeur définie pour l'ancre
Chaque occurrence de table « connectée » aura le même nom d'en-tête de groupement d'occurrences de tables ancre.
- « _ » – requis
Un tiret bas simple sépare le nom d'en-tête de groupement d'occurrences de tables ancre du premier nom abrégé de table source.
- NomTableSourceAbreviation – requis ; défini par le développeur
Chaque nom abrégé de table source est séparé par un tiret bas « _ » unique.
- « __ » – requis
Un tiret bas double sépare le nom abrégé de table source du nom de la table source.
- NomTableSource – requis ; valeur prévue pour le nom de la table source
Indique la table associée à l'occurrence de table.
- « __ » – requis
Un tiret bas double sépare le nom de la table source du nom descriptif.



- NomDescriptif
dans les occurrences de tables bouées, le nom descriptif est facultatif mais il peut contenir une description naturelle de l'occurrence de table ou, éventuellement, des métadonnées indiquant sur quoi repose le lien. Par exemple, le nom descriptif « IdSociete » peut indiquer que le lien repose sur la rubrique « IdSociete ».

Exemples :

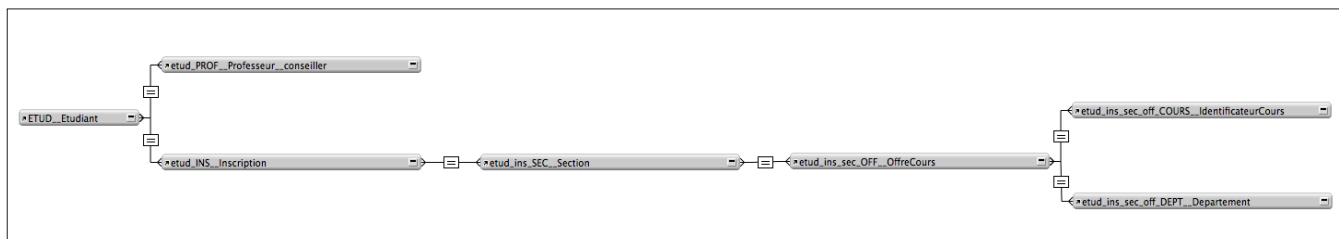


Figure 9

- Ancre : ETUD_Etudiant – Il s'agit d'un groupement d'occurrences de tables ancre puisqu'il apparaît toujours complètement à gauche et ne fait apparaître que la table source. Le nom nous indique également qu'il repose sur la table source « Etudiant ».
- etud_ins__SEC__Section – cette occurrence de table bouée fait partie du groupement ancre « etud ». C'est la troisième occurrence de table connectée à « ins » qui est connectée à l'ancre « etud ». Elle repose sur la table source « Section ». Il n'y a pas de nom descriptif.
- etud_ins_sec_off__COURS__IdentificationCours – cette occurrence de table bouée fait partie du groupement ancre « etud ». Elle est située à 5 niveaux de profondeur et a pour table source « Cours ». Elle est connectée à « off » (offreCours), « sec » (Section), « ins » (Inscription), puis à l'ancre « etud » (Etudiant). Elle utilise également un nom descriptif facultatif : IdentificationCours.

4.4 Considérations annexes :

Occurrence de table primaire (PTO) : quelle que soit la méthodologie utilisée pour le graphique des liens, quand vous créez des calculs, vous devez sélectionner le contexte à partir duquel le calcul sera évalué. Pour les rubriques dont le résultat est dérivé sans référencer des données liées, ces calculs doivent toujours reposer sur le même contexte. A cet effet, il peut être utile d'employer une occurrence de table spéciale appelée « occurrence de table primaire ». L'occurrence de table primaire est l'occurrence de table désignée pour être utilisée lors de la création de calculs référencés en interne, c'est-à-dire les calculs dérivés de données contenues exclusivement dans la même table, et non issues de tables liées. La Figure 10 montre comment utiliser l'occurrence de table primaire pour la table source Employés.



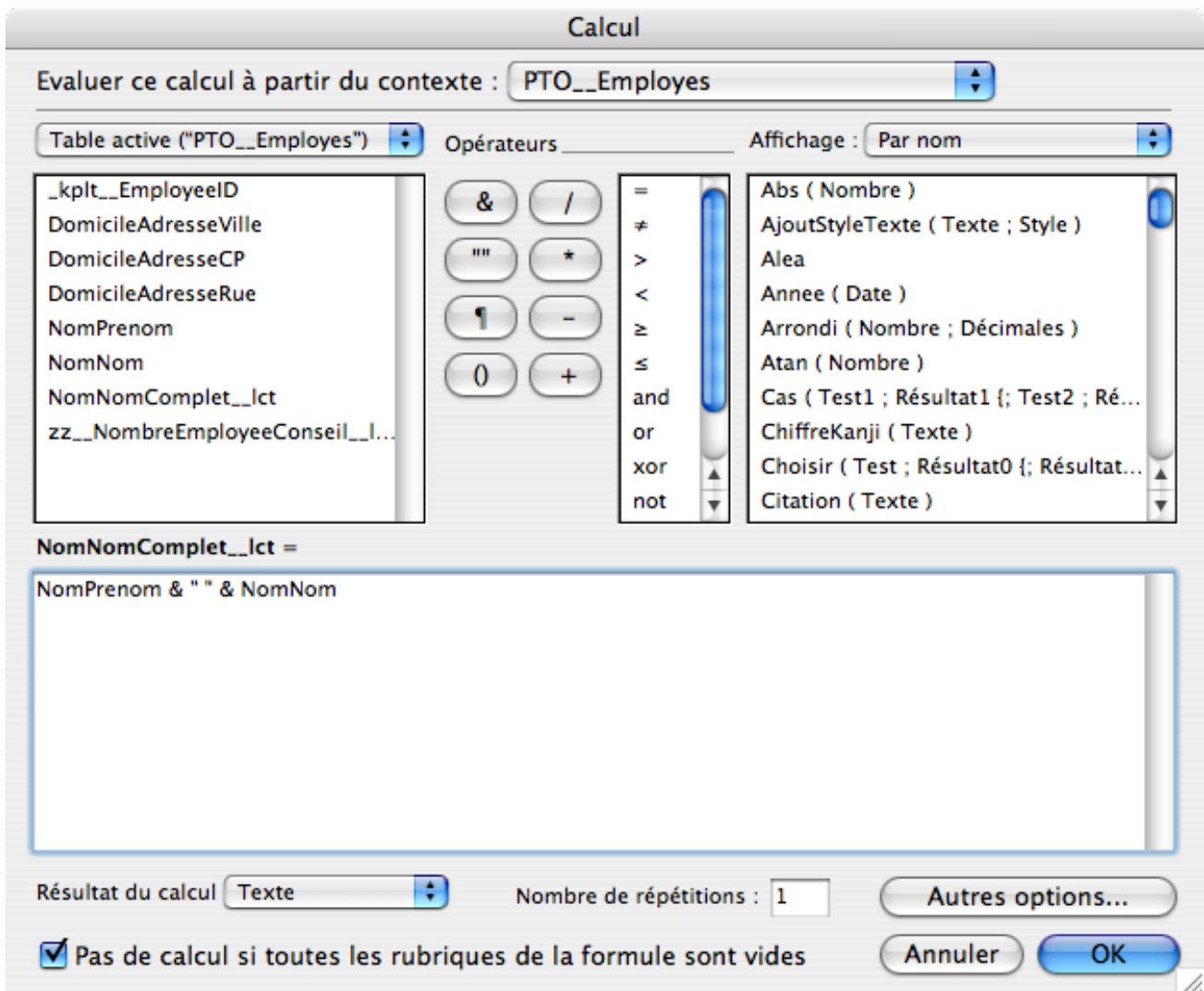


Figure 10

En utilisant une occurrence de table primaire pour chaque calcul interne, vous saurez toujours que l'occurrence de table primaire fournit le contexte dans lequel évaluer le calcul. Au fur et à mesure que la solution évoluera, vous n'aurez pas à vous souvenir de la série d'occurrences de tables utilisée pour évaluer ces calculs internes.

Si vous supprimez d'autres occurrences de tables, cela n'aura pas d'impact sur les calculs qui utilisent l'occurrence de table primaire comme contexte d'évaluation.

Syntaxe :

[PTO][__]<NomTableSource>

Recommandé : [PTO__]<<NomTableSource>>

Voir la [légende de la syntaxe](#) pour une description de la syntaxe.

- (PTO) DesignationOccurrenceTablePrimaire – requis ; défini par le développeur.
Peut être défini par le développeur. Ce document recommande d'utiliser « PTO » (abréviation de « Primary Table Occurrence », « occurrence de table primaire »).
- « __ » – requis
Deux caractères de tiret bas séparent le préfixe PTO du nom de la table source.
- NomTableSource
Nom de la table associée à l'occurrence de table.



5 ▷ Modèles

Avant de vous demander comment nommer les modèles, vous déterminer qui utilisera les noms. Dans les solutions dans lesquelles l'utilisateur final sélectionne directement le modèle à l'aide du menu déroulant situé dans la zone d'état, vous devez de préférence choisir un nom descriptif et dépourvu de métadonnées supplémentaires. A l'inverse, dans les solutions dans lesquelles l'utilisateur final n'a pas accès au menu déroulant, le nom peut être utile au développeur.

En outre, il est important de choisir un jeu de caractères qui ne nuise pas à la connectivité externe. Par exemple, les requêtes XML utilisent le nom du modèle pour établir le contexte. Les noms de modèles doivent être uniques. Une fois encore, toutes les solutions n'imposent pas ces contraintes. Toutefois, en préparant d'ores et déjà l'évolution de votre solution, vous faciliterez le développement par la suite. Il est également important de penser à une convention de casse par souci de cohérence et de professionnalisme. Le nom d'un modèle peut comporter jusqu'à 100 caractères. Toutefois, certaines boîtes de dialogue ne sont pas capables d'afficher une telle longueur. Les développeurs sont souvent confrontés au problème de ne pas pouvoir identifier l'occurrence de table liée à un modèle donné pendant le développement. A moins de quitter le contexte du travail en cours, d'entrer en mode Modèle et d'ouvrir la boîte de dialogue Format de modèle, ces informations ne sont pas disponibles. En outre, beaucoup de solutions nécessitent de pouvoir identifier et désigner une table source par la programmation pour une interface ou pour la logique du programme. Comme il est impossible d'obtenir ces informations par l'intermédiaire de la programmation, il peut être intéressant de coder ou d'inclure des métadonnées sur l'occurrence de table et sur le nom de la table source dans le nom du modèle.

La notation rend le nom plus convivial pour l'utilisateur final. Malheureusement, il n'existe aucun moyen de fournir un nom à la fois convivial pour le développeur et pour l'utilisateur. Par conséquent, suivant les exigences et la structure de votre solution, vous pourrez avoir à faire des compromis. Enfin, en l'absence de nom de modèle structuré ou codé, il n'y a aucun moyen de classer les modèles en catégories. Par exemple, les modèles sont utilisés pour les interfaces et les rapports, mais aussi pour des tâches « cachées », pour la soumission de données et pour d'autres choses spécifiques. Or, il n'y a aucun moyen de classer ces modèles en catégories. Pourtant, cela peut être utile pour comprendre le rôle des modèles. Aussi est-il intéressant d'inclure dans le nom du modèle un système permettant de distinguer des catégories.

5.1 Objectifs :

- ✓ Jeu de caractères recommandé
- ✓ Convention recommandée pour la casse
- ✓ Problèmes de longueur de nom de modèle
- ✓ Mécanisme pour identifier l'occurrence de table associée à un modèle sans entrer dans la boîte de dialogue Format de modèle
- ✓ Mécanisme permettant de voir par la programmation les métadonnées relatives au nom de table source des modèles pendant l'exécution
- ✓ Mécanisme pour obtenir par la programmation des métadonnées sur d'autres modèles que le modèle en cours d'utilisation
- ✓ Impossibilité de proposer un nom à la fois plus convivial pour l'utilisateur et plus parlant pour le développeur
- ✓ Eviter d'utiliser le même nom pour une occurrence de table et pour un modèle.
- ✓ Méthode pour aider le développeur à comprendre le rôle du modèle
- ✓ Mécanisme pour contourner l'impossibilité d'organiser les modèles par fonction, catégorie, etc.
- ✓ Problèmes liés aux technologies de connectivité (XML, ODBC et JDBC) et aux mots SQL réservés



5.2 Définition du problème :

Jeu de caractères recommandé : le jeu de caractères utilisé pour les modèles doit prendre en considération les limitations imposées par FileMaker en matière de caractères. Il doit aussi tenir compte des limitations inhérentes aux autres technologies qui utilisent les noms de modèles pour communiquer avec FileMaker, notamment les requêtes XML.

Convention recommandée pour la casse : cette recommandation n'a pas pour but de résoudre un problème particulier mais d'assurer un mode de dénomination uniforme des modèles. L'objectif est de sélectionner une méthode et de s'y tenir.

Longueur recommandée pour les noms de modèles : le nom d'un modèle FileMaker peut contenir jusqu'à 100 caractères. Toutefois, souvent, une telle longueur n'est guère pratique. A partir de la version 8, la plupart des boîtes de dialogue peuvent afficher sans problème un nom de 100 caractères. Toutefois, la boîte de dialogue Format de modèle n'est pas redimensionnable et elle limite le nombre de caractères affichables à 42 sous OS X et à 48 sous Windows.

Mécanisme pour identifier l'occurrence de table associée à un modèle sans entrer dans la boîte de dialogue Format de modèle : à moins d'avoir une très bonne mémoire, il n'y a aucun moyen de savoir à quelle occurrence de table un modèle est associé. La seule solution consiste à entrer en mode Modèle et à ouvrir la boîte de dialogue Format de modèle. Cette absence d'informations pendant le développement peut être ennuyeuse et obliger le développeur à abandonner le travail en cours pour retrouver les informations fondamentales relatives aux modèles. Aussi est-il intéressant d'inclure une notation reflétant l'occurrence de table associée dans le nom du modèle.

Mécanisme pour accéder par la programmation aux métadonnées sur le nom de table source du modèle lors de l'exécution : le développeur peut avoir besoin d'accéder par la programmation à certaines informations sur le modèle, comme le nom de l'occurrence de table ou le nom de la table, pour effectuer telle ou telle action. La fonction `Obtenir (NomTableModèle)` renvoie le nom de l'occurrence de table mais pas le nom de la table elle-même (le nom de la table source associée à l'occurrence de table). Aussi est-il intéressant d'inclure le nom de la table source dans le nom du modèle.

Mécanisme pour obtenir par la programmation des métadonnées sur les autres modèles que le modèle en cours d'utilisation : ce point est intimement lié au précédent. Le développeur peut également avoir besoin d'accéder par la programmation aux informations sur l'ensemble des modèles d'un fichier. La fonction de conception `NomsModeles (NomFichier)` renvoie tous les modèles d'un fichier. En revanche, aucune fonction ne permet d'obtenir les occurrences de tables associées. La seule solution consiste à utiliser la fonction `Obtenir (NomTableModèle)` pour chaque, et il n'y a aucun moyen de connaître le nom de la table source. Aussi est-il intéressant d'inclure le nom de la table source et celui de l'occurrence de table dans le nom du modèle.

Impossibilité de proposer un nom à la fois plus convivial pour l'utilisateur et plus parlant pour le développeur : jusqu'à la version 8 de FileMaker (inclusive), il n'est pas possible d'avoir à la fois un nom convivial pour l'utilisateur et un nom parlant pour le développeur. En tant que développeur, vous devez choisir la méthode la mieux adaptée à votre solution.

Occurrence de table et nom de modèle portant le même nom : par défaut, FileMaker attribue à un modèle le nom de l'occurrence de table sur laquelle il repose. Alors que cela permet de gagner du temps dans certains cas, cela peut poser des problèmes quand il faut accéder aux informations par la programmation.

Le développeur doit comprendre le rôle d'un modèle. Quand il conçoit une solution, le développeur a souvent intérêt à intégrer dans le nom d'un modèle des informations sur le rôle de ce modèle. Par exemple, il peut faire commencer le nom de tous les modèles de rapports par « rap » (abréviation de « rapport »).

De la même manière, il peut utiliser le préfixe « travail » pour indiquer que le modèle sert à effectuer un travail donné par la programmation dans un script. Quel que soit le système de classement, il faut disposer d'un moyen d'indiquer le « rôle » du modèle du point de vue de l'organisation et de la programmation.

Impossibilité d'organiser les modèles par fonction, catégorie et autre : que vous ayez à sélectionner un modèle à partir de l'onglet Modèle ou par l'intermédiaire d'une boîte de dialogue, il peut être fastidieux de parcourir une longue liste qui n'est classée que par ordre alphabétique. En effet, la liste des modèles ne comporte pas de système de classement. Aussi est-il intéressant de disposer d'une structure de dénomination qui subdivise cette liste en catégories ou fonctions afin que les noms de modèles soient plus faciles à retrouver.



5.3 Recommandations intégrant les standards (modèles) :

1. Le nom d'un modèle ne doit utiliser que les caractères
 - alphabétiques majuscules et minuscules aA - zZ ;
 - numériques : 0, 1, 2, 3, 4, 5, 6, 7, 8 et 9 ;
 - de tiret bas simple et double « _ » « __ ».
2. Il ne doit pas PAS contenir d'espaces.
3. Il ne doit PAS commencer par une valeur numérique.
4. Les noms de modèles doivent être systématiquement au singulier ou au pluriel.
5. Les différents mots composant le nom d'un modèle doivent être séparés de manière cohérente, à l'aide de l'une des méthodes suivantes.

Méthodes recommandées pour la séparation des mots (noms de modèles)

casseChameauMinuscule	monModele
CasseChameauMajuscule	MonModele
Tiret bas simple (minuscules)	mon_modele
Tiret bas simple (majuscule à chaque initiale)	Mon_Modele
Tiret bas simple (MAJUSCULES)	MON_MODELE

6. Syntaxe : <<PrefixeFonctionnel>>[_]NomDescriptif[_]<NomOccurrenceTable>
Voir la [légende de la syntaxe](#) pour une description de la syntaxe.
 - PrefixeFonctionnel – facultatif ; défini par le développeur
Fournit une syntaxe générale, compréhensible de tous, tout en permettant l'emploi de préfixes définis par le développeur en fonction de ses besoins et préférences.
Si vous préférez ne pas employer de préfixes fonctionnels, omettez-les simplement et commencez par le nom descriptif. Il n'y a pas besoin d'inclure le premier « _ ». La recommandation générale est d'utiliser un indicateur de 1 à 5 caractères et de sélectionner une convention de casse pour la fonction.
 - « __ » – requis
Il est recommandé d'utiliser un tiret bas double « __ » comme séparateur. Ainsi, il reste possible d'utiliser des tiret bas au sein du nom descriptif et du nom d'occurrence de table tout en offrant une meilleure lisibilité et des possibilités d'analyse.
 - NomDescriptif
Offre un moyen d'attribuer au modèle un nom un peu plus convivial.
 - « _ » – requis
Il est recommandé d'utiliser un tiret bas double « _ » comme séparateur. Ainsi, il reste possible d'utiliser des caractères de tiret bas au sein du nom descriptif et du nom d'occurrence de table tout en offrant une meilleure lisibilité et des possibilités d'analyse.
 - NomOccurrenceTable – requis ; utilise le nom d'occurrence de table défini
Offre un indicateur visuel pour la programmation en dehors du graphique des liens pour l'occurrence de table sous-jacente.



6 ▷ Fonctions personnalisées

Les fonctions personnalisées confèrent à l'environnement de développement un niveau de flexibilité inégalé. Le développeur peut ainsi aller bien au-delà des fonctions par défaut intégrées à FileMaker.

FileMaker emploie une syntaxe et une convention de syntaxe spécifiques pour l'ensemble des fonctions intégrées. Par exemple en français, la fonction de texte `MotsExtraits` s'utilise sous la forme `MotsExtraits (Texte ; MotDébut ; NombreMots)`. Cette fonction utilise la casse `CasseChameauMajuscule` et chaque paramètre utilise la casse `CasseChameauMajuscule`. Pour des raisons de cohérence, il est préférable d'utiliser la casse et la syntaxe déjà en vigueur dans le produit.

Une fonction peut être autonome ou être combinée à des fonctions personnalisées. Il n'y a aucun moyen de classer ou regrouper les fonctions liées. Certains développeurs affirment qu'une fonction personnalisée admet des éléments en entrée et en génère d'autres en sortie et qu'elle pas besoin d'être regroupée. D'autres, au contraire, estiment qu'il est important de savoir qu'une fonction repose sur d'autres fonctions ou les utilise. Dans ce document, afin de différencier les fonctions, nous parlerons de fonctions « privées » et de fonctions « personnalisées publiques ».

Lors de l'examen d'un calcul, il n'est pas toujours facile de voir qu'il utilise une fonction personnalisée. Or, il est important d'être en mesure d'identifier une fonction personnalisée dans un calcul. Le moteur de calculs ne fait pas de différence entre les fonctions intégrées et personnalisées. Aussi peut-il être difficile de comprendre un calcul ou de résoudre un problème. Il est en revanche possible d'inclure dans le nom d'une fonction personnalisée un indicateur permettant de la reconnaître facilement dans le calcul.

Par ailleurs, l'un des principaux intérêts des fonctions personnalisées est d'offrir une fonctionnalité étendue qui n'existe qu'au sein de la fonction elle-même. Toute personne qui utilise cette fonction doit comprendre à quoi elle sert et comment elle fonctionne. Par conséquent, il est important que vous documentiez librement votre document pour référence ultérieure (pour vous-même mais aussi pour les autres développeurs susceptibles d'utiliser votre travail). Le présent document contient un certain nombre de consignes de base sur les éléments qu'une fonction personnalisée doit documenter et sur la manière dont elle doit les documenter. Vous pouvez en outre vous reporter à la section [Calculs](#) de ce document pour obtenir des informations sur la mise en forme des calculs de vos fonctions personnalisées.

6.1 Objectifs :

- ✓ Jeu de caractères recommandé
- ✓ Convention recommandée pour la casse
- ✓ Séparation syntaxique
- ✓ Consignes pour la longueur
- ✓ Méthode pour grouper des fonctions liées
- ✓ Méthode pour différencier les fonctions personnalisées au sein des calculs
- ✓ Méthode pour distinguer les fonctions parentes/enfants
- ✓ Instructions pour le paramétrage des options de sécurité des fonctions privées et personnalisées publiques
- ✓ Méthode pour documenter une fonction personnalisée en incluant les éléments minimaux et le format

6.2 Définition du problème :

Jeu de caractères recommandé : les fonctions personnalisées permettent d'aller au-delà des fonctions par défaut de FileMaker Pro/Advanced. Pour des raisons de cohérence, il est recommandé de suivre le jeu de caractères utilisé dans la liste de fonctions de FileMaker Pro et de FileMaker Pro Advanced.

Convention de casse recommandée : la méthode choisie pour séparer les mots au sein du nom d'une fonction personnalisée doit être la même que pour les fonctions intégrées. Toutes les fonctions intégrées suivent la même convention de casse : `CasseChameauMajuscule` pour le nom et `CasseChameauMajuscule` pour tous les paramètres s'il y a lieu. En outre, il faut prévoir une syntaxe dans la fonction personnalisée pour séparer les préfixes éventuellement requis pour faciliter l'organisation.



Syntaxe : la syntaxe de la fonction personnalisée doit être cohérente par rapport à celle des fonctions intégrées mais doit aussi permettre de différencier les préfixes du nom et de grouper les fonctions associées par nom.

Problèmes liés à la longueur des noms de fonctions : le nom d'une fonction peut comporter jusqu'à 100 caractères. Toutefois, il faut savoir que les noms longs sont peu pratiques dans la boîte de dialogue Calcul. Cette boîte de dialogue affiche 26 caractères sous Mac OS X et 30 sous Windows dans FileMaker Pro et FileMaker Pro 8 Advanced.

Groupement des fonctions liées : il est fréquent qu'une fonction soit liée à une ou plusieurs autres fonctions pour remplir son rôle. La convention doit définir une méthode qui permette d'identifier visuellement les fonctions liées et qui mette en évidence les dépendances au sein du groupe pour le développeur.

Différenciation des fonctions au sein des calculs : il n'est pas toujours aisé, en interrogeant un calcul, de savoir qu'un élément est une fonction intégrée ou personnalisée. La convention doit définir une méthode qui permette de différencier clairement les deux.

Sécurité pour les fonctions personnalisées : la convention doit sensibiliser le développeur aux problèmes que peut entraîner la mise à la disposition de fonctions publiques ou privées.

Documentation des fonctions personnalisées : la convention doit fournir un modèle de documentation pour les différents éléments d'une fonction personnalisée. Ce modèle doit couvrir les éléments eux-mêmes, leur mise en forme et leur emplacement.

6.3 Recommandations intégrant les standards (fonctions personnalisées) :

6.3.1 Fonctions personnalisées publiques

Une fonction personnalisée publique est une fonction personnalisée conçue pour être appelée directement. Il peut s'agir de la fonction principale au sein d'un groupe de fonctions personnalisées (elle peut d'ailleurs elle-même appeler d'autres fonctions personnalisées), ou d'une fonction personnalisée autonome, sans aucune dépendance. Dans tous les cas, ce terme désigne une fonction prévue pour être appelée directement. Par exemple, imaginons que nous ayons trois fonctions personnalisées :

- `UrlSurbrillance`
- `UrlDernierCar`
- `UrlRenvoyerTout`
- `UrlRenvoyerUn`

En utilisant le groupement par nom, il est facile de s'apercevoir que ces fonctions personnalisées sont liées. Elles jouent toutes un rôle lié aux URL. Ce qui n'est pas clair, c'est leur structure. Sans connaître ces fonctions au préalable, il n'est pas possible de savoir si certaines d'entre elles dépendent d'autres fonctions. En classant chaque fonction dans la catégorie Privée ou Publique, nous pouvons savoir quelle fonction doit être appelée en premier.

- `UrlSurbrillance__CFpub`
- `UrlSurbrillance__CFpvt`
- `UrlRenvoyerTout__CFpvt`
- `UrlRenvoyerUn__CFpvt`

Cette convention nous indique qu'au sein du groupe URL, la fonction `UrlSurbrillance__CFpub` est conçue pour être appelée en premier et chacune des autres fonctions lui est subordonnée.

Syntaxe :

`NomFonctionPersonnalisee[___][CFpub]`

Voir la [légende de la syntaxe](#) pour une description de la syntaxe.



- Utilisation du suffixe «_CFpub» (CF en majuscules, pub en minuscules) – indique, dans un calcul, que la fonction est personnalisée et de type public.
- Le nom d’une fonction personnalisée ne doit utiliser que les caractères :
 - alphabétiques majuscules et minuscules aA - zZ ;
 - numériques : 0, 1, 2, 3, 4, 5, 6, 7, 8 et 9 ;
 - de tiret bas simple et double «_» «__».
- Il ne doit PAS contenir d’espaces.
- Il ne doit PAS commencer par une valeur numérique.
- Il ne doit pas PAS contenir de points.
- Il doit utiliser la casse CasseChateauMajuscule pour le nom de la fonction.
- Exemple : `GenerateurTableau_CFpub`

6.3.2 Fonctions personnalisées privées

Le terme « privé » n’a pas pour but de signaler un accès limité mais de communiquer l’intention du concepteur de la fonction :

Une fonction privée est destinée à être appelée par une autre fonction personnalisée, qu’il s’agisse d’une fonction personnalisée publique ou d’une fonction personnalisée privée. En classant une fonction personnalisée en tant que fonction publique, vous indiquez qu’elle est destinée à être appelée directement et qu’elle peut avoir des dépendances avec d’autres fonctions personnalisées. A l’inverse, une fonction personnalisée privée est toujours destinée à être appelée par une autre fonction personnalisée et jamais directement.

Que se passe-t-il si une fonction privée est appelée directement ?

Rien ! Les termes « public » et « privé » servent uniquement à illustrer une relation de dépendance, et non à restreindre les possibilités d’une fonction. En pratique, si vous avez besoin d’appeler une fonction privée directement, il est probablement préférable d’en faire une fonction publique.

Syntaxe :

`NomFonctionPersonnalisee[___][CFpvt]`

Voir la [légende de la syntaxe](#) pour une description de la syntaxe.

- Utilisation du suffixe «_CFpvt» (CF en majuscules, pvt en minuscules) – indique, dans un calcul, que la fonction est personnalisée et de type privé.
- Le nom d’une fonction personnalisée ne doit utiliser que les caractères :
 - alphabétiques majuscules et minuscules aA - zZ ;
 - numériques : 0, 1, 2, 3, 4, 5, 6, 7, 8 et 9 ;
 - de tiret bas simple et double «_» «__».
- Il ne doit pas PAS contenir d’espaces.
- Il ne doit PAS commencer par une valeur numérique.
- Il ne doit pas PAS contenir de points.
- Il doit utiliser la casse CasseChateauMajuscule pour le nom de la fonction.
- Exemple : `GenerateurTableauRecherche_CFpvt`

6.3.3 Paramètres de fonctions personnalisées

Dans les fonctions privées ou publiques, les paramètres doivent suivre le format standard de toutes les fonctions FileMaker.

Syntaxe :

`NomFonctionPersonnalisee[___] [CFpub] ou [CFpvt] (ParametreUn, ParametreDeux)`

Voir la [légende de la syntaxe](#) pour une description de la syntaxe.



- Le nom d'une fonction personnalisée ne doit utiliser que les caractères :
 - alphabétiques majuscules et minuscules aA - zZ ;
 - numériques : 0, 1, 2, 3, 4, 5, 6, 7, 8 et 9 ;
 - de tiret bas simple et double « _ » « __ ».
- Il ne doit pas PAS contenir d'espaces.
- Il ne doit PAS commencer par une valeur numérique.
- Il ne doit pas PAS contenir de points.
- Il doit utiliser la casse CasseChameauMajuscule.
- Exemple : MaFonctionPersonnalisee__CFpub (ParametreUn, ParametreDeux)

6.3.4 Exemples de dénomination de fonctions personnalisées

La Figure 11 illustre un ensemble de fonctions personnalisées utilisant la convention de dénomination intégrant les standards.

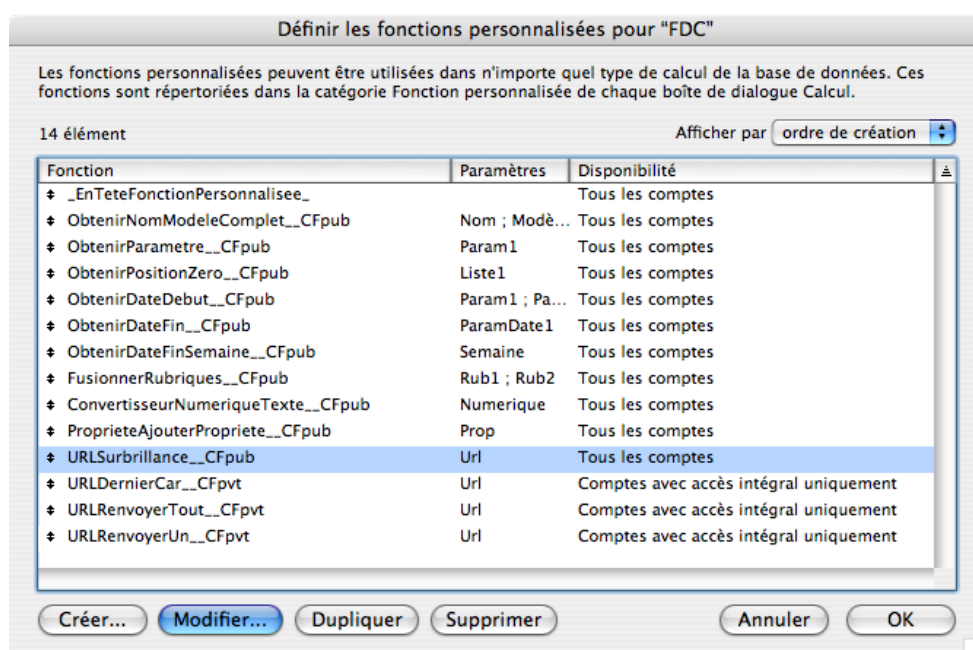


Figure 11

6.3.5 Documentation des fonctions personnalisées

Chaque fonction personnalisée doit contenir un minimum de documentation. Elle doit être représentée ligne par ligne, dans l'ordre suivant :

1. Nom – nom de la fonction
2. Historique – (peut inclure un ou plusieurs des éléments suivants, ou d'autres encore si nécessaire)
 - Nom du créateur – nom du développeur de la fonction
 - Adresse électronique du créateur – adresse électronique du développeur de la fonction
 - Nom de l'auteur de la dernière modification – nom du dernier développeur qui a modifié la fonction
 - Adresse électronique de l'auteur de la dernière modification – adresse électronique du dernier développeur qui a modifié la fonction
 - Date de création – date au format JJ-MMM-AAAA de création de la fonction
 - Date de la dernière modification – date au format JJ-MMM-AAAA de la dernière modification de la fonction



3. Objet – description de ce que la fonction est censée faire
4. Paramètres – liste des paramètres éventuellement utilisés pour la fonction
5. Notes importantes – informations supplémentaires
6. Exemple

/*

Nom :

MaFonctionPersonnalisee_CFpub

Historique :

Créée par Jean SERIEN

Date de création : 01-JAN-2005

Date de modification : 01-JAN-2005

Objet :

Il s'agit d'une fonction d'exemple montrant le format de consignation.

Paramètres :

parametreUn : explication d'exemple de paramètre. Peut inclure tous les détails pertinents.

parametreDeux : explication d'exemple de paramètre. Peut inclure tous les détails pertinents.

Notes importantes :

Informations supplémentaires pouvant être utiles aux futurs développeurs et pour une référence ultérieure.

*/

6.3.6 Mise en forme des fonctions personnalisées

Reportez-vous à la section [Calculs](#) de ce document pour obtenir des informations sur la mise en forme des calculs de vos fonctions personnalisées.

6.4 Considérations annexes

Il est également possible de paramétrer une fonction personnalisée sur Tous les comptes ou sur Uniquement les comptes dotés de tous les privilèges si elle doit être utilisée dans le moteur de calculs. Pour protéger le caractère privé des fonctions, vous pouvez envisager de paramétrer vos fonctions personnalisées publiques sur Tous les comptes et vos fonctions personnalisées privées sur Uniquement les comptes dotés de tous les privilèges. Ainsi, vos fonctions personnalisées publiques seront disponibles pour le moteur de calcul et les fonctions privées seront invisibles.



7 ▷ Scripts

La gestion des scripts intervient à deux endroits de l'organisation et de la documentation. Premièrement, l'organisation des scripts ne doit pas entrer en ligne de compte dans le processus de dénomination. ScriptMaker n'inclut pas d'outils organisationnels, et de nombreux développeurs utilisent leur propre système de dénomination pour faciliter les choses avec les scripts. Alors qu'il est possible de déplacer manuellement des scripts vers le haut et vers le bas dans une longue liste, il n'y a pas réellement de moyen de classer un ensemble de scripts en catégories. Par conséquent, la position et le nom sont les seules options restantes. Cette version du document ne tentera pas de proposer une nomenclature répondant aux besoins d'organisation. Toutefois, un certain nombre d'aspects de la documentation et de règles de dénomination peuvent aider le développeur à éviter des écueils.

7.1 Objectifs :

- ✓ Jeu de caractères recommandé
- ✓ Jeu de caractères recommandé pour les variables
- ✓ Convention recommandée pour la casse (variables)
- ✓ Problèmes de longueur de nom de script
- ✓ Mécanisme pour voir qu'un script nécessite des paramètres à partir de son nom
- ✓ Impossibilité de proposer un nom à la fois plus convivial pour l'utilisateur et plus parlant pour le développeur
- ✓ Impossibilité d'arrêter un script à la fin, alors que le débogueur nécessite une action de script supplémentaire
- ✓ Modèle cohérent pour la documentation d'un script
- ✓ Problèmes liés aux technologies de connectivité (appel de scripts à partir de XML)

7.2 Définition du problème :

Jeu de caractères recommandé : les caractères autorisés par FileMaker pour les noms de scripts sont eux aussi soumis à un certain nombre de restrictions de base. En outre, quand un script est appelé en externe (par l'intermédiaire de code XML, par exemple), les caractères utilisés peuvent poser des problèmes. En tant que développeur, vous devez avoir conscience des contraintes de FileMaker et de tous les systèmes externes avec lesquels vos solutions doivent interagir.

Convention recommandée pour la casse : cette recommandation n'a pas pour but de résoudre un problème particulier mais d'assurer un mode de dénomination uniforme des scripts. L'objectif est de sélectionner une méthode et de s'y tenir. Modèle cohérent pour la documentation des

Scripts : la convention doit fournir un modèle de documentation pour les différents éléments d'un script. Ce modèle doit couvrir les éléments eux-mêmes, leur mise en forme et leur emplacement.

7.3 Recommandations intégrant les standards (scripts) :

7.3.1 Noms de scripts

1. Le nom d'un script ne doit utiliser que les caractères
 - alphabétiques majuscules et minuscules aA - zZ ;
 - numériques 0, 1, 2, 3, 4, 5, 6, 7, 8 et 9 ;
 - de tiret bas simple et double « _ » « __ ».
2. Le nom d'un script ne doit pas contenir plus de 100 caractères.



7.3.2 Variables

1. Le nom d'une variable doit commencer par \$ pour une variable locale ou par \$\$ pour une variable globale.
2. Il ne doit pas PAS contenir d'espaces.
3. Les différents mots composant le nom d'une variable doivent être séparés de manière cohérente, à l'aide de l'une des méthodes suivantes :

Méthodes recommandées pour la séparation des mots (variables) \$, \$\$

casseChameauMinuscule	\$maVariable
CasseChameauMajuscule	\$MaVariable
Tiret bas simple (minuscules)	\$ma_variable
Tiret bas simple (majuscule à chaque initiale)	\$Ma_Variable
Tiret bas simple (MAJUSCULES)	\$MA_VARIABLE

7.3.3 Documentation des scripts

L'action de script Commentaire est très pratique pour insérer de la documentation. Le commentaire doit être la première étape du script. Il doit inclure chacun des composants de la liste ci-dessous et être mis en forme comme dans cet exemple. Si un commentaire dépasse la limite de 30 K caractères, il est possible d'ajouter une nouvelle action de script.

1. Nom – nom du script
2. Historique – (peut inclure un ou plusieurs des éléments suivants, ou d'autres encore si nécessaire)
 - Nom du créateur – nom du développeur du script
 - Adresse électronique du créateur – adresse électronique du développeur du script
 - Nom de l'auteur de la dernière modification – nom du dernier développeur qui a modifié le script
 - Adresse électronique de l'auteur de la dernière modification – adresse électronique du dernier développeur qui a modifié le script
 - Date de création – date au format JJ-MMM-AAAA de création du script
 - Date de la dernière modification – date au format JJ-MMM-AAAA de la dernière modification du script
3. Rôle – description de ce que le script est censé faire
4. Variables déclarées – liste des variables que le script déclare
5. Variables référencées – liste des variables que le script référence, comme les variables globales déclarées précédemment
6. Paramètres – Liste des paramètres requis pour le script
7. Notes importantes – Informations supplémentaires

La Figure 12 montre un exemple.



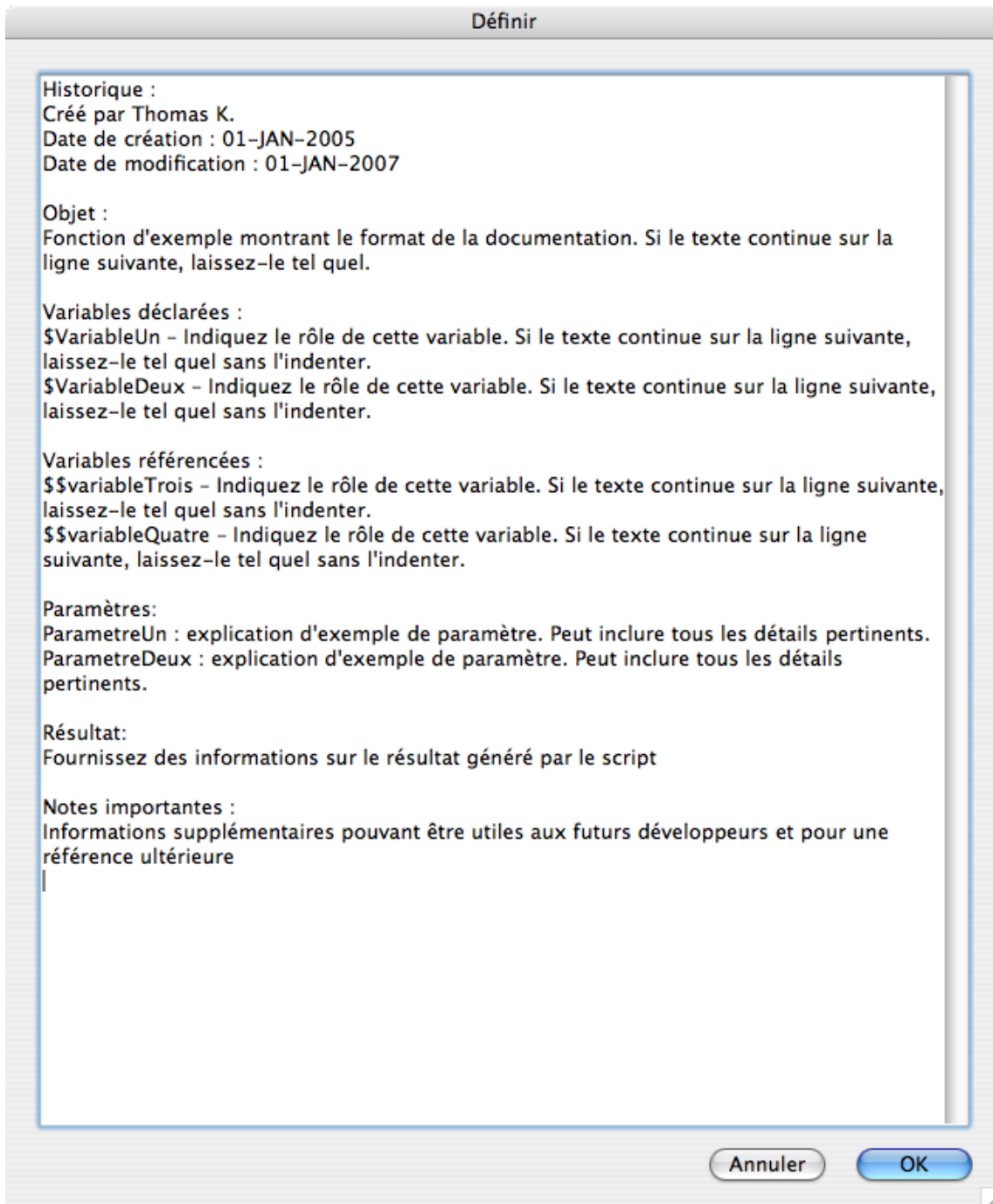


Figure 12



8 ▷ Calculs

Le format et les commentaires d'un calcul peuvent le rendre plus clair. C'est au développeur qu'il incombe de prévoir un format vraiment informatif. Un bon commentaire doit plus expliquer le « pourquoi » que le « comment ». Il doit en outre permettre de comprendre le code si celui-ci ne parle pas de lui-même. Dans certains cas, les techniques de codage peuvent faciliter la compréhension. Par exemple, l'utilisation quasi systématique de la fonction Définir() dans les calculs même les plus simples peut améliorer considérablement la compréhension du code et le rendre plus lisible.

La convention recommandée vise à proposer une approche cohérente de la mise en forme et des commentaires utilisés pour les calculs. Les avantages de la mise en forme et de la documentation ne sont pas forcément évidents pour les calculs simples. Toutefois, au fur et à mesure que les calculs deviennent complexes, il est important de les organiser et de les documenter pour référence ultérieure (pour le développeur d'origine ou pour son successeur).

8.1 Objectifs

- ✓ Modèle cohérent pour les variables utilisées dans les calculs.
- ✓ Modèle cohérent pour la documentation des calculs.
- ✓ Catégories de commentaires recommandées. Exemples d'espacement et de mise en forme pour pallier l'impossibilité d'utiliser des couleurs pour coder les différents composants, tels que : les fonctions, les chaînes, les rubriques, les occurrences de tables, les opérateurs et les fonctions personnalisées.

8.2 Définition du problème

Documentation des calculs : la plupart des développeurs commentent plus ou moins leur travail. Certains ont une approche minimaliste tandis que d'autres sont plus bavards. Toutefois, dans l'ensemble, tous les développeurs s'accordent à penser que la documentation est utile, surtout en cas de calculs complexes. La recommandation intégrant les standards propose une quantité minimale d'informations ainsi qu'un emplacement et un format. Il est évidemment possible d'aller au-delà, mais ce minimum doit être respecté.

Mise en forme des calculs : quand les calculs sont mis en forme correctement, ils sont plus lisibles et leurs composants sont plus faciles à identifier. En outre, ils sont plus faciles à comprendre pour vous et pour vos collègues. Etant donné que le moteur de calcul ne met pas en forme automatiquement les calculs, c'est au développeur de s'en charger. Le présent document contient des instructions générales mais il ne recommande pas un format particulier.

8.3 Recommandations intégrant les standards (calculs) :

8.3.1 Variables

1. Le nom d'une variable doit commencer par \$ pour une variable locale ou par \$\$ pour une variable globale.
2. Il ne doit pas PAS contenir d'espaces.
3. Les différents mots composant le nom d'une variable doivent être séparés de manière cohérente, à l'aide de l'une des méthodes suivantes :

Méthodes recommandées pour la séparation des mots (variables) \$, \$\$

casChameauMinuscule	\$maVariable
CasChameauMajuscule	\$MaVariable
Tiret bas simple (minuscules)	\$ma_variable
Tiret bas simple (majuscule à chaque initiale)	\$Ma_variable
Tiret bas simple (MAJUSCULES)	\$MA_VARIABLE



8.3.2 En-tête de commentaires dans les calculs

Chaque calcul commenté doit contenir un en-tête contenant les sections suivantes :

- Objet – description générale de ce que le calcul est censé faire.
Dépendances – informations sur les éléments éventuels dont le calcul dépend.
- Il peut s’agir de rubriques, de fonctions personnalisées ou de variables globales, par exemple.
- Historique – espace réservé permettant d’insérer des informations pertinentes sur le calcul, telles que le nom du créateur, la date de création et la version.

Exemple :

```
/*
```

Objet :

Ce calcul est un exemple montrant comment il faut mettre en forme une rubrique de type calcul et la commenter. Le commentaire Objet apparaîtra dans la liste des rubriques.

Dépendances :

RubriqueTexte : rubrique au format texte utilisée par ce calcul

RubriqueNombre : rubrique de type numérique utilisée comme compteur

MaFonctionPersonnalisee__CFpub : fonction personnalisée utilisée dans ce calcul

Historique :

Créée par Thomas K

Date : 01/01/2005

Modifié le : 02/07/2006 pour la version x de la solution

```
*/
```

8.3.3 Commentaires entre les lignes de calculs

Un calcul doit être commenté dans l’en-tête des commentaires, et non au fil du calcul. Il y a toutefois une exception : lors de la création de variables avec la fonction `Définir()`. A moins que le nom de la variable parle de lui-même, le développeur doit commenter cette variable. S’il n’y a pas suffisamment d’espace pour commenter la variable après sa création, indentez la ligne suivante et placez le commentaire sur cette ligne.

Utilisez l’indentation de manière libérale pour améliorer la lisibilité. De nombreuses fonctions se prêtent très bien à l’indentation, comme le montre l’exemple suivant avec les fonctions `Définir()` et `Cas()`. Il est ainsi possible de séparer d’emblée les éléments, tels que le corps du calcul (l’instruction `Cas`) de la création de la variable. (Pour plus d’informations sur ce sujet, reportez-vous à la section [Mise en forme des calculs.](#))

```
Définir(  
[  
  $Texte = RubriqueTexte1;  
  // Mettre la valeur d’une rubrique dans une variable afin qu’elle soit appelée une fois  
  $Var1 = expression1;  
  // Valeur de la première expression  
  $Var2 = FonctionPersonnalisee1_Cfpub($Texte)  
  // Ligne de commentaire optionnelle pour le cas où le calcul  
  // est trop long pour placer le commentaire juste derrière.  
];  
Cas(  
  test1; resultat1;  
  test2; resultat2;  
  resultatParDefaut)  
)
```



Certaines rubriques de type calcul sont extrêmement simples et leur rôle est évident. Dans ces calculs, le commentaire d'objet est intéressant pour documenter la base de données mais il n'est pas nécessaire d'ajouter d'autres commentaires.

8.3.4 Mise en forme des calculs

Quand les calculs sont mis en forme correctement, ils sont plus lisibles et leurs composants sont plus faciles à identifier. En outre, ils sont plus faciles à comprendre pour vous et pour vos collègues. Etant donné que le moteur de calcul ne met pas en forme automatiquement les calculs, c'est au développeur de s'en charger. Le présent document contient des instructions générales mais il ne recommande pas un format particulier.

8.3.4.1 EXEMPLE DE MISE EN FORME N° 1

Nous pouvons voir que le calcul suivant effectue une certaine forme de remplacement. Toutefois, comme la chaîne de texte est très longue, il est difficile de savoir si la fonction `Remplace2` est celle qui commence et termine ce calcul. Avec une mise en forme correcte, il est facile de voir que la fonction `Remplace2` commence et termine ce calcul, en ajoutant un retour chariot à la fin.

```
Remplace2(DATA::zz_ligne_actuelle; Debut(DATA::zz_ligne_actuelle;
Position(DATA::zz_ligne_actuelle; "/" ; 1 ; 1) - 1); Extrait(DATA::zz_ligne_actuelle;
Position(DATA::zz_ligne_actuelle; "/" ; 1;
1) + 1; Longueur(DATA::zz_ligne_actuelle) - 1)) & "¶"
```

En regardant rapidement ce calcul ainsi mis en forme, vous pouvez voir que la fonction `Remplace2` admet trois paramètres :

```
Remplace2 ( Texte ; ChaîneRecherche ; ChaîneRemplacement )
```

```
Remplace2 (
  DATA::zz_ligne_actuelle ;
  Debut (
    DATA::zz_ligne_actuelle ;
    Position ( DATA::zz_ligne_actuelle ; "/" ; 1 ; 1 ) - 1
  ) ;
  Extrait (
    DATA::zz_ligne_actuelle ;
    Position (DATA::zz_ligne_actuelle ; "/" ; 1 ; 1 ) + 1 ;
    Longueur (DATA::zz_ligne_actuelle ) - 1
  )
) & "¶"
```

Une fois les autres fonctions décomposées correctement, ces paramètres sont très faciles à identifier. Voir la Figure 13 ci-après.



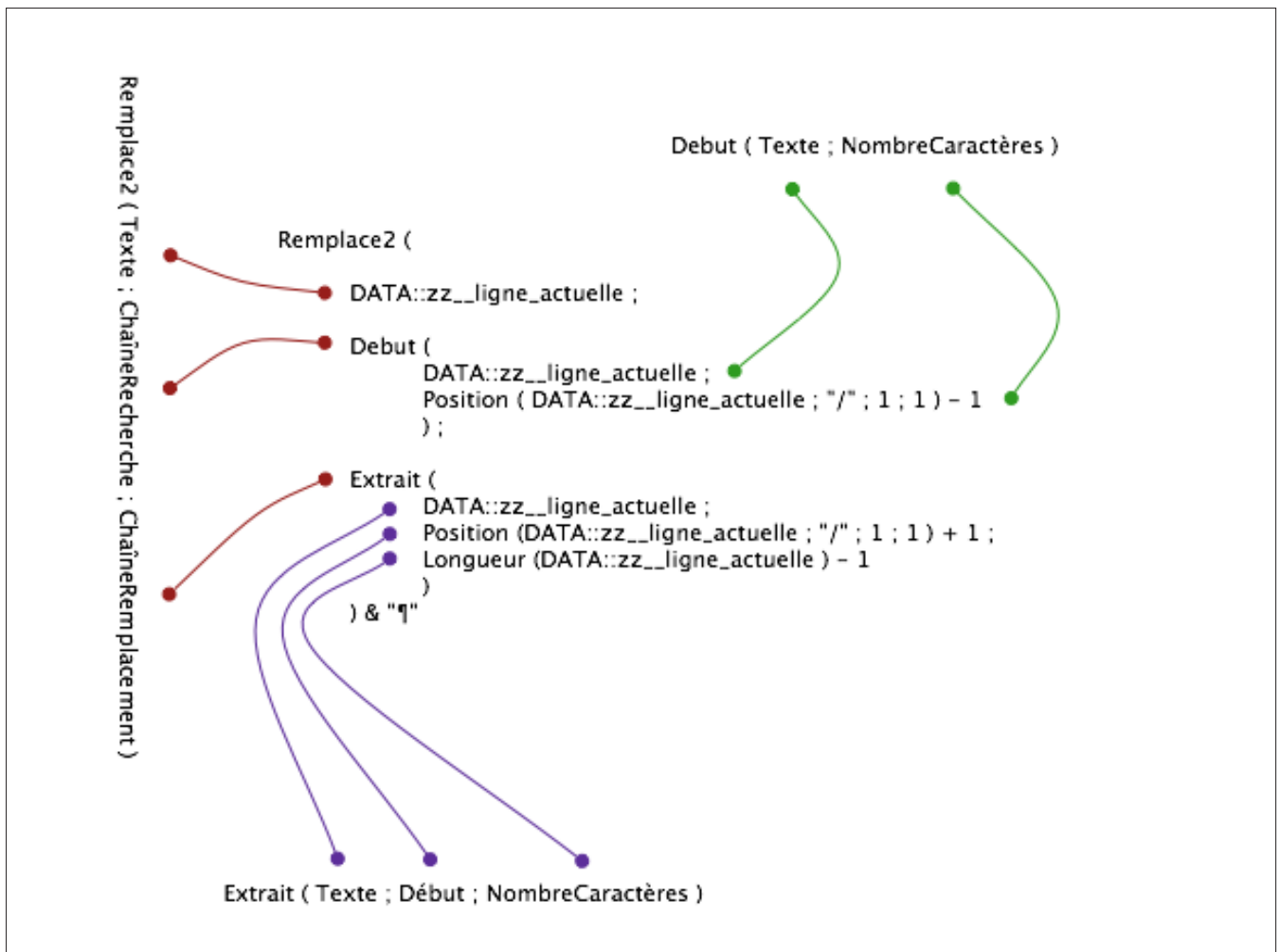


Figure 13

En décomposant chaque paramètre, il est beaucoup plus facile de comprendre la structure des fonctions.

8.3.4.2 EXEMPLE DE MISE EN FORME N° 2

L'idée générale est d'ajouter des espaces et des sauts de ligne aux calculs afin d'identifier clairement le début et la fin des fonctions et des arguments de fonctions. Si un appel de fonction est trop long pour tenir sur une ligne, il est décomposé en plusieurs parties de telle sorte que :

- Le nom de la fonction et la parenthèse initiale soient suivis d'une nouvelle ligne.
- Chaque argument de la fonction commence sur une nouvelle ligne et soit indenté un niveau en dessous du nom la fonction.
- La parenthèse finale commence une nouvelle ligne, indentée au même niveau que le nom de la fonction.



```

Si(
  Position( selection_menu; "Autre fruit"; 0; 1 );
  Cas(
    NomJour( Obtenir( DateActuelle ) ) = "Mardi";
    "Oranges";
    NomJour( Obtenir( DateActuelle ) ) = "Jeudi";
    "Pommes";
    /* sinon */
    "Bananes"
  );
  "Piores"
) & " and " &
Si(
  Position( selection_menu; "Autre legume"; 0; 1 );
  Cas(
    NomJour( Obtenir( DateActuelle ) ) = "Jeudi";
    "Pois";
    /* sinon */
    "Carottes"
  );
  "Maïs"
)

```

Il faut tenir compte du fait que la boîte de dialogue Calcul de FileMaker Pro n'utilise pas une police à chasse fixe. Par conséquent, il est nécessaire de commencer le premier argument de la fonction sur une nouvelle ligne au lieu de placer l'appel de fonction et son premier argument sur la même ligne. Par exemple, le code suivant, qui est plus compact que le précédent, est moins pratique quand il est affiché avec une police à chasse variable :

```

Si(Position( selection_menu; "Autre fruit"; 0; 1 );
  Cas(NomJour( Obtenir( DateActuelle ) ) = "Mardi";
    "Oranges";
    NomJour( Obtenir( DateActuelle ) ) = "Jeudi";
    "Pommes";
    /* sinon */
    "Bananes"
  );
  "Piores"
) & " and " &
Si(Position( selection_menu; "Autre legume"; 0; 1 );
  Cas(NomJour( Obtenir( DateActuelle ) ) = "Jeudi";
    "Pois";
    /* sinon */
    "Carottes"
  );
  "Maïs"
)

```

Utilisez un format qui permette de détecter plus facilement les paires de parenthèses. Ainsi, le système de formatage fait toujours commencer la parenthèse fermante sur une nouvelle ligne quand une fonction est subdivisée en plusieurs lignes plutôt que de renvoyer quelque chose du type :



```

Si(
  Position( selection_menu; "Autre fruit"; 0; 1 );
  Cas(
    NomJour( Obtenir( DateActuelle ) ) = "Mardi";
    "Oranges";
    NomJour( Obtenir( DateActuelle ) ) = "Jeudi";
    "Pommes";
    /* sinon */
    "Bananes");
  "Poires") & " and " &
Si(
  Position( selection_menu; "Autre legume"; 0; 1 );
  Cas(
    NomJour( Obtenir( DateActuelle ) ) = "Jeudi";
    "Pois";
    /* sinon */
    "Carottes");
  "Maïs")

```

Il est possible d'indenter le code qui ne tient pas sur une même ligne et passe à la ligne suivante.

```

Fin(
  Extrait( 10 ^ 11 + Arrondi( Abs( Montant ); Precision ); 1; 3 ) & "," &
  Extrait( 10 ^ 11 + Arrondi( Abs( Montant ); Precision ); 4; 3 );
  Longueur( Ent( Arrondi( Abs( Montant ); Precision ) ) ) +
  Si( Precision > 0; 1 + Precision; 0 ) +
  Ent(
    ( Longueur( Ent( Arrondi( Abs( Montant ); Precision ) ) ) - 1 ) / 3
  )
)

```

Sans indentation, le résultat se présenterait de la manière suivante :

```

Fin(
  Extrait( 10 ^ 11 + Arrondi( Abs( Montant ); Precision ); 1; 3 ) & "," &
  Extrait( 10 ^ 11 + Arrondi( Abs( Montant ); Precision ); 4; 3 );
  Longueur( Ent( Arrondi( Abs( Montant ); Precision ) ) ) +
  Si( Precision > 0; 1 + Precision; 0 ) +
  Ent(
    ( Longueur( Ent( Arrondi( Abs( Montant ); Precision ) ) ) - 1 ) / 3
  )
)

```

8.4 Considérations annexes

Mise en forme automatique des calculs FileMaker : Debi Fuchs, d'Aptworks Consulting, a créé un outil de mise en forme automatique (en anglais) capable d'effectuer tout ce travail à votre place. Vous pouvez l'essayer à l'adresse <http://www.apptworks.com/tools>. Toutefois, cet outil n'indente pas le code qui passe à la ligne. Le développeur peut compléter cette mise en forme en ajoutant sa propre indentation s'il le souhaite.



9 ► Listes de valeurs

L'impact des conventions sur les listes de valeurs est relativement limité. Toutefois, un certain nombre de recommandations générales peuvent aider à éliminer les problèmes courants et faciliter l'organisation et les extensions. Il est préférable de n'utiliser que des caractères qui ne posent pas de problèmes avec les technologies externes, telles que XML, qui référencent des noms de listes de valeurs. Vous devez également choisir une casse précise et vous y tenir. Certains développeurs incluent des métadonnées pour insérer des commentaires ou identifier un type de liste. Si c'est votre cas, réfléchissez à l'endroit où placer ces métadonnées et à la forme à leur donner. Il peut être intéressant de placer les métadonnées à la fin du nom. Ainsi, il reste possible d'utiliser le tri alphabétique dans les noms de listes, mais aussi la saisie partielle à partir du nom de liste plutôt qu'à partir des métadonnées. Il est également plus facile de présenter les listes sous le format désiré. Il faut aussi réfléchir à la forme que doivent prendre les métadonnées. Ce document utilise un espace réservé qui définit une certaine notation de base mais il offre également une certaine flexibilité au développeur en lui permettant de définir cette partie de la syntaxe en fonction de ses propres besoins. Toute extension doit être documentée dans la section « Conformité » de votre solution. (Pour plus d'informations, reportez-vous à la section [Conformité](#).)

9.1 Objectifs :

- ✓ Jeu de caractères recommandé
- ✓ Convention recommandée pour la casse
- ✓ Séparation syntaxique recommandée
- ✓ Consignes pour la longueur
- ✓ Recommandations liées à l'usage du singulier et du pluriel
- ✓ Méthode pour prendre en charge les commentaires dans les listes de valeurs
- ✓ Méthode pour prendre en charge les métadonnées (Source/Type) hors de la boîte de dialogue Définir les listes de valeurs
- ✓ Méthodes pour organiser les listes de valeurs autrement qu'en les triant manuellement
- ✓ Remarque : Problèmes liés à la mise à jour des listes de valeurs personnalisées indépendamment des données ou du schéma dans un fichier.
- ✓ Liens de calcul vers un nom plutôt que vers un ID de liste de valeur

9.2 Définition du problème :

Jeu de caractères recommandé : même si les caractères choisis pour une liste de valeurs ne sont pas appropriés, l'impact est minime. Toutefois, un certain nombre de facteurs doivent être pris en considération. Le développeur doit choisir des caractères qui n'entrent pas en conflit avec les calculs FileMaker susceptibles d'interagir avec des noms de listes de valeurs. En outre, le développeur doit se méfier des caractères qui risquent de poser des problèmes avec les requêtes XML. En respectant ces recommandations pour choisir votre jeu de caractères, vous devriez éviter les problèmes de ce type.

Convention recommandée pour la casse : le mode de séparation des mots au sein des listes de valeurs a un impact sur la cohérence générale des conventions. La recommandation générale est de choisir l'une des casses suivantes pour les noms de listes de valeurs et de se tenir à ce choix : casseChameauMinuscule, CasseChameauMajuscule, Caractères de tiret bas ou MAJUSCULES.

Séparation syntaxique recommandée : si un développeur souhaite inclure des métadonnées à propos d'une liste de valeurs, même dans une certaine mesure seulement, il est important que les développeurs actuels et futurs, ainsi que l'utilisateur final, sachent clairement ce qui est utilisé pour séparer le nom de la rubrique des métadonnées/de la notation. Les caractères choisis pour séparer les métadonnées du nom d'une liste de valeurs doivent être différents de ceux utilisés pour séparer les différents mots d'un nom de liste de valeurs.



Longueur du nom d'une liste de valeurs : le nom d'une liste de valeurs FileMaker peut contenir jusqu'à 100 caractères. Toutefois, souvent, une telle longueur n'est guère pratique. A partir de la version 8, sous Mac OS, toutes les boîtes de dialogue peuvent afficher le nom complet sans problème. Sous Windows, toutefois, il peut arriver que les noms de listes comportant plus de 30 caractères ne s'affichent pas correctement. Le nom complet est toutefois accessible dans la boîte de dialogue Définir les listes de valeurs. Le choix de la longueur est donc plus une question de préférences personnelles que de limite précise.

Recommandations liées à l'usage du singulier et du pluriel : la distinction entre le singulier et le pluriel n'est faite ici que pour signaler qu'en tant que développeur, vous devez faire un choix et vous y tenir.

9.3 Recommandations intégrant les standards (listes de valeurs) :

1. Le nom d'une liste de valeurs ne doit utiliser que les caractères
 - alphabétiques majuscules et minuscules aA - zZ ;
 - numériques 0, 1, 2, 3, 4, 5, 6, 7, 8 et 9 ;
 - de tiret bas simple et double « _ » « __ ».
2. Il ne doit pas PAS contenir d'espaces.
3. Les noms de listes de valeurs doivent être systématiquement au singulier ou au pluriel.
4. Les différents mots composant le nom d'une liste de valeurs doivent être séparés de manière cohérente, à l'aide de l'une des méthodes suivantes.

Méthodes recommandées pour la séparation des mots (noms de listes de valeurs)

casseChameauMinuscule	maListeValeur
CasseChameauMajuscule	MaListeValeur
Tiret bas simple (minuscules)	ma_liste_valeur
Tiret bas simple (majuscule à chaque initiale)	Ma_Liste_Valeur
Tiret bas simple (MAJUSCULES)	MA_LISTE_VALEUR

5. Syntaxe :

NomListeValeur[__]{suffixe}

(Voir la [légende de la syntaxe](#) pour une description de la syntaxe.)

Explication de la syntaxe : La syntaxe des listes de valeurs est conçue pour résoudre la plupart des problèmes susceptibles de se produire avec les listes de valeurs. L'utilisation d'un suffixe plutôt que d'un préfixe offre au développeur une plus grande souplesse pour les noms de listes de valeurs tout en assurant une certaine cohérence. Ce système permet d'insérer des métadonnées de base ainsi que de grouper une liste de valeurs par fonction.

- NomListeValeur
suit les recommandations ci-dessus, avec deux caractères de tiret bas « __ » ensuite.
- « __ » – requis
Les deux caractères de tiret bas font office de séparateur entre le nom de la liste de valeurs et le suffixe.
- suffixe – facultatif ; défini par le développeur (recommandation de base fournie)
Il peut s'agir de l'un des éléments du tableau suivant mais la liste de ce tableau n'est pas exhaustive. Toutefois, toute extension doit prendre la forme d'un suffixe afin de permettre de trier les éléments sur la base du nom et d'utiliser la saisie partielle à partir du nom plutôt qu'à partir du préfixe de métadonnées. Toute extension doit être documentée dans la section « Conformité » de votre solution. (Pour plus d'informations, reportez-vous à la section [Conformité](#).) Le document « Conventions de développement FileMaker » recommande les suffixes suivants :



Suffixes de base recommandés pour les listes de valeurs

- c** Valeur personnalisée (c est l'abréviation de "Custom", "Personnalisée")
- x** Liste de valeurs provenant d'un autre fichier
- d** Liste de valeurs provenant d'une rubrique, incluant toutes les valeurs (d est l'abréviation de "Dynamic", "Dynamique")
- r** Liste de valeurs provenant d'une rubrique, incluant des valeurs liées (r est l'abréviation de "Related", "Lié")



10 ► Comptes et sécurité

Les comptes FileMaker peuvent être configurés pour une authentification interne ou externe. Les comptes internes identifient de manière unique un utilisateur donné et sont gérés entièrement au sein de FileMaker. Les comptes authentifiés en externe identifient de manière unique un groupe externe, qui peut contenir un ou plusieurs utilisateurs. L'appartenance à un groupe externe est gérée hors de FileMaker. FileMaker utilise des « jeux de privilèges » pour déterminer les droits dont un compte authentifié bénéficie au sein de la base de données. La Figure 14 montre comment ces composants sont liés les uns aux autres.

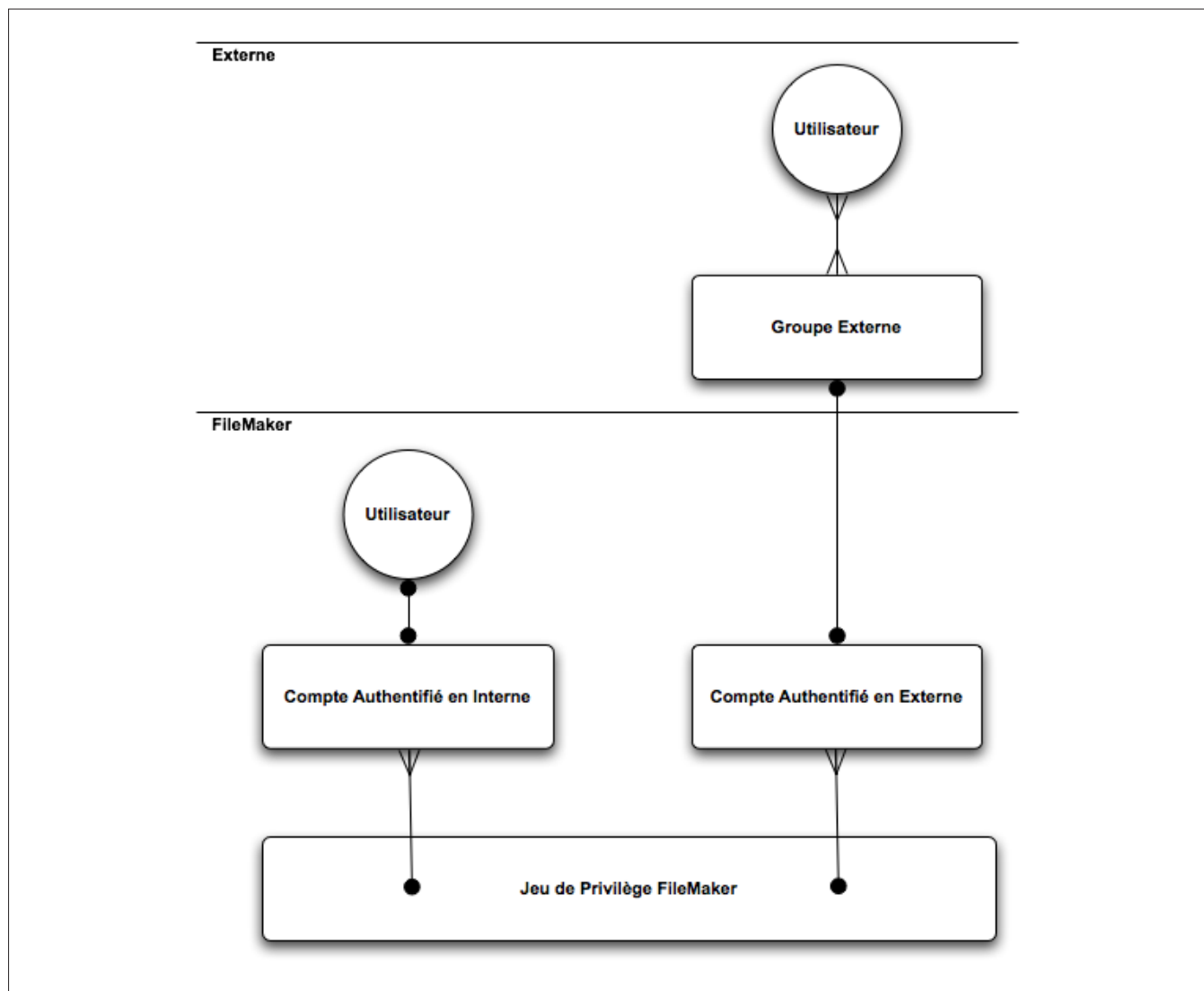


Figure 14

Le jeu de privilèges définit les droits dont un utilisateur ou un groupe d'utilisateurs bénéficie au sein du fichier. Il représente en quelque sorte son rôle. Par exemple, vous pouvez avoir des comptes Développeur, Administrateur, Lecture seule, Saisie données, etc. Le nom doit représenter le rôle. C'est particulièrement important quand l'administration a lieu hors de FileMaker.

L'un des principaux avantages de l'authentification externe est qu'elle permet de tirer parti de l'infrastructure de sécurité de l'entreprise. Cela permet d'administrer à partir d'un même endroit toutes les ressources disponibles pour tous les utilisateurs dans le cadre de l'entreprise. Cela signifie aussi que les administrateurs système, qui ne connaissent pas forcément les caractéristiques de votre solution ou de l'ensemble des solutions de l'environnement, doivent définir l'appartenance par nom. Vous devrez articuler clairement les fichiers ou la solution et le rôle pour cette appartenance. Une certaine cohérence doit par ailleurs être respectée pour les jeux de privilèges et les comptes authentifiés en externe.



Quand plusieurs composants sont combinés, ils doivent être inclus dans le nom du compte ou groupe authentifié en externe pour être identifiés. Cette combinaison doit contenir au minimum le nom de la solution afin d'identifier le groupe associé à la solution. Elle doit également contenir le rôle de cette solution. Par exemple, imaginons que nous ayons une solution appelée Suivi des ventes. Elle contient trois jeux de privilèges : Opérateurs, VentesAmeriqueNord et VentesMondiales. Imaginons maintenant que nous ayons une autre solution appelée Suivi des expéditions, contenant les mêmes jeux de privilèges. L'hypothèse de départ est que chaque rôle ne correspond pas au même utilisateur pour chaque solution, ce qui serait d'ailleurs difficilement possible dans la plupart des cas. Vous devez alors spécifier la solution et le rôle que l'utilisateur doit recevoir. Pour ce faire, vous devez disposer d'une convention de dénomination qui utilise à la fois le jeu de privilèges et le nom de la solution pour l'authentification externe.

10.1 Objectifs :

- ✓ Jeu de caractères recommandé
- ✓ Convention recommandée pour la casse
- ✓ Séparation syntaxique recommandée
- ✓ Consignes pour la longueur
- ✓ Méthode pour identifier aisément une association de groupe externe à partir du nom pour une solution donnée
- ✓ Méthode pour identifier aisément des autorisations de groupe externes à partir du nom pour une solution donnée
- ✓ Méthode pour prendre en charge les solutions dupliquées sur des serveurs différents avec l'authentification externe

10.2 Définition du problème

Jeu de caractères recommandé : les caractères autorisés par FileMaker et par le système d'exploitation en charge de l'authentification sont soumis à un certain nombre de restrictions de base. Afin d'éviter des problèmes de caractères, il est recommandé d'utiliser les caractères ACSII majuscules et minuscules et les chiffres.

Convention recommandée pour la casse : le choix fait ici a un impact sur les autres endroits où une convention s'applique. La décision n'a pas pour but de résoudre un problème particulier mais de définir une approche cohérente de la casse dans toutes les autres conventions. L'objectif est de sélectionner une méthode et de s'y tenir.

Séparation syntaxique recommandée : dans le domaine de la dénomination des comptes, la séparation syntaxique désigne la manière de différencier la syntaxe du nom. Les développeurs utilisent souvent des préfixes ou des suffixes pour indiquer qu'un fichier fait partie d'un groupe ou pour signaler des caractéristiques d'identification. Il est souhaitable d'employer une méthode uniforme pour différencier les métadonnées sous forme de préfixes ou de suffixes du nom du compte.

Consignes pour la longueur : les noms de jeux de privilèges et de comptes sont limités à 100 caractères. Sous Windows Server 2003, les noms de groupes sont limités à 64 caractères. Sous Mac OS X Server, les noms de groupes peuvent comporter jusqu'à 100 caractères.

Identification de la solution : spécifique à l'authentification externe. Les groupes créés hors de l'environnement de développement doivent être rigoureusement identiques à ceux créés au sein de l'environnement de développement dans les fichiers FileMaker Pro. Sachant que beaucoup d'entreprises utilisent un grand nombre de solutions, il peut être difficile d'identifier les groupes qui ne concernent qu'une solution donnée. La tâche est toutefois beaucoup plus facile quand le nom de la solution est inclus dans celui du groupe.

Identification des autorisations (rôles) : spécifique à l'authentification externe. Les groupes créés hors de l'environnement de développement doivent être rigoureusement identiques à ceux créés au sein de l'environnement de développement. Sachant que beaucoup d'entreprises utilisent un grand nombre de solutions, il peut être difficile d'identifier les groupes qui ne concernent qu'une solution donnée. Il faut en outre localiser le rôle correspondant à cette solution. En incluant le rôle dans le nom, le problème ne se pose plus.



Solutions dupliquées sur des serveurs séparés : dans certains cas, une même solution est hébergée par plusieurs serveurs. Toutes les copies sont rigoureusement identiques mais elles sont destinées à des utilisateurs différents. Il convient alors de compléter les recommandations générales pour prendre en charge cette situation particulière et plutôt rare.

10.3 Recommandations intégrant les standards (sécurité) :

10.3.1 Jeux de privilèges

1. Le nom d'un jeu de privilèges ne doit utiliser que les caractères
 - alphabétiques majuscules et minuscules aA - zZ ;
 - numériques 0, 1, 2, 3, 4, 5, 6, 7, 8 et 9 ;
 - de tiret bas simple et double « _ » « __ ».
2. Il ne doit pas PAS contenir d'espaces.
3. Les noms de jeux de privilèges doivent être systématiquement au singulier ou au pluriel.
4. Ils ne doivent pas dépasser 100 caractères. (Si vous suivez les conventions recommandées pour la longueur des noms de comptes authentifiés en externe, vous devez prendre en considération la longueur totale de la syntaxe, qui ne doit pas dépasser 64 caractères.)
5. Les différents mots composant le nom d'une liste de valeurs doivent être séparés de manière cohérente, à l'aide de l'une des méthodes suivantes.

Méthodes recommandées pour la séparation des mots (jeux de privilèges)

casseChameauMinuscule	monPrivilege
CasseChameauMajuscule	MonPrivilege
Tiret bas simple (minuscules)	mon_privilege
Tiret bas simple (majuscule à chaque initiale)	Mon_Privilege
Tiret bas simple (MAJUSCULES)	MON_PRIVILEGE

10.3.2 Noms de comptes authentifiés en interne

1. Le nom d'un compte authentifié de en interne ne doit utiliser que les caractères
 - alphabétiques majuscules et minuscules aA - zZ ;
 - numériques 0, 1, 2, 3, 4, 5, 6, 7, 8 et 9 ;
 - de tiret bas simple et double « _ » « __ ».
2. Il ne doit pas PAS contenir d'espaces.
3. Les noms de comptes authentifiés en interne doivent être systématiquement au singulier ou au pluriel.
4. Ils ne doivent pas dépasser 100 caractères.
5. Les différents mots composant le nom d'un compte authentifié en interne doivent être séparés de manière cohérente, à l'aide de l'une des méthodes suivantes.

Méthodes recommandées pour la séparation des mots (comptes authentifiés en interne)

casseChameauMinuscule	nomCompte
CasseChameauMajuscule	NomCompte
Tiret bas simple (minuscules)	nom_compte
Tiret bas simple (majuscule à chaque initiale)	Nom_Compte
Tiret bas simple (MAJUSCULES)	NOM_COMPTE



10.3.3 Noms de comptes authentifiés en externe (noms de groupes)

1. Le nom d'un compte authentifié en externe ne doit utiliser que les caractères
 - alphabétiques majuscules et minuscules aA - zZ ;
 - numériques 0, 1, 2, 3, 4, 5, 6, 7, 8 et 9 ;
 - de tiret bas simple et double « _ » « __ ».
2. Il ne doit pas PAS contenir d'espaces.
3. Les noms de comptes authentifiés en externe doivent être systématiquement au singulier ou au pluriel.
4. Ils ne doivent pas comporter plus de 64 caractères pour fonctionner à la fois sous Windows et sous Mac OS X.
5. Les différents mots composant le nom d'un compte authentifié en externe doivent être séparés de manière cohérente, à l'aide de l'une des méthodes suivantes.

Méthodes recommandées pour la séparation des mots (comptes authentifiés en externe)

casseChameauMinuscule	nomCompte
CasseChameauMajuscule	NomCompte
Tiret bas simple (minuscules)	nom_compte
Tiret bas simple (majuscule à chaque initiale)	Nom_Compte
Tiret bas simple (MAJUSCULES)	NOM_COMPTE

6. Syntaxe :

<<DesignationGroupeFilemaker>>[_]<NomSolution>[_]<NomPrivilege>

La syntaxe est conçue pour offrir une manière cohérente de nommer les groupes authentifiés en externe. Elle facilite la gestion des groupes externes en offrant une convention qui identifie le groupe en tant que groupe de sécurité FileMaker, une solution (ou un fichier) précise, et le rôle du jeu de privilèges. Avec ces trois éléments, les groupes sont bien organisés et plus simples à gérer. Deux caractères de tiret bas « _ » sont utilisés pour séparer chaque section. Ces caractères sont compatibles avec les versions d'Active Directory antérieures et postérieures à Windows 2000. Ils sont également pris en charge par Open Directory. Avec les versions antérieures à Windows 2000, les noms de groupes ne peuvent pas contenir les caractères : ; | = + , * ? < > " / \ [] .

REMARQUE : Dans certains cas, les noms de groupes peuvent être dictés par les règles d'organisation. Dans pareil cas, vous devez utiliser les règles en vigueur pour adapter votre convention si nécessaire.

- DesignationGroupeFilemaker – requis ; défini par le développeur
Identifie le groupe en tant que groupe de sécurité FileMaker.
Cela permet de rassembler tous les groupes de sécurité FileMaker ensemble au sein de l'outil Workgroup Manager. La recommandation est d'utiliser « fm », en minuscules. Ce choix est arbitraire mais recommandé pour la cohérence de l'organisation.
- NomSolution – identifie la solution (ou le fichier) précise à laquelle le groupe est associé. Ce système permet d'effectuer des tris et de regrouper tous les groupes se rapportant à une solution donnée. Le nom doit correspondre exactement à celui que vous avez choisi pour le fichier principal ou aux fichiers de point d'entrée, sans l'extension .fpX.
- NomJeuPrivileges – requis ; utilise le nom du jeu de privilèges
Identifie le rôle des membres du groupe ou les droits qu'ils ont au sein de la solution.
Le fait d'inclure cet élément dans le nom de compte ou de groupe permet d'identifier facilement le groupe qui doit être modifié et facilite la gestion externe.
- Jeux de privilèges spéciaux – FileMaker intègre trois jeux de privilèges prédéfinis :
 - ▶ [Accès intégral] ([Full Access])
 - ▶ [Saisie de données uniquement] ([Data Entry Only])
 - ▶ [Accès en lecture seule] ([Read-Only Access]).



Ces jeux de privilèges ne sont pas modifiables et ils ne peuvent pas être renommés ni supprimés. Pour utiliser un ou plusieurs de ces jeux de privilèges, omettez les crochets [] autour du nom du jeu de privilèges pour configurer le compte de groupe authentifié en externe. Sous les versions de Windows antérieures à la version 2000, ces caractères sont interdits dans les noms de groupes.

[Accès en lecture seule] = `Acces_En_Lecture_Seule`, `AccesEnLectureSeule`, `accesEnLectureSeule`, `ACCES_EN_LECTURE_SEULE`

ou

[Read Only] = `Read_Only`, `ReadOnly`, `readOnly`, `READ_ONLY`

Exemple : `fm_MaSolution_AccesEnLectureSeule`

ou

Exemple : `fm_MaSolution_Read_Only`

7. Accès intégral avec les comptes authentifiés en externe : Dans le livre (en anglais) *FileMaker Security: The Book*, de Steven H. Blackwell, l'auteur commente l'utilisation du jeu de privilèges Accès intégral avec les comptes authentifiés en externe : « Je conseille vivement aux développeurs de ne jamais soumettre un compte configuré pour utiliser le jeu de privilèges [Accès intégral] par défaut à l'authentification externe. Un tel compte doit être authentifié exclusivement à l'aide de la méthode interne de FileMaker Pro. Premièrement, chaque fichier doit avoir au moins un compte [Accès intégral] authentifié en interne. Deuxièmement, dans leur version initiale, FileMaker Server 7 et FileMaker Server 7 Advanced ne prennent pas en charge le système GUID (Global Universal ID, ID universel global). Si un pirate ou un espion industriel obtient une copie physique d'un fichier FileMaker Pro et parvient à recréer ou à imiter la structure de domaine en devinant les noms de groupes, il aura un accès illimité au fichier. C'est d'ailleurs une autre raison d'utiliser des noms de groupes distinctifs. » Le document « Convention de développement FileMaker » cautionne cette vision des choses.
8. Une autre note spéciale concerne les noms de groupes dans l'exposé technique FileMaker, Inc. consacré à l'authentification externe par l'intermédiaire d'un serveur, disponible à l'adresse <http://www.filemaker.fr/products/upgraded/techbriefs.html>

« Sous Mac OS X Server, FileMaker Server recherche le nom abrégé du groupe tel qu'il est renvoyé par les services d'annuaire. Il s'agit du nom officiel qui identifie le groupe auprès du système, et non du nom long (convivial). Ainsi, dans la définition des comptes dans FileMaker Pro pour l'authentification externe par l'intermédiaire d'un serveur, le nom de groupe défini doit correspondre au nom abrégé du groupe tel qu'il figure dans le service d'annuaire. Souvent, le nom long et le nom court sont identiques mais il arrive qu'ils ne le soient pas. Les espaces et les caractères au format ASCII supérieur peuvent être supprimés, par exemple. Aussi recommandons-nous de les éviter. Les développeurs et les administrateurs doivent vérifier le nom abrégé. »

10.4 Considérations annexes :

Désignation du serveur hôte : dans certains cas, une même solution est hébergée par plusieurs serveurs. Toutes les copies sont rigoureusement identiques mais elles sont destinées à des utilisateurs différents. Dans pareil cas, il peut être intéressant d'ajouter un élément à la syntaxe du nom de groupe externe. En incluant la désignation du serveur hôte, il est plus facile de définir l'appartenance de groupe pour la solution sur le serveur concerné.

Syntaxe :

<<DesignationGroupeFilemaker>>[_]<ServeurHote>[_]<NomSolution>[_]<NomPrivilege>

ServeurHote – doit correspondre au nom DNS, au nom système ou au nom personnalisé du serveur d'hébergement.

Généralement, ces noms sont identiques mais il arrive qu'ils ne le soient pas. Choisissez le nom **ServeurHote** qui identifie le serveur tel qu'il est reconnu par l'entreprise.

Exemples :

- Syntaxe standard :
 - `fm_MaSolution_AccesEnLectureSeule`
 - `fm_MaSolution_General_Utilisateurs`
 - `fm_MaSolution_associésCommerciaux`



- Utilisation des considérations annexes pour le serveur hôte :
 - fm_fmserveur1__MaSolution__Utilisateurs généraux
 - fm_fmserveur2__MaSolution__Utilisateurs généraux



11 ► Conformité

Cette section part de l'hypothèse que vous avez trouvé un intérêt au document « Conventions de développement FileMaker » et que vous êtes prêt à intégrer à votre solution ne serait-ce qu'une partie des recommandations qu'il contient à propos des standards. Une grande partie des informations contenues dans ce document peuvent donner lieu à des extensions ou à des adaptations pour différents motifs. Aussi est-il intéressant de documenter ces écarts par rapport à la recommandation de base et d'utiliser une approche cohérente à cet effet. Par conformité, nous n'entendons pas réellement le fait de suivre strictement les règles présentées dans ce document mais le fait de documenter les endroits où le développeur s'écarte de la règle. L'objectif est de considérer le document « Conventions de développement FileMaker » comme la référence de base et de limiter le travail à fournir pour documenter le pourquoi et le comment de votre propre convention. En considérant ce document comme votre référence de base, vous réduirez considérablement ce travail. Vous devez justement signaler à quels endroits vous vous écartez ou vous allez au-delà de ce document. Ainsi, les futurs développeurs pourront se référer aux principes de base du système de dénomination quelles que soient les considérations supplémentaires que vous choisirez d'intégrer.

11.1 Objectifs de la convention

- ✓ Référencer la version utilisée du document « Conventions de développement FileMaker »
- ✓ Référencer la version des écarts par rapport à ce document »
- ✓ Référencer les autres informations notables relatives à la solution
- ✓ Définir le stockage et l'accessibilité des données
- ✓ Définir le jeu minimal d'informations

11.2 Définition du problème

Référencer la version utilisée du document « Conventions de développement FileMaker » : au fur et à mesure que la gamme de produits FileMaker évoluera, le document « Conventions de développement FileMaker » évoluera lui aussi afin de s'adapter. Conséquence logique, une convention définie pour une solution repose sur une version donnée de ces documents et de vos propres extensions. Il est donc important d'indiquer la version du document « Conventions de développement FileMaker ».

Référencer la version des écarts par rapport à ce document : comme le document « Conventions de développement FileMaker » lui-même, vos propres écarts et extensions seront probablement appelés à changer. Il est donc là aussi important d'indiquer leur version.

Référencer les autres informations notables relatives à la solution : certains projets ou solutions nécessitent d'indiquer la version, le nom du créateur et d'autres éléments de métadonnées. Toutefois, le concept de version ne s'applique qu'à une phase limitée du cycle de vie du produit, qui correspond au développement initial. Ces éléments sont moins intéressants à long terme ou au cours de la phase d'évolution d'un projet. Même si l'essentiel du travail de développement initial est réalisé par un même développeur, de nombreux autres développeurs peuvent intervenir dans le cycle de vie du projet. Toutefois, pour une période donnée ou pour une solution particulière, il est intéressant de disposer de ces métadonnées.

Définir le stockage et l'accessibilité des données : Pour des raisons de cohérence, les informations doivent être disponibles à un endroit universel.

Définir le jeu minimal d'informations : on considère généralement que toute solution doit avoir une documentation d'une forme ou d'une autre. Certaines solutions contiennent même leur propre système d'aide, qui peut inclure des informations pour l'utilisateur et pour le développeur. Au lieu d'imposer une méthode particulière pour l'élaboration d'une documentation ou d'un système d'aide, la recommandation doit simplement mettre en évidence les éléments minimaux requis et indiquer comment y accéder.



11.3 Recommandations pour une convention intégrant les standards

Chaque solution doit comporter une section « A propos de... » Tout utilisateur ou développeur disposant de privilèges à cet effet doit pouvoir obtenir les éléments suivants en ouvrant le fichier.

- Version utilisée du document « Conventions de développement FileMaker »
 - URL permettant d'accéder au document « Conventions de développement FileMaker » sur le site Web de FileMaker
 - Document « Conventions de développement FileMaker » stocké au format PDF dans une rubrique multimédia exportable
- Ecarts par rapport au document « Conventions de développement FileMaker » avec le numéro de version
 - URL permettant d'accéder au document contenant les écarts par rapport au document « Conventions de développement FileMaker » avec le numéro de version sur votre site Web
 - Document stocké au format PDF dans une rubrique multimédia exportable contenant tous les écarts et extensions applicables à la solution
- Métadonnées de documentation – autres informations, à la discrétion du développeur, considérées comme nécessaires. Ces informations peuvent être stockées dans une ou plusieurs rubriques. Elles doivent être disponibles à partir de l'interface « A propos de » qui est présentée. Elles peuvent inclure des éléments tels que :
 - le nom de l'entreprise ;
 - la version de la solution ;
 - les remerciements ;
 - la date de publication ;
 - le nom du développeur ;
 - les informations de contact.

Remarque :

Les développeurs de versions d'exécution sont soumis à des obligations légales concernant la structure de la section « A propos de ». Pour plus d'informations, reportez-vous au guide de développement FileMaker.



Annexe A ▷ Mots réservés SQL

Le tableau suivant dresse la liste de tous les mots réservés du langage SQL.

Source : <http://developer.mimer.se/validator/sql-reserved-words.tml>

SQL-92	SQL-99	SQL-2003
ABSOLUTE	ABSOLUTE	
ACTION	ACTION	
ADD	ADD	ADD
	AFTER	ALL
ALL	ALL	
ALLOCATE	ALLOCATE	ALLOCATE
ALTER	ALTER	ALTER
AND	AND	AND
ANY	ANY	ANY
ARE	ARE	ARE
	ARRAY	ARRAY
AS	AS	AS
ASC	ASC	
	ASENSITIVE	ASENSITIVE
ASSERTION	ASSERTION	
	ASYMMETRIC	ASYMMETRIC
AT	AT	AT
	ATOMIC	ATOMIC
AUTHORIZATION	AUTHORIZATION	AUTHORIZATION
AVG		
	BEFORE	
BEGIN	BEGIN	BEGIN
BETWEEN	BETWEEN	BETWEEN
		BIGINT
	BINARY	BINARY
BIT	BIT	
BIT_LENGTH		
	BLOB	BLOB
	BOOLEAN	BOOLEAN
BOTH	BOTH	BOTH
	BREADTH	
BY	BY	BY
CALL	CALL	CALL
	CALLED	CALLED
CASCADE	CASCADE	
CASCADED	CASCADED	CASCADED
CASE	CASE	CASE



SQL-92	SQL-99	SQL-2003
CAST	CAST	CAST
CATALOG	CATALOG	
CHAR	CHAR	CHAR
CHAR_LENGTH		
CHARACTER	CHARACTER	CHARACTER
CHARACTER_LENGTH		
CHECK	CHECK	CHECK
	CLOB	CLOB
CLOSE	CLOSE	CLOSE
COALESCE		
COLLATE	COLLATE	COLLATE
COLLATION	COLLATION	
COLUMN	COLUMN	COLUMN
COMMIT	COMMIT	COMMIT
CONDITION	CONDITION	CONDITION
CONNECT	CONNECT	CONNECT
CONNECTION	CONNECTION	
CONSTRAINT	CONSTRAINT	CONSTRAINT
CONSTRAINTS	CONSTRAINTS	
	CONSTRUCTOR	
CONTAINS		
CONTINUE	CONTINUE	CONTINUE
CONVERT		
CORRESPONDING	CORRESPONDING	CORRESPONDING
COUNT		
CREATE	CREATE	CREATE
CROSS	CROSS	CROSS
	CUBE	CUBE
CURRENT	CURRENT	CURRENT
CURRENT_DATE	CURRENT_DATE	CURRENT_DATE
	CURRENT_DEFAULT_TRANSFORM_	
GROUP	CURRENT_DEFAULT_TRANSFORM_	
GROUP		
CURRENT_PATH	CURRENT_PATH	CURRENT_PATH
	CURRENT_ROLE	CURRENT_ROLE
CURRENT_TIME	CURRENT_TIME	CURRENT_TIME
CURRENT_TIMESTAMP	CURRENT_TIMESTAMP	CURRENT_TIMESTAMP
	CURRENT_TRANSFORM_GROUP_FO	



SQL-92	SQL-99	SQL-2003
R_TYPE	CURRENT_TRANSFORM_GROUP_ FO	
R_TYPE		
CURRENT_USER	CURRENT_USER	CURRENT_USER
CURSOR	CURSOR	CURSOR
	CYCLE	CYCLE
	DATA	
DATE	DATE	DATE
DAY	DAY	DAY
DEALLOCATE	DEALLOCATE	DEALLOCATE
DEC	DEC	DEC
DECIMAL	DECIMAL	DECIMAL
DECLARE	DECLARE	DECLARE
DEFAULT	DEFAULT	DEFAULT
DEFERRABLE	DEFERRABLE	
DEFERRED	DEFERRED	
DELETE	DELETE	DELETE
DEPTH		
	DEREF	DEREF
DESC	DESC	
DESCRIBE	DESCRIBE	DESCRIBE
DESCRIPTOR	DESCRIPTOR	
DETERMINISTIC	DETERMINISTIC	DETERMINISTIC
DIAGNOSTICS	DIAGNOSTICS	
DISCONNECT	DISCONNECT	DISCONNECT
DISTINCT	DISTINCT	DISTINCT
DO	DO	DO
DOMAIN	DOMAIN	
DOUBLE	DOUBLE	DOUBLE
DROP	DROP	DROP
	DYNAMIC	DYNAMIC
	EACH	EACH
		ELEMENT
ELSE	ELSE	ELSE
ELSEIF	ELSEIF	ELSEIF
END	END	END
	EQUALS	
ESCAPE	ESCAPE	ESCAPE
EXCEPT	EXCEPT	EXCEPT
EXCEPTION	EXCEPTION	
EXEC	EXEC	EXEC
EXECUTE	EXECUTE	EXECUTE



SQL-92	SQL-99	SQL-2003
EXISTS	EXISTS	EXISTS
EXIT	EXIT	EXIT
EXTERNAL	EXTERNAL	EXTERNAL
EXTRACT		
FALSE	FALSE	FALSE
FETCH	FETCH	FETCH
	FILTER	FILTER
FIRST	FIRST	
FLOAT	FLOAT	FLOAT
FOR	FOR	FOR
FOREIGN	FOREIGN	FOREIGN
FOUND	FOUND	
	FREE	FREE
FROM	FROM	FROM
FULL	FULL	FULL
FUNCTION	FUNCTION	FUNCTION
	GENERAL	
GET	GET	GET
GLOBAL	GLOBAL	GLOBAL
GO	GO	
GOTO	GOTO	
GRANT	GRANT	GRANT
GROUP	GROUP	GROUP
	GROUPING	GROUPGROUPING
HANDLER	HANDLER	HANDLER
HAVING	HAVING	HAVING
	HOLD	HOLD
HOUR	HOUR	HOUR
IDENTITY	IDENTITY	IDENTITY
IF	IF	IF
IMMEDIATE	IMMEDIATE	IMMEDIATE
IN	IN	IN
INDICATOR	INDICATOR	INDICATOR
INITIALLY	INITIALLY	
INNER	INNER	INNER
INOUT	INOUT	INOUT
INPUT	INPUT	INPUT
INSENSITIVE	INSENSITIVE	INSENSITIVE
INSERT	INSERT	INSERT
INT	INT	INT
INTEGER	INTEGER	INTEGER
INTERSECT	INTERSECT	INTERSECT



SQL-92	SQL-99	SQL-2003
INTERVAL	INTERVAL	INTERVAL
INTO	INTO	INTO
IS	IS	IS
ISOLATION	ISOLATION	
	ITERATE	ITERATE
JOIN	JOIN	JOIN
KEY	KEY	
LANGUAGE	LANGUAGE	LANGUAGE
	LARGE	LARGE
LAST	LAST	
	LATERAL	LATERAL
LEADING	LEADING	LEADING
LEAVE	LEAVE	LEAVE
LEFT	LEFT	LEFT
LEVEL	LEVEL	
LIKE	LIKE	LIKE
LOCAL	LOCAL	LOCAL
	LOCALTIME	LOCALTIME
	LOCALTIMESTAMP	LOCALTIMESTAMP
	LOCATOR	
LOOP	LOOP	LOOP
LOWER		
	MAP	
MATCH	MATCH	MATCH
MAX		
		MEMBER
		MERGE
	METHOD	METHOD
MIN		
MINUTE	MINUTE	MINUTE
	MODIFIES	MODIFIES
MODULE	MODULE	MODULE
MONTH	MONTH	MONTH
		MULTISET
NAMES	NAMES	
NATIONAL	NATIONAL	NATIONAL
NATURAL	NATURAL	NATURAL
NCHAR	NCHAR	NCHAR
	NCLOB	NCLOB
NEXT	NEXT	
NO	NO	NO
	NONE	NONE



SQL-92	SQL-99	SQL-2003
NOT	NOT	NOT
NULL	NULL	NULL
NULLIF		
NUMERIC	NUMERIC	NUMERIC
	OBJECT	
OCTET_LENGTH		
OF	OF	OF
	OLD	OLD
ON	ON	ON
ONLY	ONLY	ONLY
OPEN	OPEN	OPEN
OPTION	OPTION	
OR	OR	OR
ORDER	ORDER	ORDER
	ORDINALITY	
OUT	OUT	OUT
OUTER	OUTER	OUTER
OUTPUT	OUTPUT	OUTPUT
	OVER	OVER
OVERLAPS	OVERLAPS	OVERLAPS
PAD	PAD	
PARAMETER	PARAMETER	PARAMETER
PARTIAL	PARTIAL	
	PARTITION	PARTITION
PATH	PATH	
POSITION		
PRECISION	PRECISION	PRECISION
PREPARE	PREPARE	PREPARE
PRESERVE	PRESERVE	
PRIMARY	PRIMARY	PRIMARY
PRIOR	PRIOR	
PRIVILEGES	PRIVILEGES	
PROCEDURE	PROCEDURE	PROCEDURE
PUBLIC	PUBLIC	
	RANGE	RANGE
READ	READ	
	READS	READS
REAL	REAL	REAL
	RECURSIVE	RECURSIVE
	REF	REF
REFERENCES	REFERENCES	REFERENCES
	REFERENCING	REFERENCING



SQL-92	SQL-99	SQL-2003
RELATIVE	RELATIVE	
	RELEASE	RELEASE
REPEAT	REPEAT	REPEAT
RESIGNAL	RESIGNAL	RESIGNAL
RESTRICT	RESTRICT	
	RESULT	RESULT
RETURN	RETURN	RETURN
RETURNS	RETURNS	RETURNS
REVOKE	REVOKE	REVOKE
RIGHT	RIGHT	RIGHT
ROLLBACK	ROLLBACK	ROLLBACK
	ROLLUP	ROLLUP
ROUTINE	ROUTINE	
	ROW	ROW
ROWS	ROWS	ROWS
	SAVEPOINT	SAVEPOINT
SCHEMA	SCHEMA	
	SCOPE	SCOPE
SCROLL	SCROLL	SCROLL
	SEARCH	SEARCH
SECOND	SECOND	SECOND
SECTION	SECTION	
SELECT	SELECT	SELECT
	SENSITIVE	SENSITIVE
SESSION	SESSION	
SESSION_USER	SESSION_USER	SESSION_USER
SET	SET	SET
	SETS	
SIGNAL	SIGNAL	SIGNAL
	SIMILAR	SIMILAR
SIZE	SIZE	
SMALLINT	SMALLINT	SMALLINT
SOME	SOME	SOME
SPACE	SPACE	
SPECIFIC	SPECIFIC	SPECIFIC
	SPECIFICTYPE	SPECIFICTYPE
SQL	SQL	SQL
SQLCODE		
SQLERROR		
SQLEXCEPTION	SQLEXCEPTION	SQLEXCEPTION
SQLSTATE	SQLSTATE	SQLSTATE
SQLWARNING	SQLWARNING	SQLWARNING



SQL-92	SQL-99	SQL-2003
	START	START
	STATE	
	STATIC	STATIC
		SUBMULTISET
SUBSTRING		
SUM		
	SYMMETRIC	SYMMETRIC
	SYSTEM	SYSTEM
SYSTEM_USER	SYSTEM_USER	SYSTEM_USER
TABLE	TABLE	TABLE
		TABLESAMPLE
TEMPORARY	TEMPORARY	
THEN	THEN	THEN
TIME	TIME	TIME
TIMESTAMP	TIMESTAMP	TIMESTAMP
TIMEZONE_HOUR	TIMEZONE_HOUR	TIMEZONE_HOUR
TIMEZONE_MINUTE	TIMEZONE_MINUTE	TIMEZONE_MINUTE
TO	TO	TO
TRAILING	TRAILING	TRAILING
TRANSACTION	TRANSACTION	
TRANSLATE		
TRANSLATION	TRANSLATION	TRANSLATION
	TREAT	TREAT
	TRIGGER	TRIGGER
TRIM		
TRUE	TRUE	TRUE
	UNDER	
UNDO	UNDO	UNDO
UNION	UNION	UNION
UNIQUE	UNIQUE	UNIQUE
UNKNOWN	UNKNOWN	UNKNOWN
	UNNEST	UNNEST
UNTIL	UNTIL	UNTIL
UPDATE	UPDATE	UPDATE
UPPER		
USAGE	USAGE	
USER	USER	USER
USING	USING	USING
VALUE	VALUE	VALUE
VALUES	VALUES	VALUES
VARCHAR	VARCHAR	VARCHAR
VARYING	VARYING	VARYING



SQL-92	SQL-99	SQL-2003
VIEW	VIEW	
WHEN	WHEN	WHEN
WHENEVER	WHENEVER	WHENEVER
WHERE	WHERE	WHERE
WHILE	WHILE	WHILE
	WINDOW	WINDOW
WITH	WITH	WITH
	WITHIN	WITHIN
	WITHOUT	WITHOUT
WORK	WORK	
WRITE	WRITE	
YEAR	YEAR	YEAR
ZONE	ZONE	



Annexe B ► Tableau sur l'utilisation des caractères

- [X] Non autorisé ou avertissement en cas de tentative d'utilisation
 [NR] Non recommandé
 [C] Utilisation en utilisant la Convention
 [R] Réservé

	Caractère	Noms de fichiers	Noms de tables	Noms d'occurrences de tables	Noms de rubriques	Noms de modèles	Noms de fonctions personnalisées	Noms de listes de valeurs	Noms de scripts	Noms de comptes	Variables
.	Point	X	X	X	X	NR	NR	NR	NR	NR	NR
+	Plus ou addition	NR	X	X	X	NR	X	NR	NR	NR	NR
*	Astérisque	NR	X	X	X	NR	X	NR	NR	NR	NR
^	Accent circonflexe	NR	X	X	X	NR	X	NR	NR	NR	NR
=	Egal	NR	X	X	X	NR	X	NR	NR	NR	NR
>	Supérieur à	NR	X	X	X	NR	X	NR	NR	NR	NR
(Parenthèse ouvrante	NR	X	X	X	NR	X	NR	NR	NR	NR
«	Guillemets doubles droits	NR	X	X	X	NR	X	NR	NR	NR	NR
:	Deux-points	NR	NR	NR	X	NR	NR	NR	NR	NR	NR
,	Virgule	NR	X	X	X	NR	X	NR	NR	NR	NR
-	Moins ou soustract°	NR	X	X	X	NR	X	NR	NR	NR	NR
/	Barre de fraction	NR	X	X	X	NR	X	NR	NR	NR	NR
&	Et commercial	NR	X	X	X	NR	X	NR	NR	NR	NR
≠	Pas égal à	NR	X	X	X	NR	X	NR	NR	NR	NR
<	Inférieur à	NR	X	X	X	NR	X	NR	NR	NR	NR
)	Parenthèse fermante	NR	X	X	X	NR	X	NR	NR	NR	NR
;	Point-virgule	NR	X	X	X	NR	X	NR	NR	NR	NR
::	Indicateur relationnel	NR	NR	X	X	NR	X	NR	NR	NR	NR
{	Accolade ouvrante	NR	NR	X	NR	NR	NR	NR	NR	NR	NR
}	Accolade fermante	NR	X	X	NR	NR	NR	NR	NR	NR	NR
?	Point d'interrogat°	NR	NR	NR	NR	NR	NR	NR	NR	NR	NR
~	Tilde	NR	NR	NR	NR	NR	NR	NR	NR	NR	NR
'	Apostrophe	NR	NR	NR	NR	NR	NR	NR	NR	NR	NR
!	Point d'exclamat°	NR	NR	NR	NR	NR	NR	NR	NR	NR	NR
%	Pour cent	NR	NR	X	NR	NR	NR	NR	NR	NR	NR



	Caractère	Noms de fichiers	Noms de tables	Noms d'occurrences de tables	Noms de rubriques	Noms de modèles	Noms de fonctions personnalisées	Noms de listes de valeurs	Noms de scripts	Noms de comptes	Variables
	Barre verticale	NR	NR	R	NR	NR	NR	NR	NR	NR	NR
\$	Dollar	NR	X	X	X	NR	NR	NR	NR	NR	R
\$\$	Double dollar	NR	X	X	X	NR	NR	NR	NR	NR	R
[Crochet ouvrant	NR	X	X	X	NR	X	NR	NR	NR	NR
]	Crochet fermant	NR	X	X	X	NR	X	NR	NR	NR	NR
—	Double tiret bas	C	C	C	C	C	C	C	C	C	NR



Annexe C ▷ Terminologie et définitions

Casse chameau : système d'écriture consistant à agréger les mots ou phrases sans espace entre eux mais en utilisant une majuscule au début de chaque mot. Le nom vient des bosses que forment les majuscules au sein d'un agrégat, qui rappellent celles d'un chameau.

```
voiciUnExempleDeCasseChameau  
laCasseChameauMinusculeEstIdentique  
VoiciUnExempleDeCasseChameauMajuscule
```

Il y a deux variantes de la casse chameau, qui se distinguent par la manière dont elles représentent l'initiale. La variante qui utilise une majuscule pour l'initiale est souvent appelée CasseChameauMajuscule ou casse Pascal. La variante qui utilise une minuscule pour l'initiale est souvent appelée casseChameauMinuscule. Par souci de clarté, nous utiliserons respectivement les termes CasseChameauMajuscule et casseChameauMinuscule dans ce document.

Source : <http://en.wikipedia.org/wiki/CamelCase>
(voir aussi l'article sur le site français : <http://fr.wikipedia.org/wiki/CamelCase>).

Fonction personnalisée publique : une fonction personnalisée publique est la première fonction appelée (ou fonction parente) au sein d'un groupe de fonctions personnalisées. Souvent, plusieurs fonctions personnalisées sont combinées pour fournir un éventail complet de fonctionnalités. Dans d'autres cas, une fonction personnalisée peut se suffire à elle-même. Dans un cas comme dans l'autre, la fonction appelée directement est considérée comme une fonction publique personnalisée.

Fonction personnalisée privée : une fonction personnalisée privée n'est jamais appelée directement dans un calcul. Elle est uniquement appelée par une autre fonction personnalisée publique. Les fonctions privées sont subordonnées aux fonctions publiques. Il est tout à fait possible d'avoir une fonction personnalisée qui soit toujours privée. Par exemple, il est possible de créer une fonction personnalisée conçue pour faire office de fonction assistante pour d'autres fonctions. En d'autres termes, elle ne sera utilisée qu'en combinaison avec une autre fonction personnalisée.

Occurrence de table primaire (PTO) : une occurrence de table primaire est une occurrence de table spéciale. L'occurrence de table primaire est l'occurrence de table désignée pour être utilisée lors de la création de calculs 'références en interne' ^a. Ces calculs sont dérivés de données contenues exclusivement dans le contexte de la table même, et non issues d'une autre table.

Table d'espacement : système permettant de créer des tables étiquettes, ou tables de séparation, au sein de la boîte de dialogue Définir la base de données. Cette technique utilise des tables sans aucune rubrique pour regrouper et classer les tables et les occurrences de tables.

Table : recueil de données appartenant à un sujet, par exemple des clients ou des prix unitaires. Un fichier de base de données contient une ou plusieurs tables, consistant en des rubriques et des enregistrements. Lors de la création d'une nouvelle table, une représentation visuelle (ou occurrence de table) de la table s'affiche dans le graphique des liens.

Vous pouvez indiquer plusieurs occurrences (avec des noms uniques) de la même table de façon à travailler avec des liens complexes dans le graphique.

Occurrence de table (TO) : une occurrence de table désigne une instance d'une table dans le graphique des liens. N'oubliez pas que toutes les interactions avec une table au sein de l'environnement de développement interagissent avec les occurrences de tables. C'est le seul moyen de communiquer avec une table et son contenu.

Table source : occurrence de table est associée à une table. La table source est la table à laquelle l'occurrence de table est associée. Par exemple, une occurrence de table nommée InterfaceAdmission_Classes_ListeClasses possède une table source nommée classes.

Identificateur de solution logique (LSI) : suffixe utilisé pour chaque fichier secondaire ; c'est également un suffixe facultatif utilisé pour chaque fichier primaire d'une solution.

Désignation de groupe FileMaker : préfixe utilisé pour identifier les groupes authentifiés en externe.



Annexe D ► Légende de la syntaxe

Le document « Conventions de développement FileMaker » utilise un style particulier pour indiquer les différents composants de la syntaxe utilisée dans les différentes sections. Le tableau suivant décrit les différentes notations.

Légende de la syntaxe du document « Conventions de développement FileMaker »

	L'absence de caractères de part et d'autre d'un élément indique que la méthode de dénomination est naturelle/libre
[]	Les éléments entre crochets sont destinés à être affichés
{ }	Les éléments entre accolades sont facultatifs et définis par le développeur
()	Les éléments entre parenthèses sont facultatifs mais ils ont une valeur définie ou induite s'ils sont utilisés
< >	Les éléments entre signes inférieur à et supérieur à sont censés être fournis mais ils ont une valeur définie ou induite
<< >>	Les éléments entre doubles signes inférieur à et supérieur à sont censés être fournis et ils sont définis par le développeur

