

STANDARDS ET NOMENCLATURE DE DÉVELOPPEMENTS

SECTION N°1

**RELATIONS, TABLES et GROUPE DE TABLES,
RUBRIQUES, LISTES DE VALEURS et MODELES**



Ugo Di Luca - DL SYSTEMS - Novembre 2005

Les nouvelles opportunités offertes par FileMaker 7 et la toute nouvelle version 8 nous ont amenés à revoir considérablement ce que nous avons progressivement construit depuis les premières versions de FileMaker.

Les modifications radicales du schéma relationnel et l'introduction des paramètres de scripts, dans un premier temps, puis dans un second la naissance d'outils de conception d'interfaces et l'adoption tant attendue des variables, constituent autant d'opportunités nouvelles de développement que de raisons d'effectuer de profonds remaniements, dans nos anciennes applications et dans notre approche des nouveaux projets.

La philosophie générale du logiciel reste la même, certes, mais développer avec FileMaker 7 et FileMaker 8 est à la fois plus souple, plus complexe et finalement plus simple.

Plus souple car la structure n'est plus téléguidée et qu'elle sera désormais le reflet des besoins et non des limitations de la plate-forme de développement.

Plus complexe, car ces choix de structures nécessitent une approche plus fine qu'auparavant, mais aussi parce que l'intégration des nouvelles fonctionnalités ne se fera pas sans un minimum de programmation, voire sans un minimum d'expérimentations.

Plus simple enfin car une fois le cœur du développement défini, et certains principes adoptés, la complexité laissera place à l'optimisation et vous aurez alors dans les mains un produit souple, complexe, simple et solide.

FileMaker n'est pas nécessairement devenu un outil réservé aux programmeurs chevronnés. Mais sans l'adoption de standards de développement et de nomenclatures spécifiques, il est illusoire de vouloir tirer un véritable profit de ces multiples évolutions, et nul doute des prochaines également.

Traditionnellement, adopter des nomenclatures facilite le travail de développement et la lecture, voire la relecture de vos réalisations. Au sein d'équipes de travail, mais aussi dans les cas d'échanges avec d'autres développeurs, ces standards servent de guide de lecture.

Compte tenu des nouvelles fonctions introduites par FileMaker 7, ces standards s'inscrivent aujourd'hui dans une stratégie globale d'optimisation de vos développements. Car ces nomenclatures vous permettront de manipuler simplement rubriques, liens, listes de valeurs ou modèles, au travers des scripts en faisant usage des fonctions d'extraction de texte pour en déterminer le contenu.

Vous trouverez quelques exemples illustrant cela au sein de ce document.

Au terme de plusieurs développements réalisés avec FileMaker 7, et de quelques mois à digérer les nouveautés de FileMaker 8, il m'a semblé utile de rassembler quelques idées et de chercher à bâtir une charte de nomenclature pour mes prochains projets.

Ce document n'est donc qu'un recueil d'idées. Il est probablement trop lié à mon approche de FileMaker, à mes méthodes de travaux, et aux types de solutions que je développe au quotidien.

J'espère qu'il sera au moins source d'inspirations.

1] LES TABLES, OCCURRENCES ET GROUPES D'OCCURRENCES :

A/ GENERALITES :

Il s'agit là bien entendu de la principale évolution de FileMaker 7. Bien des ouvrages et discussions ont apporté leur éclairage sur le sujet, donc je ne me contenterai ici que de quelques rappels ou compléments d'ordre général.

Une occurrence de table (OT) n'a pas d'apparence "physique", elle n'est qu'un **alias** de la Table mère permettant de distribuer l'information là où cela s'avère utile.

Lorsque votre projet évoluera, les 20 tables de votre solution deviendront sans nul doute une bonne centaine d'occurrences de tables, au moins.

Les occurrences de Tables peuvent être **groupées** sur ce qu'on appelle désormais communément le "Point de Vue". Nul besoin par exemple de disposer d'un lien vers les Salles de Classes au moment de l'inscription d'un élève.

Ces Groupes d'Occurrences (GOT) éclatent le graphe en plusieurs morceaux permettant ainsi de mieux visualiser les relations et/ou de limiter l'accès à certaines données.

Un Modèle est attaché à une OT, mais toutes les OT n'ont pas nécessairement un modèle attaché.

Une occurrence peut en effet n'être utile que pour **propager** les liens.

Si un calcul ne s'évalue qu'en fonction d'un **contexte**, il peut très bien exploiter des données évaluées au-delà du contexte référencé.

Cela signifie que des calculs doivent pouvoir s'évaluer librement indépendamment du GOT "actif".

Certaines occurrences pourront donc n'avoir que des vertues **utilitaires**, permettant l'évaluation des calculs ou des références externes au-delà du GOT.

Les GOT ne sont généralement pas liés entre eux, mais toutes les OT d'un même groupe ne le sont pas non plus.

S'il est possible de visualiser dans le graphe chacune des OT et leur donner des signes distinctifs par la couleur, elles apparaîtront nécessairement **triées par ordre alphabétique** dans les dialogues de FileMaker.

Les Tables d'Occurrences seront listées :

- Filtrées selon le contexte dans les boîtes de dialogue des calculs, détachées si elles ne sont pas liées à l'OT au départ de laquelle le calcul s'évalue.
Les données d'une OT non liée ne sont généralement pas accessibles, à l'exception des **rubriques globales** qui peuvent être transférées et exploitées au-delà de tout contexte, sans même nécessité d'établir de liens.
- Non filtrée dans scriptMaker.

B/ CATEGORISATION & PROPOSITIONS DE NOMENCLATURE :

Les constatations effectuées précédemment ont bien entendu une influence considérable sur le développement d'un applicatif FileMaker. Comment distinguer les occurrences lorsque plus d'une centaine cohabitent ensemble et comment surtout identifier les bonnes ?

Comme nous l'avons vu, une OT sert à tout autre chose qu'à stocker des données et on la retrouvera mêlée à d'autres OT dans le graphe à plusieurs reprises.

Un nommage du type 'Elèves 1', 'Elèves 2' ne pourra pas être suffisant, vous le comprendrez bien, même si par défaut, c'est le système proposé par FileMaker lors de la "duplication" d'une OT.

Un premier tri pourrait être effectué en exploitant le GOT auquel l'OT est "attachée" en utilisant un **prefixe**.

Par exemple, toutes les occurrences appartenant au groupe de table 'Gestion du Catalogue Produit' utiliseront le préfixe "GP" et celles groupées sous 'Gestion des Clients' utiliseront le préfixe "GC".

Ainsi, lors de l'affichage de nos Occurrences par le biais des listes de valeurs, les occurrences seront triées **sur la base du Point de Vue**, comme dans l'illustration N°1.

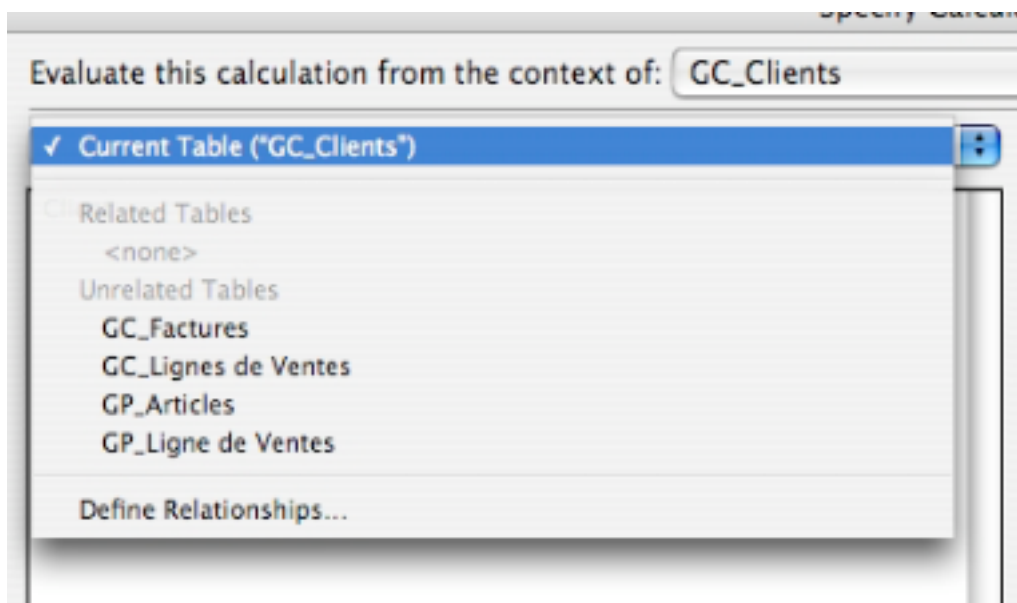


Illustration N°1

Cela ne suffit cependant pas car dans certain cas, notamment lorsque vous utiliserez des tables de jonction ou des filtres, il vous sera difficile de distinguer instantanément la table source.

C'est pourquoi, nous pourrions ajouter un autre préfixe (de 2 ou 3 caractères) pour la Table au delà du GOT, que nous transformerons en minuscule pour une meilleure vue, par exemple:

gc_CL_Clients
gc_LV_Filtre Factures
gc_FA_Factures

FM8 permet d'effectuer certains **commentaires** dans le Graphe Relationnel. Ceci est fort utile, mais ces informations, pour l'heure ne sont pas exploitables au-delà du graphe lui-même et n'offrent donc pas d'utilité au-delà d'une clarification ponctuelle.

Vous pourriez utiliser ces commentaires pour définir le **titre du Groupe** d'Occurrences. En élargissant la boîte de ce commentaire, vous pourrez également encadrer vos groupes si besoin.

Lorsque votre solution sera exposée en de multiples GOT, vous aurez malgré tout nécessairement besoin d'identifier un groupe et un autre.

Vous pourriez exploiter une Table 'TitresGroupes' pour cela, vous permettant alors d'obtenir par exemple le type de liste présentée dans l'illustration N°2.

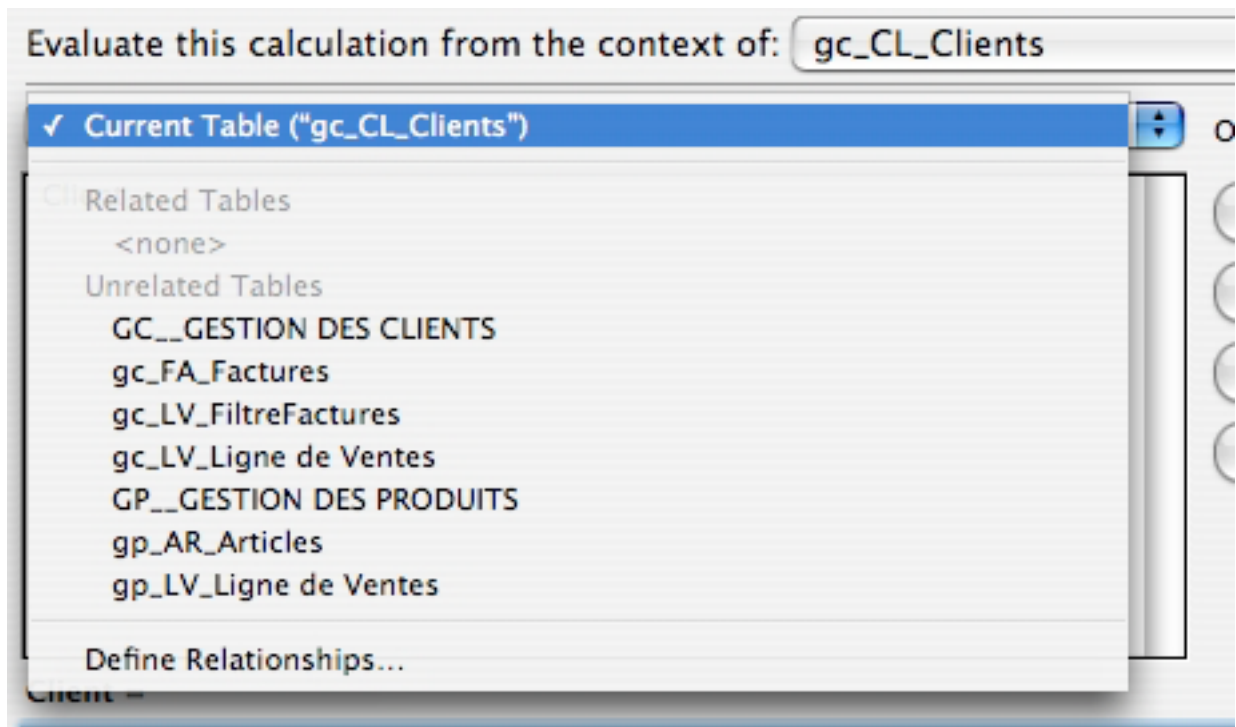


Illustration N°2

(notez le double “_” pour que le titre se situe toujours en tête de liste)

Désormais, vous visualiserez déjà mieux vos occurrences et pourrez naviguer de l'une à l'autre en exploitant le clavier également en entrant “gc” ou “gp”...

Sur la base des constatations effectuées précédemment, nous pourrions catégoriser les Occurrences en :

- “Distributrices” (pour propager les liens),
- “Utilitaires” (utilisée en dehors de tout GOT)
- “Visuelles” (pour la visualisation de données).

Les **OT utilitaires** seront de préférence regroupées dans un ou plusieurs groupes distincts.

Si elles seront ainsi facilement localisables, vous constaterez qu'une fois défini le contenu de vos tables, ce sont ces ressources utilitaires que vous exploiterez le plus au cours du développement.

Afin de les avoir à portée de mains quelque soit le nombre d'occurrences, et sans avoir à utiliser le clavier, vous pourriez les faire glisser en utilisant un autre “_” comme préfixe, qui les déplacera en tête de liste des tables non liées.

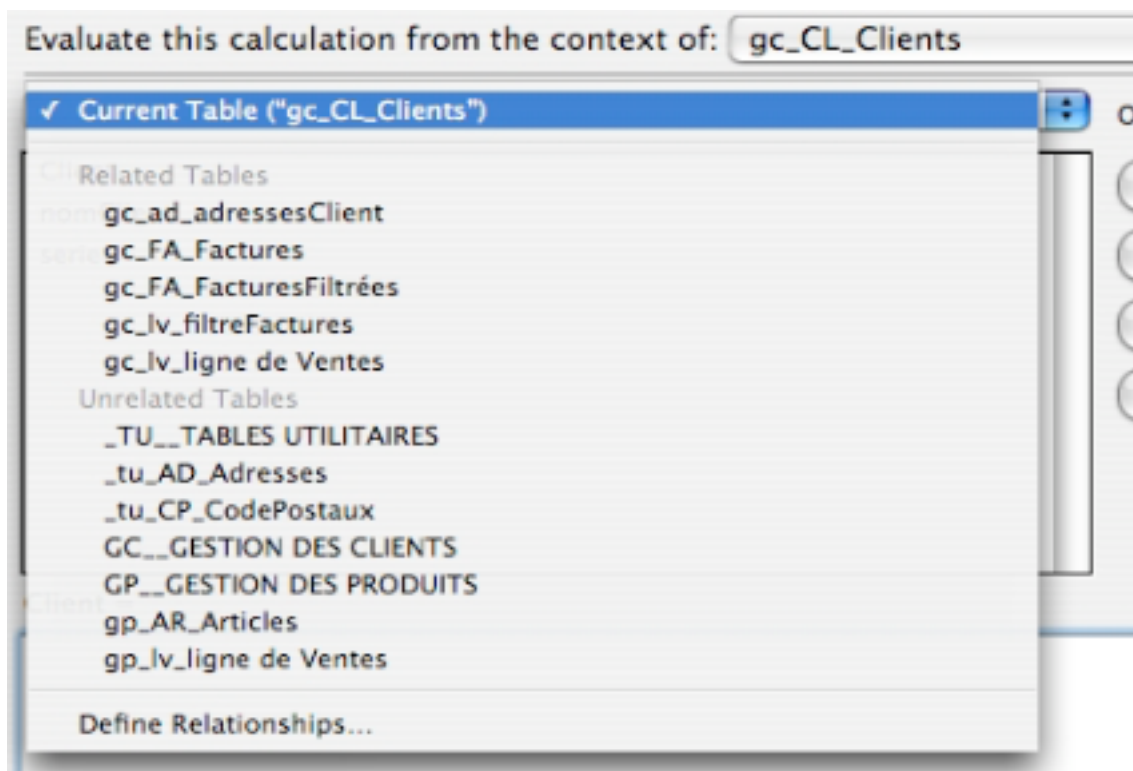


Illustration N°3

Les **OT “visuelles”** (celles attachées à un modèle) doivent également être dissociées de celles “distributrices”, notamment dans les scripts, où pour rappel, toutes les occurrences de tables seront listées. Pour ce faire, les OT distributrices seront en minuscules (préfixes et premier caractère du nom), comme ci-dessus.

Même si FM8 en a réduit leur utilité encore davantage, les rubriques Globales servant de réservoirs de données ou de marqueurs, pourraient être placées dans une Table distincte, nommée pour 'Variables Globales'.

Là encore, à moins que vous exploitiez à 100% le pouvoir des nouvelles variables et délaissiez leurs ancêtres, vous constaterez qu'il vous faut souvent accéder aux données de cette table.

Nous la ferons glisser en toute tête de liste en lui greffant 2 underscores comme préfixe, et exceptionnellement, cette occurrence étant **flottante**, nous la dissociérons de tout groupe. Chaque rubrique étant associée à l'occurrence de la table de provenance, le double underscore vous permettra de mieux localiser une globale dans un calcul.

Il est possible de créer d'autres occurrences de cette table 'Variables Globales'. Mais les passages des globales par script ou par calculs devraient n'utiliser que cette occurrence de table.

Enfin, si les occurrences sont des alias, il n'en reste pas moins que l'**original** contient des données et qu'il est bon de disposer d'occurrences spécifiques lors de certaines procédures, à commencer par les imports.

Toutes les tables de votre solution pourraient alors être listées, cette fois en fin de liste en utilisant un préfixe "z" comme dans l'illustration N°4.

Vous constaterez que les noms d'occurrences ont été modifiées, en premier lieu en portant au pluriel les occurrences provenant de Tables de Fusion, et aussi et surtout en incorporant un underscore supplémentaire. D'une façon générale, seules les 26 lettres de l'alphabet, le caractère underscore (" _ ") et les nombres de 0 à 9 sont autorisés. Ni accent ni espace à ce stade.

Nous le verrons dans la section 2, mais voilà bien un moyen de **dynamiser vos scripts** en utilisant une partie du nom de l'occurrence pour par exemple naviguer d'une OT à une autre ou pour définir dynamiquement une valeur en fonction d'une extraction, en exploitant les fonctions GetField () et Evaluate () notamment.

Voilà, pour être précis, à quoi ressemblerait votre liste lorsque vous l'exploitez dans un calcul lié à un script.

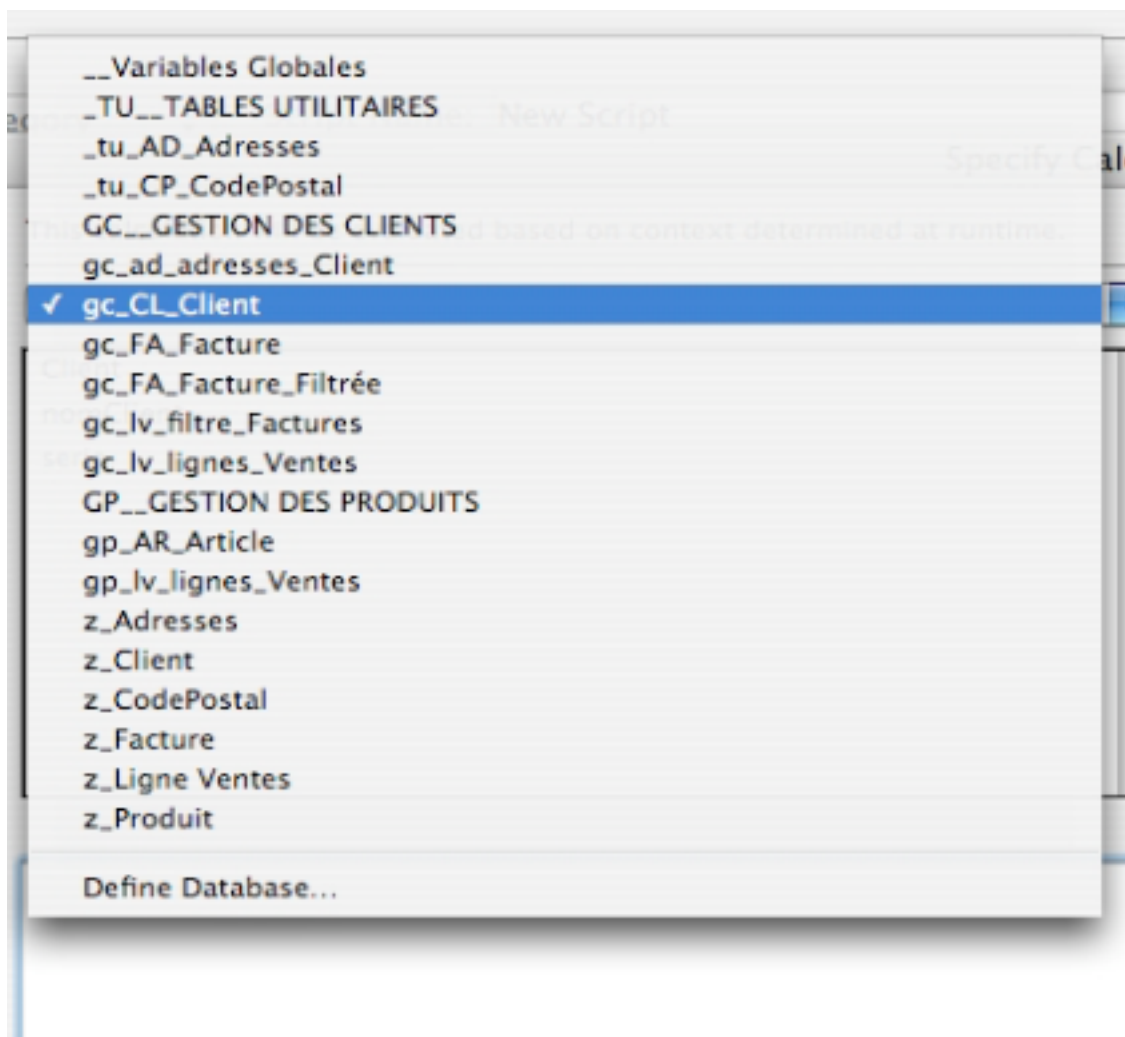


Illustration N°4

2] LES RUBRIQUES ET CLEFS :

A/ GENERALITES :

La révolution engagée au niveau de l'architecture relationnelle et de l'interface n'a pas encore eu d'effet sur les rubriques, qui n'ont pas subi de profondes transformations.

Seules certaines **nouvelles options** viendront ça et là troubler le développeur dans l'intégration de ces rubriques.

Une rubrique appartient certes physiquement à une table donnée, mais son exploitation dans un modèle dépendra là encore du **contexte**.

FileMaker nous offre cependant un moyen d'extraire le nom de la rubrique seul, ou son nom et l'occurrence de table à laquelle celle-ci est attachée, grâce respectivement aux fonctions Get(ActiveFieldName) ou Get(ActiveTableName), de sorte qu'il est nullement nécessaire d'attribuer un préfixe référant le nom de la Table.

A eux seuls, les préfixes des GOT et de OT tels que définis précédemment seront un plus appréciable dans la manipulation des rubriques au sein de scripts.

B/ CATEGORISATION :

Les rubriques peuvent être néanmoins catégorisées selon leurs fonctions, il existe globalement 6 types de rubriques :

- Clef :

Utilisées pour les liens d'une occurrence de table vers une autre, le contenu de ces rubriques est généralement "**cachée**" à l'utilisateur.

- Développeur :

Comme leur nom l'indique, ces rubriques serviront à la manipulation en **arrière plan** de la base de données et ne sont d'aucune utilité pour l'utilisateur final.

- Interface :

Ces rubriques sont des **indicateurs**, généralement graphiques sans incidence réelle sur les données.

- Utilisateur :

Celles réservées à l'utilisateur et exploitables par ses propres soins dans les dialogues, les imports ou exports de données

- Contrôle :

Celles mises à disposition de l'utilisateur, non modifiables autrement que par script, lui offrant des **points de repères** alternatifs sur les données. Il pourrait s'agir des numéros de série, de certains décomptes.

- Variables :

Ces rubriques sont mises à disposition de l'utilisateur et du développeur, mais leur contenu est tel qu'elles ne sont pas exploitables directement par l'utilisateur. Il s'agit principalement des **globales**, ou de toute rubrique contenant des données variables.

Les rubriques de type 'Utilisateur' et 'Contrôle', devant pouvoir être exploitées par l'utilisateur lui même, il conviendra donc de leur trouver des dénominations facilement **compréhensibles**.

L'idéal serait que ces noms de rubriques puissent être également utilisés comme **labels** et je ne vois donc pas de contre indication à l'utilisation d'espaces dans leur définition. Les accents et autres caractères spéciaux seront cependant à éviter.

Il convient ici de préciser qu'un numéro de facture n'est pas à mes yeux considéré comme une clef en soi. Ce numéro de facture sera intégré aux rubriques de Contrôle, et on lui préférera une autre clef unique pour lier une facture à tout autre type de données.

Toutes les autres rubriques nécessitent à mon avis une **identification parfaite**, à commencer par les clefs.

Les rubriques Utilisateurs et Contrôle doivent nécessairement apparaître en tête de liste. De cette manière, il leur sera facile de les identifier pour toutes les opérations où son intervention est nécessaire.

Afin d'isoler ces rubriques, nous utiliserons un préfixe commençant par un "z" pour toutes les autres, et nous trierons les listes de rubriques par ordre alphabétique. Comme pour les GOT, il convient de disposer d'un titre ou d'un **séparateur formel** entre ces deux groupes de rubriques.

Nous utiliserons 2 rubriques globales de type Date pour l'occasion, respectivement nommée "_ Rubriques Utilisateurs_" et "z_**Rubriques Développeurs**_", de sorte que chacun des titres apparaissent en tête de leur groupe respectif.

C/ PROPOSITION DE NOMENCLATURE :

Au delà de ces premières petites conventions, voici les préfixes que j'utilise désormais dans mes développements :

- Développeur : zd zd_monNomDeRubrique
- Interface : zi zi_monNomDeRubrique
- Variable : zv zv_monNomDeRubrique
- Clef : zcx zcx_maClef

Comme vous pouvez le constater, les clefs comportent 3 caractères, le x étant substitué par un autre en fonction du type de clef :

Clefs indexables :

- Clef Primaire : zcp_maClefPrimaire

Il s'agit de la clef principale dans chaque Table. Nécessairement, elle comportera l'intitulé "id" suivi du nom de la Table.

ex : zcp_idClient, zcp_idLignesVente, zcp_idFacture, etc.

- Clef Secondaire : zcs_maClefSecondaire

Il s'agit de la clef secondaire (Foreign Key) elle aussi comportant nécessairement l'intitulé "id" suivi du nom de la Table Source dans ce cas.

ex : zcs_idClient dans la Table Facture, zcs_idArticle dans la table LignesdeVentes.

- Clef Alternative : zca_maClefAlternative

Il peut s'agir d'une clef calculée en fonction d'une catégorie par exemple, ou toute autre clef permettant d'atteindre un enregistrement par un autre moyen que par son identifiant.

ex : zca_categorie ou zca_pays

- Clef Multilignes : zkm_maClefMultiLignes

Une clef multilignes offre la possibilité de stocker plusieurs valeurs, et ainsi de permettre à l'enregistrement d'être accessible de façon plus large, sous la forme d'un lien de type "OU"

ex : zkm_typeClient comprenant "Plombier ¶ Peintre"

- Clef Concaténation : zkc_maClefConcatenation

Une clef de concaténation permet à l'inverse d'une clef multiligne de réduire le champ d'action du lien sous la forme d'un lien de type "AND" indépendamment des possibilités offertes par les liens multiples dans le Graphe lui-même

ex : zkc_langues comprenant "Anglais - Italien"

- Clef Multiple : zkk_maClefMultiple

Une clef multiple est une combinaison entre la clef Multilignes et la Clef de Concaténation. Il s'agit très souvent d'une clef utilisée pour accomplir un filtrage conditionnel.

ex : zkk_langues comprenant "Anglais¶Italien¶Anglais - Italien"

Clefs non mémorisables :

- Clef Filtre : zkf_maClefFiltre

Il peut s'agir d'une clef indexable également, ou constante comportant par exemple une valeur fixe (ex: "Anglais") ou d'une valeur globale permettant un filtre dynamique du lien.

ex : zkf_langues comprenant "Anglais" de manière fixe ou
zkf__langues exploitant une globale (marquée par le double "_")

- ClefCréation : zkn_maClefCreation

Clef à part, cette dernière permet si elle est utilisée à “gauche” d’un lien autorisant la création d’enregistrements liés, de créer dynamiquement de nouveaux enregistrements et d’en récupérer l’identifiant. La clef magique.

ex : zkn_clefMagique

D/ DISTINCTIONS ADDITIONNELLES :

FM7 a introduit une fonctionnalité fort intéressante. Une rubrique auto-entrée peut désormais se comporter comme un calcul, se réévaluant automatiquement.

Dès lors, distinguer un calcul d’une rubrique auto-entrée ou d’une rubrique dont la valeur est établie par script devient à la fois difficile et peu explicite.

Pour offrir un moyen de distinction malgré tout, un calcul aura une syntaxe quelque peu différente d’une rubrique auto-entrée, en cherchant toujours à le rendre le plus explicite possible.

ex : xTotalVentesClient sera un calcul et totalVentesClient éventuellement une rubrique nombre (auto-entrée ou pas).

Les rubriques statistiques elles aussi devront pouvoir être distinguées des rubriques calculs.

ex : TOTAL_VentesClient présentera un moyen de distinguer la rubrique Statistique de xTotalVentesClient.

E/ LES VARIABLES : CAS PARTICULIER :

Les variables, introduites avec la version FM8 constituent un cas à part à plus d'un titre.

- Il ne s'agit pas de rubrique
- elles peuvent être locales ou globales
- elles peuvent être incorporées dans un calcul ou rester localisées dans un script

Sans les avoir parcourues en entier, il est fort improbable que l'utilisateur se voit donner accès à un calcul exploitant une variable.

Malgré tout, pour les distinguer, un calcul exploitant des variables sera marqué d'un autre préfixe, "v_"

ex : zi_v_xOngletsRep sera un calcul répétitif permettant d'identifier l'onglet actif au travers d'une variable.

Dans la mesure où ces variables n'ont aucune présence physique, et que le DDR lui-même ne permet pas de les matérialiser, il nous faudra les **référencer** dans une Table à part, à l'usage du développeur.

A raison d'une variable par enregistrement avec un **commentaire** précis sur son usage et si possible les scripts pour lesquels cette variable est utilisée, vous aurez ainsi constitué votre propre documentation.

Encore une fois, afin de bien distinguer la variable dans un calcul, et de pouvoir si besoin l'appeler dynamiquement, il peut s'avérer intéressant de lui ajouter un underscore sous la forme \$_variableLocale et \$\$_variableGlobale.

Enfin, es variables par nature comportent un nombre de répétitions non fixé par le développeur.

Ainsi convient-il de marquer les variables qui véritablement exploiteront des valeurs répétitives.

\$_R_variableLocale et \$\$_R_variableGlobale est une suggestion de départ.

F/ SACHEZ OPTIMISER :

Comme je le précisais dans l'introduction, ces standards peuvent être utiles au-delà de la phase de développement.

Toutes ces occurrences, rubriques et listes ne sont que des chaînes de texte. S'il est encore impossible de décider dynamiquement du nom de la rubrique à définir, il est en revanche fort simple au contraire de définir le contenu d'une rubrique en en référant une autre.

Prenons par exemple un script de navigation de fiches à fiches, pour laquelle il vous est nécessaire de définir à la volée le contenu d'une globale en y plaçant le contenu de la clef primaire de l'enregistrement en cours.

```
Set Field [MaGlobale;
Evaluate (
Middle (
Get ( LayoutTableName ) ; Position( Get ( LayoutTableName ); "_";2;1)+1;3)
&
"zkp_id" & $$_TableSourceActive)
```

Dans la cas présent, nous avons :

1) extrait du nom de l'occurrence à laquelle le modèle actuel est rattaché (exemple gc_CL_Client) le préfixe de la Table source accompagné de l'underscore, c'est à dire "CL_"

2) nous lui avons ajouté notre préfixe de clef primaire, soit "zkp_id"

3) enfin, nous avons ajouté le nom de la Table active par le biais d'une variable qui est elle mise à jour à chaque changement de Point de Vue.

pour obtenir par exemple "CL_zkp_idClient"

Disposer d'une charte unique qui vous est propre, figée et solide vous permet de multiples interactions entre les scripts et les rubriques. Une dose d'automatisation des tâches qui vous simplifie la vie lorsqu'un seul script suffit là où précédemment une bonne douzaine auraient été nécessaires, ou une bonne douzaine de If et Else If.

Les pouvoirs de la version 8 sont désormais quasi illimités si on considère que la fonction GetNthRecord () permet de récupérer une valeur donnée, dans un champ donné, d'une table donnée, et que la syntaxe peut être paramétrée au sein du script lui-même.

3] LES LISTES DE VALEURS :

A/ GENERALITES :

Une liste de valeur, une fois définie, pourra être exploitée sous de multiples formes, qu'il s'agisse de cases à cocher, d'une liste pop-up ou autre menus.

Il ne convient donc pas d'inclure dans le nom de la liste la forme qu'elle prendra.

Les valeurs peuvent provenir de différentes sources. Il peut s'agir de valeurs personnalisées, de valeurs liées ou tout simplement du contenu d'une valeur provenant d'un autre fichier.

Il conviendra donc de les différencier car leur comportement sera différent, et d'utiliser une structure qui permettra de manipuler ces listes via script si besoin.

Car il est possible ici aussi de définir le contenu de la fonction ValueListItems () de manière dynamique, et la faire ainsi varier à votre convenance.

B/ DENOMINATIONS :

Je conserve dans ses grandes lignes les propositions faites par CoreSolutions dans un document antérieur destiné aux versions 6 et antérieures.

Les préfixes suivant seront utilisés :

- Les listes de valeurs personnalisées : "pr_"
- Les listes de valeurs exploitant le contenu d'une rubrique : "cr_"
- Les listes de valeurs utilisant une liste de valeur d'un autre fichier : "vf_"
- Les listes de valeurs exploitant le contenu d'une rubrique liée : "rl_"

Seuls les 26 lettres de l'alphabet, l'underscore et le signe "#" sont autorisés, le signe "#" étant utilisé pour indiquer la Table de référence.

Si la Table de référence se situe dans un fichier externe, alors on ajoutera un autre underscore après le "#"

ex : rl_idContacstLiés_gc_CL_Clients#gc_EM_Employes

Dans l'exemple ci dessus, nous chercherons les identifiants des contacts de l'occurrence gc_CL_Employes au départ de l'OT gc_CL_Clients

Si elle n'est probablement pas esthétique, cette dénomination offre une vue immédiate sur la source et la destination des listes liées.

4] LES MODELES :

A/ GENERALITES :

Vous en avez peut-être vous même fait l'expérience. Le nombre de modèles nécessaires est très fortement lié à la complexité du graphe relationnel. Puisqu'un modèle est nécessairement attaché à une occurrence de Tables, et puisqu'il est d'usage de multiplier ces dernières, vos choix de structure vous orienteront peut-être vers cette démultiplications des modèles.

Même si ce n'est pas la vocation de ce document, posez-vous malgré tout des questions sur le bien fondé de vos choix initiaux si le nombre de modèles devient excessif, bien entendu en écartant les modèles dédiés aux rapports

Peut-être votre programmation pourrait être simplifiée, par exemple en évitant de systématiquement vous déplacer vers un modèle dédié d'une OT pour récupérer les données d'un enregistrement lié, voire en visualiser le contenu.

FM8 vous propose désormais GetNthRecord, et en outre, de nouveaux schémas relationnels "standardisés" ont été introduits, permettant par exemple de n'utiliser qu'une seule et unique occurrence pour chaque Table.

Finalement, l'introduction des onglets vous permettra de décomposer vos données au sein d'un même modèle, voire de disposer de n modèles différents tous cachés les uns derrière les autres, en utilisant un panneau de tabulation élargi à la taille du modèle.

Quel que soit le nombre des modèles de votre applications, sans avoir défini des dénominations précises là encore, vous ne pourrez pas profiter des fonctionnalités de FileMaker 7 et FM8, et vous le ressentirez automatiquement dans la conception de votre navigation.

Il est assez difficile de définir des règles, car chaque application comporte ces propres logiques. Il est néanmoins possible de catégoriser les modèles selon la fonction qu'ils remplissent et leur apparence.

B/ CATEGORISATION / TYPE DE MODELES :

Dès lors qu'un modèle contient différentes rubriques, on pourrait déterminer que la fonction principale d'un modèle des de visualiser les données. C'est vrai bien entendu.

Cependant, un modèle est également dans certains cas une passerelle vers d'autres données, un lieu de passage, où seront effectuées certaines opérations sans que l'utilisateur ne s'en rende compte. Et bien sûr des impressions.

Chaque modèle couvre en définitive sa fonction spécifique, qu'il s'agisse de visualiser des données sous la forme de liste, de disposer d'une vue détaillée d'un enregistrement ou au contraire d'une vue synthétique.

Voici une liste non limitée des fonctionnalités pouvant être utilisées et pour lesquelles des modèles dédiés sont nécessaires.

- Préparation d'un nouvel Enregistrement -> Nouveau

Nous utiliserons un modèle spécifique, le plus généralement exploitant des globales qui seront envoyées vers le nouvel enregistrement. Ces modèles sont souvent exploités au sein d'une fenêtre distincte flottante.

- Modification d'un Enregistrement -> Modification

Il est assez fréquent de diriger l'utilisateur vers un modèle spécifique pour effectuer des modifications de certains enregistrements

- Visualisation d'un enregistrement en Détail -> Détail

C'est le type de modèle le plus utilisé. L'arrivée des onglets apporte dans leur cas un confort indéniable et un moyen immédiat de réduire le nombre de modèles.

- Visualisation d'un ensemble trouvé sous forme de liste. -> Liste

Une vue en Liste....à vous de choisir les rubriques à y inclure

- Visualisation d'un ensemble trouvé sous forme de tableau. -> Tableau

Une vue en Tableau....à vous de choisir les rubriques à y inclure

- Rapport de données groupées -> Rapport

Une vue en Liste nécessairement triée et visualisable en mode Prévisualisation ou à imprimer simplement après tri.

- Recherche de Données -> Recherche

Si une recherche peut être effectuée sur n'importe quel modèle, vous pouvez également rediriger l'utilisateur vers un modèle où seules les rubriques utiles seront présentes, et non modifiables en mode Utilisation.

- Visualisation d'un enregistrement de manière Synthétique -> Focus

De manière générale, ce type de Modèle sera utilisé au sein d'une fenêtre flottante, pour donner une information spécifique, ou un groupe d'information. Les onglets ici encore peuvent aussi permettre d'atteindre cet objectif

- Visualisation d'une liste d'enregistrement dans une Table Externe -> Portal

Outre un visuel, ce type de modèle, souvent exploités au sein d'une fenêtre flottante peuvent aussi être utilisés pour effectuer des sélections par simple clic.

- Utilitaires : Redirection / Boucles /Copier -> Go / Boucle/ Copy

Ces modèles utilitaires sont devenus indispensables avec FM7. Ils couvrent diverses fonctionnalités qui doivent nécessairement être exécutées au sein d'un modèle. A noter que celui-ci pourra être totalement vierge.

....à vous de choisir les rubriques à y inclure

C/ PROPOSITION DE NOMENCLATURE :

Comme pour les points précédents, chaque modèle doit comporter un nom explicite mais répondant également à une signalétique vous permettant d'identifier de manière simple chaque modèle, et de pouvoir programmer certains scripts en fonction de la signalétique.

1. Le nom du modèle doit permettre d'identifier au moins la table Source à laquelle il est lié. Ce nom, sous la forme d'un préfixe en minuscule, doit correspondre à celui utilisé dans la définition des OT, et sera placé comme premier "paramètre".

2. Les préfixes suivant pourront être utilisés comme second paramètre :

DET = Visualisation d'un enregistrement en Détail

LST = Visualisation d'un enregistrement en Liste

TBL = Visualisation d'un enregistrement en Tableau

FCS = Visualisation synthétique (Focus) d'un enregistrement

SEL = Selection d'enregistrement dans une Table Externe

POR = Visualisation d'enregistrement dans une Table Externe

RCH = Modèle dédié aux recherches

CRE = Modèle dédié au processus de Création

MOD = Modèle dédié au processus de Modification

RAP = Rapport de données

UTL = Utilitaires

(Lorsque le modèle sera utilisé au sein d'une fenêtre, le nom du modèle commencera par un "w".)

3. Une description sommaire et compréhensible du modèle suivra ensuite, éventuellement complétée par un autre moyen de distinction

4. Le nom du modèle pourra également comporter un véritable paramètre, exploitable par le script. Il suivra le signe "#". et conclura le nom du modèle.

Voici donc quelques exemples :

lv_RAP_RapportDesVentes_Annuelles

Rapport des Ventes exploitant les enregistrements de Lignes de Ventes regroupées par année.

lv_RAP_RapportDesVentes_Mensuelles

Rapport des Ventes exploitant les enregistrements de Lignes de Ventes regroupées par mois.

lv_RAP_Facture#pdf

Facture destinée à être envoyée en pdf au terme de l'impression en exploitant le paramètre "pdf"

cl_wFCS_Adresses #verrou

Ce modèle affichera dans une fenêtre flottante les adresses liées à un client. La fenêtre sera bloquée par une boucle avec une pause.

Un script de navigation pourra alors sans peine se diriger vers un modèle au travers de paramètres de scripts constituant autant de clefs nécessaires au déchiffrement du nom.

Basculer d'un modèle Détail (cl_DET_Client) vers un modèle sous forme de liste (cl_LST_Client) s'effectuera par la simple substitution de 'DET' par 'LST' pour fournir un exemple assez simple.