Cahier Technique

Les scripts d'édition

Sage Petites Entreprises
Documentation technique

Sommaire

I. Introduction

1. Qu'est ce qu'un script d'édition

Le script d'édition est un langage de programmation assurant le traitement des données en vue de leur édition. Il régit les données disponibles, les classements réalisés, l'ordre et les conditions des données imprimées sous forme de "bloc d'impression".

Pour accéder au script d'édition du paramétrage, rendez-vous dans le menu "Utilitaires / Paramétrages des éditions", sélectionnez le paramétrage que vous souhaitez modifier puis cliquez sur l'icône  .

Le langage décrit dans ce cahier technique est utilisé pour la rédaction des scripts d'édition ainsi que pour l'élaboration des formules présentes dans les paramétrages (bouton de formule de calcul ).

Chaque script ou formule est constitué d'un nombre variable de lignes contenant chacune des instructions décrivant des tâches élémentaires que le logiciel doit exécuter, sur diverses données spécifiées.

Important :L'écriture d'un script fait appel à des compétences de programmation avancées. Toute modification non souhaitée d'un script peut entraîner des dysfonctionnements lors des éditions ou dans le déroulement du programme (calcul).

2. Principe de conception d'un script d'édition

L'élaboration d'un script d'édition s'apparentant à de la programmation, les étapes antérieures à l'écriture proprement dite du script sont primordiales.

a) Définition des besoins

Ainsi, établissez au mieux une définition précise des besoins car une fois l'écriture commencée, il est difficile de revenir sur un classement ou une rupture oubliée.

Cette étape doit répondre aux questions élémentaires du type :

- Que doit faire cette édition ?
Ex : "Imprimer la liste des devis client"

- Dans quel ordre les données seront imprimées ?
Ex : "Classée par secteur géographique par client"

- Quels regroupements et cumuls je souhaite faire apparaître ?
Ex : "avec un cumul HT par secteur géographique et en fin d'édition"

- Quels filtres pourront être appliqués ?
Ex: "pour tous les secteurs géographiques ou un seul secteur"

Soyez sûr d'avoir répondu à toutes ces questions avant de procéder à l'étape suivante : l'écriture "en clair" ou l'algorithme.

b) Décomposition des besoins en étape "globale"

Maintenant les besoins bien définis, il vous faut traduire ceux-ci dans un langage courant qui vous est familier, simplement sur une feuille de papier.

Reprenons l'exemple "Imprimer la liste des devis client classée par secteur géographique par client" et analysons qu'elles étapes sont nécessaires.

- Constituer la liste des clients
- Classer les clients par secteur géographique
- Pour chaque client, lister les devis de celui-ci

c) Décomposition des étapes en séquence d'impression

Il est maintenant nécessaire de rapprocher les étapes "Globales" à la mise en page que vous souhaitez.

Rajoutons que pour des questions de présentation, vous souhaitez imprimer le détail du secteur géographique lors du changement de celui-ci (les clients étant classés par secteur géographique).

En écrivant de manière hiérarchique, en réalisant des décalages horizontaux de la même manière que les chapitres d'un texte, vous pourriez obtenir un schéma similaire à celui-ci-dessous :

- Constituer une liste des clients, la classer par secteur géographique

- Pour chaque client de la liste

  - Si le secteur géographique du client a changé

    - Imprimer les informations du secteur géographique

  - Imprimer le détail des informations du client

    - Lister les devis de celui

  - Pour chaque Devis

    - Imprimer les informations voulues

N'hésitez pas à détailler des étapes du schéma si vous ne parvenez pas à exprimer les conditions d'impression.

Vous êtes maintenant en possession d'un schéma clair, répondant à la totalité des besoins de mise en page et de classement formulés.

II. Théorie du langage des scripts

Ce chapitre détaille les instructions disponibles dans le langage des scripts d'édition dans le but de convertir le schéma ou les besoins exprimés dans le premier chapitre, en un langage compréhensible par le programme.

1. La mise en page : Les blocs d'impression

a) La définition d'une page : L'instruction "PageDef"

Une page est généralement constituée d'un bloc Haut de page en début de page incluant le titre, la date de l'édition, etc., et d'un bloc Bas de page en fin de chaque page contenant le décompte des pages, les informations société, etc.

La syntaxe suivante permet de préciser le nom et la position de ces blocs à placer systématiquement :

PageDef "NomBlocHautPage…NomBlocBasPage"

Cette syntaxe indique que *NomBlocHautPage* devra être imprimé en haut de toutes les pages et *NomBlocBasPage* en bas de chacune d'elles. L'espace restant entre les deux sera utilisé par les autres blocs indiqués dans le script par les commandes *Imprimer "NomBloc"*.

L'usage des points de suspension n'est nécessaire que pour **séparer** le bloc de Haut de page du bloc de Bas de page.

Vous pouvez utiliser une instruction de type PageDef "NomBloc" si vous souhaitez uniquement un habillage avec un Haut de page.

De même, la syntaxe PageDef "…NomBloc" permet de ne prévoir qu'un bas de page.

b) Les modes d'impression des blocs : L'instruction "Imprimer"

Impression d'un bloc

Imprime le bloc *NomBloc* dans la page courante.

Imprimer "NomBloc"

La page ne sera réellement imprimée qu'une fois remplie ou une fois le script terminé.

Impression d'un groupe insécable de blocs

Imprime consécutivement les trois blocs NomBloc1, NomBloc2 et NomBloc3.

Ces trois blocs sont alors considérés comme insécables. Si l'espace disponible dans la page courante ne permet pas leur impression globale, la page courante sera éjectée et une nouvelle sera préparée pour assurer l'impression associée des trois blocs.

Imprimer "NomBloc1, NomBloc2, NomBloc3"

Impression d'un groupe sécable de blocs

Imprime consécutivement les trois blocs NomBloc1, NomBloc2 et NomBloc3. Cette fonction permet cependant l'impression des trois commandes sur des pages différentes.

Imprimer "NomBloc1"
Imprimer "NomBloc2"
Imprimer "NomBloc3"

Impression des blocs avec réservation de place

Imprime les blocs NomBloc1, etc. à condition qu'il reste suffisamment de place sur la page courante pour les intégrer, ainsi que ceux notés entre parenthèses. Néanmoins, seuls les blocs qui ne sont pas entre parenthèses seront imprimés.

Imprimer "NomBloc1, etc…, (NomBlocN),(etc…)"

Exemple :Lors de l'édition des clients regroupés par département, le nom du département est imprimé chaque fois qu'il change. Pour éviter que le nom d'un département soit isolé en bas de l'une des pages, indiquez dans le script :

Imprimer "BlocDépartement, (BlocClient) "

Cette syntaxe permet d'imprimer le bloc BlocDépartement uniquement si la page courante peut le contenir avec le bloc BlocClient. Cela garantit l'édition d'au moins un client après le nom du département.

2. L'accès aux données : L'instruction "Liste"

a) Accéder à une table ou à une requête SQL : Les listes explicites

La création d'une liste explicite :

Crée une liste d'enregistrements à lire dans la table Access *NomTable* ou à générer selon la requête SQL.

NomVar = Liste "NomTable"
ou
NomVar=Liste " RequêteSQL"

Exemple : L'exemple suivant construit une liste contenant tous les clients :

ListeClient = Liste "Client"

ou

ListeClient = Liste "SELECT * FROM Client"

Le classement d'une liste explicite :

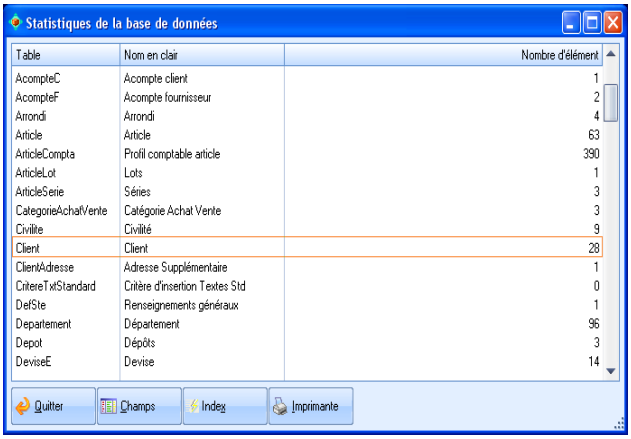
Il est ensuite possible de forcer le classement de cette liste par un index présent sur la table Crée une liste d'enregistrements à lire dans la table Access *NomTable,* triée selon l'index *NomIndex* correspondant à un index existant.

NomVar = Liste "NomTable" classée par "NomIndex"

Exemple** : **Construire une liste de clients classée par compte

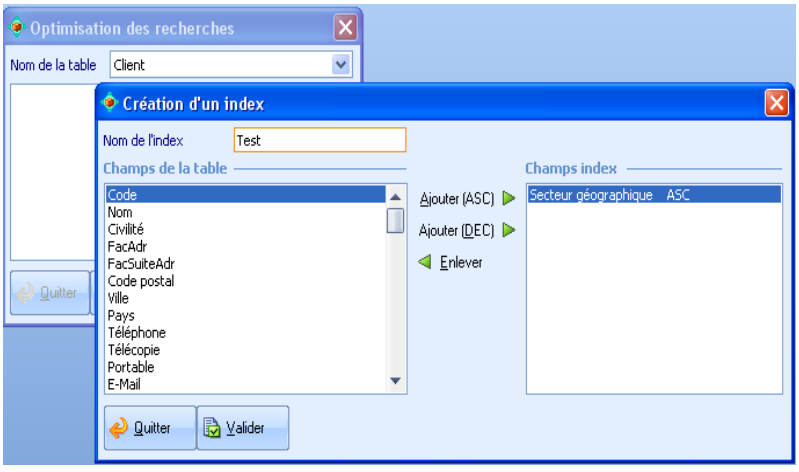
ListeClient = Liste "Client" classée par "Compte"

Vous obtenez la liste des index d'une table via le *menu "Utilitaires / Statistique de la base"* puis le bouton **Index**.



Si nécessaire, vous pouvez créer un index via le menu *"Utilitaires / Configuration avancée / Optimisation des recherches"*.

Les captures ci-dessous illustrent le mode de création d'un index (classement) sur la table "client" postant sur le secteur géographique de manière ascendante.



b) Utilisation des filtres et classement d'avant édition : Les listes implicites

Très souvent, une édition est obtenue à partir d'une liste de données affichée, sur laquelle des filtres ou des sélections ont été appliqués. Dans ce cas, le script d'édition doit tenir compte des présélections réalisées. La syntaxe suivante permet de manipuler les données de la vue courante :

NomVar = Liste courante

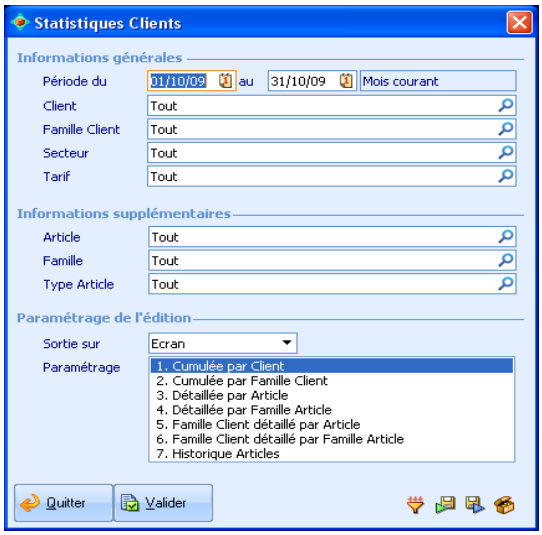
Les mots **Liste** et **Courante** doivent être tapés littéralement.

La mention *Classée par "Champ"* permet de s'assurer que la liste courante est bien classée par le champ précisé.

Cette vérification de l'ordre de classement est indispensable lorsque le script d'édition prévoit la gestion des regroupements d'enregistrements avec des ruptures sur un champ.

Lors de l'édition, si la vue n'est pas classée par le champ spécifié, le logiciel affichera un message du type « Edition impossible : la liste doit être classée par "Champ". »

NomVar = Liste courante classée par "Champ"



Vous pourrez utiliser cette instruction si vous souhaitez récupérer les filtres et classements des statistiques clients dans votre liste.

c) Construction d'une liste liée à l'enregistrement d'une autre : Les listes liées

Vous pouvez accéder aux enregistrements d'une liste liée à l'enregistrement d'une autre liste, en utilisant la syntaxe suivante :

NomVar2= Liste "NomTable" de NomVar1

Exemple : Pour éditer tous les clients et, pour chacun, la liste des devis réalisés, précisez :

Cli = Liste "Clients" =>créez une variable associée à une liste de clients.

Dev = Liste "Devis" de Cli =>créez une variable associée à la liste des devis du client désigné par la variable Cli.

3. Fonctions et instructions liées aux listes

a) Les instructions de positionnement

Une liste étant de manière générale constituée de plusieurs enregistrements, les instructions suivantes vous permettent de manipuler celle-ci afin de vous positionner sur l'enregistrement souhaité.

Se positionner au début de la liste

LITPREMIER(NomVarListe)
Se positionner à la fin de la liste
LITDERNIER(NomVarListe)

Lire l'enregistrement suivant

LITSUIVANT(NomVarListe)

Lire l'enregistrement précédent

LITPRECEDENT(NomVarListe)

Lire un enregistrement suivant une valeur de l'index

Cette instruction ne fonctionne que sur les listes construites à partir d'un index, du type *Liste "Client" classée par "Code"*

LIT NomVarListe , ValeurIndex

Exemple** : **LCli = Liste "Client" Classée par "Code"

Lit LCli, "00000010"

Positionne la liste des clients sur le client dont le code est égal à "00000010"

b) Les autres fonctions

Dans toutes les fonctions suivantes, NomVarListe représente une variable associée à une liste de données, créée à l'aide de la syntaxe NomVarListe=Liste ″NomTable″.

Calculer le nombre d'enregistrements d'une liste.

NOMBRE(NomVarListe)

Déterminer s'il existe un enregistrement suivant.

Cette fonction renvoie "VRAI" si la liste est positionnée après le dernier enregistrement.

FIN(NomVarListe)

Déterminer s'il existe un enregistrement précédent.

Cette fonction renvoi "VRAI" si la liste est positionnée avant le premier enregistrement.

DEBUT(NomVarListe)

Note : Les fonctions FIN et DEBUT sont généralement utilisées conjointement avec les fonctions LIITSUIVANT et LITPRECEDENT afin de déterminer si tous les enregistrements d'une liste ont été parcourus.

4. L'architecture d'un script

a) Réaliser des instructions lorsqu'une condition est validée

Vous disposez de la syntaxe suivante :

Si Condition Alors Instruction1 Sinon Instruction2

Condition de comparaison

Remplacez « Condition » par une expression composée de valeurs et d'opérateurs de comparaisons afin de réaliser les tests suivants :

= égalité
< infériorité
> supériorité
<= infériorité ou égalité
>= supériorité ou égalité
<> différence

Exemple : Si A<2 Alors B=2 Sinon B=1

Les comparaisons peuvent être combinées grâce aux arguments OU et ET.
Si la condition est vérifiée, l'instruction1 est exécutée. Sinon, l'instruction2 est exécutée.
Il n'est pas obligatoire de préciser une instruction amenée par Sinon.

Exemple : Si (A>5) ou (client.Code="ABC") Alors Imprimer "Bloc"

Pour exécuter plusieurs instructions après le Alors ou le Sinon, utilisez la formule suivante :
Si Condition Alors
Instruction1
Instruction2
…
Sinon
InstructionN
InstructionN+1
…
FinSi
Condition de ressemblance
Vous disposez de la syntaxe suivante :
Si Variable Comme "Masque"

Exemple : Si Client.Nom Comme "A*Z" Alors…

Cette syntaxe réalisera un traitement si le code du client commence par A et se termine par Z.

La codification du masque est la suivante :

1. L'étoile "*" indique un nombre indéterminé de caractères (sans test de type),

2. Le point d'interrogation " ?" indique un caractère (sans test de type),

3. Le dièse "# " indique un chiffre de 0 à 9,

4. Les crochets "[","]" permettent de préciser un ou plusieurs intervalles de valeurs pour un caractère,

5. Le point d'exclamation " !" à l'intérieur d'un intervalle exprime l'exclusion,

6. Un autre caractère indique une position dont la valeur est connue.

Exemple : A ??E : Commence par "A" et finit par "E" sur 4 caractères.

[A-FT-Z]* : Commence par une lettre entre "A" et "F" ou "T" et "Z".

****[!A-F] : Ne finit pas par une lettre entre "A" et "F".***

b) Répéter des opérations : les boucles

Répéter un bloc d'instructions pour chaque élément d'une liste

Cette fonction permet la répétition de toutes les instructions situées entre "Pour chaque" et "Boucler", jusqu'au parcours complet de tous les enregistrements de la liste représentée par la variable NomVarListe.

Pour chaque NomVarListe

(instructions)

Boucler

Répéter un bloc d'instructions jusqu'à ce qu'une condition soit validée

L'activation de cette fonction entraîne la répétition de toutes les instructions situées entre Faire et Boucler, jusqu'à la rencontre de l'instruction Arrêter. Vous devez donc impérativement insérer une instruction Arrêter dans ce type de boucle, au risque de générer une boucle infinie.

Faire
…
Si condition alors Arrêter
…
Boucler

5. Les manipulations de données

a) Opérations sur les nombres

Calculer le sinus d'un nombre.
SIN( )

***Exemple* : Sin(45) donne 0.707106**

Calcule le cosinus d'un nombre.
COS( )

Exemple : **COS(30) donne 0.866025**

Calculer la tangente d'un nombre.
TAN( )

Exemple : **TAN(66) donne 2.246036**

Calculer la racine carrée d'un nombre.
RAC( )

Exemple : **RAC(16) donne 4**

Arrondir un nombre
ARRONDI(Val, Nbdéc)

***Exemple* : ARRONDI(TotalTTC, 2), le champ 'Total TTC' sera arrondi à 2 décimales.**

Convertir un nombre en lettres avec la notion Franc(s)
LETTRES <Chiffre>

Exemple : **LETTRES 412 =" Quatre cent douze Francs "**

Convertir un nombre en lettres avec la notion Euro(s)
LETTRESEURO (<Chiffre>)

Exemple : **LETTRESEURO 412 = " Quatre cent douze Euros"**

Note :LETTRES et LETTRESEURO peuvent également être appliquées comme formats sur une variable numérique.

CaractèresASCII
Permet d'obtenir les caractères ASCII de rang n.
CAR(<Chiffre>)

Exemple : CAR(215) donne " ×".

b) Manipuler les chaînes de caractères

Extraction des n premiers caractères.
GAUCHE(Txt, n)

Exemple : **GAUCHE("ABCD",2) donne "AB"**

Extraction des n derniers caractères.
DROITE(Txt, n)

Exemple : **DROITE("ABCD",2) donne "CD"**

Extraction des n caractères à partir de la position p.
MILIEU(Txt, n)

Exemple : **MILIEU("ABCDE",2,3) donne "BCD"**

Forcer la casse en majuscules.
MAJUS(Txt)

Exemple : **MAJUS("Durand") donne "DURAND"**

Forcer la casse en minuscules.
MINUS(Txt)

Exemple : **MINUS("DURAND") donne "durand"**

Compter le nombre de caractères d'un texte.
LNG(Txt)

Exemple : **LNG("Durand") donne 6**

c) Fonctions sur les dates

Calculer une date par rapport à l'ajout de période sur une autre date

DATEADD(période, n, date)

Le type de période à ajouter est spécifié par l'un des symboles suivants :

1. yyyy Année
2. q Trimestre
3. m Mois
4. d Jour
5. ww Semaine
6. h Heure
7. n Minute
8. s Seconde

Exemple *:* **DATEADD("m", 2, "01/01/08") donne "01/03/08".**

DATEADD("d", -3, "01/01/08") donne "29/12/07.

Calculer la différence entre deux dates de références.

DATEDIFF(période, date1, date2)

Les types de périodes sont identiques à la fonction précédente.

Exemple : DATEDIFF("d","01/01/08", "26/04/08") donne 115 (jours).

DATEDIFF("ww","01/01/08", "26/04/08") donne 17 (semaines).

6. Les autres fonctions

Modifier les attributs d'un bloc.

MODIFBLOC (″<Page>″ , ″<description de l'attribut>″)

Les attributs de blocs peuvent être les suivants :

1. + ou – C : cache ou active le bloc à l'impression
2. + V : vide le bloc de tous ces champs
3. + Sav : saut de page avant impression du bloc
4. + Sap : saut de page après impression du bloc...
5. temp : signifie que l'action de l'attribut est temporaire, le bloc reprendra ces attributs normaux après le passage.

Exemple : ModifBloc("HautPage" ," +C,temp") permet de ne pas éditer temporairement le bloc HautPage.

Provoquer l'éjection de la page courante.

EJECTEPAGE

Formater une variable.

Les différents formats utilisables sont disponibles dans l'aide.

FORMAT(<Nom de la variable>,<Format choisi>)

Exemple : Format(Devis.Date, " mmmm yyyy") permet de formater la date du devis comme suit : « nom du mois » et «année ».

Exécuter une requête SQL

Que ce soit une requête d'action de suppression ou de mise à jour, etc.

EXECUTE "<Requête>"

***Exemple** : EXECUTE " UPDATE Client SET Client.CP = '33 000';" permet la mise à jour des codes postaux des clients ; le code doit ici être '33000'.*

Afficher un message, contrôler une valeur

Cette instruction interrompt momentanément l'édition afin d'afficher une valeur dans une fenêtre. Elle sert principalement à rechercher les anomalies de logique lorsque des scripts complexes ne fournissent pas les résultats attendus.

VOIR (Valeur)

Exemple : Cli=Liste "ClientDef

Pour chaque Client

VOIR " Code du client : " & Cli.Code

Imprimer BlocClient

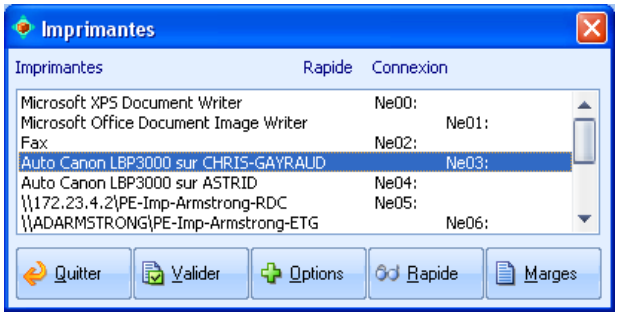
Boucler

Dans cet exemple, la fenêtre de suivi affichera successivement le code de chacun des clients.

Changer d'imprimante en cours d'impression.

SELECTIMPRIMANTE (" <Nom de l'imprimante> ")

Le nom de l'imprimante doit être identique à celui indiqué dans la fenêtre "Utilitaires / Paramétrage Imprimantes"

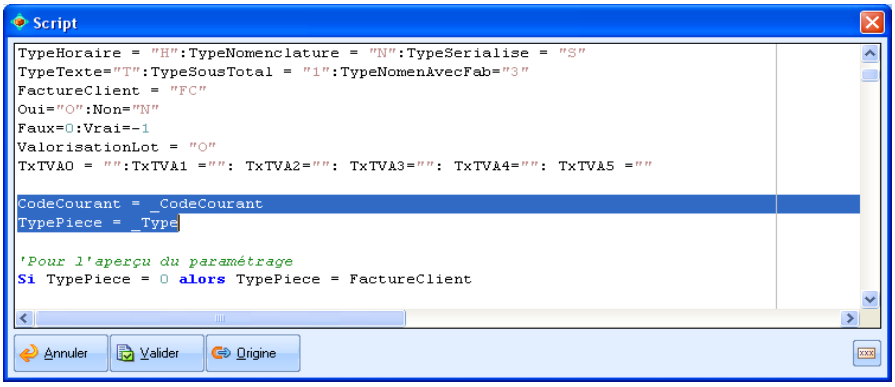


***Exemple** : SELECTIMPRIMANTE ("Auto Canon LBP3000 sur CHRIS-GAYRAUD")*

7. Les appels de fonctions externes

a) Principes

Les commandes précédées d'un « _ » précisent à l'interpréteur de script qu'il s'agit de commandes spécifiques au type de paramétrage utilisé.



L'exploitation de ces commandes sous-entend que le programmeur a mis en place un traitement particulier pour la commande dans les sources du programme.

Lors de l'exécution du script, lorsque le programme rencontre une commande préfixée d'un « _ », il invoque le programme correspondant au sein du logiciel. Une erreur apparaîtra si vous mentionnez une commande qui n'a pas été préalablement définie par le programmeur.

Aussi, utilisez **uniquement** les commandes précédemment codées par le programmeur dans les scripts (l'utilisateur ne peut pas créer ses propres commandes).

Ces commandes assurent la réalisation de traitements difficile, voire impossible à réaliser avec la syntaxe du script.

Par ailleurs, elles permettent également d'établir un dialogue entre le paramétrage et les données définies dans les masques d'éditions.

b) Exemples

Codecourant

Renvoie le code de la pièce traitée (devis ou facture).

_CODECOURANT

Espacelibre

Renvoie la valeur de l'espace libre sur la feuille.

_ESPACELIBRE

Espacesuffisant

Renvoie la variable ‘Vrai’ s’il y a assez de place pour imprimer le ou les bloc(s) souhaité(s).

_ESPACESUFFISANT <Chaîne de caractère contenant le Nom des blocs à tester>

Valeureuro

Renvoie la valeur de l’euro : 6.55957.

_VALEUREURO

8. La saisie de paramètres utilisateur

Lors de réalisation de paramétrage contenant des statistiques personnalisées, il est souvent nécessaire de laisser à l'utilisateur la possibilité de saisir des éléments qui seront repris sur l'édition.(avoir la liste des devis d'un client par exemple)

a) La syntaxe du bloc FORM / FINFORM

Tous les paramètres saisissables sont placés dans un bloc encadré par les instructions FORM et FINFORM et à chaque paramètre correspond une ligne commençant par l'instruction "Demander".

FORM

demander Libellé, Var, Type, Format, TypeListe

FINFORM

Le ‘Libellé’ représente le texte qui apparaîtra à l’écran.
'Var’ est la variable dans laquelle sera stockée la valeur saisie.

‘Type’ correspond au format de saisie (TEXT, NUM, DATE, MULTI).
'TypeListe' correspond à l'identifiant d'une liste pouvant s'afficher pour sélectionner l'élément.

Exemple** : **FORM

demander "Code client",Codecli,TEXT,9,CLIENT

demander "Nom client",Nomcli,TEXT,20

demander "Montant",Nontant,NUM,3.2

demander "Editer option", Edtopt, MULTI,"&Oui ;&Non"

...

FINFORM

b) Les différents types de données saisissables

Au "Type" correspond 4 types de données différents sur lequel un format particulier peut être appliqué.

TEXT :
Donnée de type texte, le format applicable détermine sa longueur maximale (20 pour 20 caractères maximum)

NUM :
Données numériques, le format applicable détermine le nombre de chiffre saisissable avant et après la virgule (3.2 pour 3 chiffres avant la virgule et 2 décimales)

DATE:
Une date, le format est toujours JJ/MM/AA

Important :Le format des dates utilisé dans les bases Access est le format Américain, ainsi, il est nécessaire d'appliquer le format "MM/DD/YY" à la variable saisie si vous souhaitez l'utiliser dans une requête.

MULTI:
Champ Multichoix dont les choix sont spécifiés dans le format.
Chaque choix est séparé par un point-virgule ";" , le caractère "&" désigne la lettre identifiant le choix.

c) Sélectionner un paramètre par une liste

Suivant le paramètre attendu, la saisie de celui-ci peut être simplifiée en l'associant à une liste, permettant ainsi à l'utilisateur de sélectionner l'élément souhaité.

demander "Code client",Codecli,TEXT,9,CLIENT

Le paramètre "CLIENT" indiqué en fin de ligne permet d'appeler la liste des clients.

Note :Chaque produit possédant une liste de paramètres différents, ils ne seront pas détaillés dans cet assistant.Toutefois, il vous est possible d'obtenir cet identifiant en ouvrant la table "SysVues", dans le champ "TypeVue".

III. Les formules de calculs

1. Leur rôle, différence par rapport au script

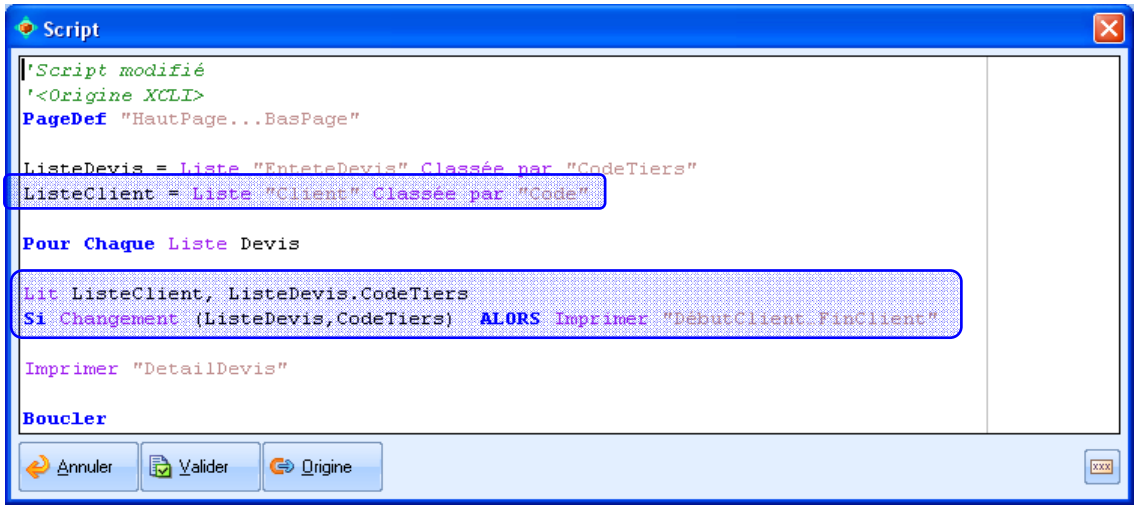
Une formule permet, lors de l'impression, de réaliser des calculs ou des tests de condition non prévus par le script d'édition.

Leur rôle premier est de calculer une valeur en vu de son impression, cela peut être un simple calcul (le prix unitaire remisé sur les factures clients par exemple) ou un cumul (un sous-total TTC des devis par client).

Leur syntaxe est similaire au script d'édition, il est ainsi tout à fait possible de créer des listes, des variables, des boucles, des conditions....mais contrairement au script présent tout au long de l'édition, la formule de calcul s'exécute lors de l'impression d'un bloc.

Nous allons aborder la réalisation d'un sous-total TTC par client sur une édition libre listant les devis client. Ainsi une modification du script est nécessaire afin d'ajouter les blocs d'impression nécessaire à la rupture client.

Les modifications apportées sont encadrées :

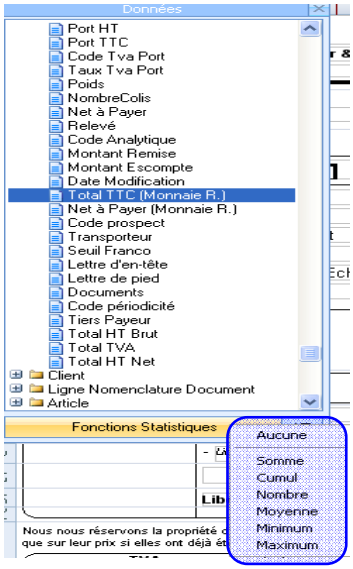


Une fois ces modifications apportées, les blocs "Debutclient" et "FinClient" sont rajoutés sur l'édition et les données du client seront disponibles.
Vous pourrez alors utiliser 2 méthodes pour créer la formule totalisant les Total HT par client, via les **fonctions statistiques** ou via les **variables globales.**

2. Réaliser un cumul via les fonctions statistiques

Pour établir la somme des montants TTC des devis d'un client, utilisez la fonction statistique 'Somme', issue de la liste des rubriques.

Pour placer sur le paramétrage la somme des totaux T.T.C., réalisez un glisser-déplacer en cliquant et en maintenant le bouton gauche de la souris enfoncé jusqu'à l'emplacement souhaité, du champ « Total T.T.C. » de la table 'Devis' vers le paramétrage.



En cliquant sur le bouton "Fonction Statistiques", 6 fonctions vous sont proposées :

1. **Somme** : additionne une donnée et remet le calcul à zéro après chaque impression du résultat.

2. **Cumul** : additionne une donnée, sans jamais remettre le calcul à zéro.

3. **Nombre** : compte le nombre d'impressions d'une donnée.

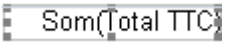
4. **Moyenne** : calcule la moyenne des valeurs d'une liste de données (*Somme()* divisé par *Nombre()*).

5. **Minimum** : donne la plus petite valeur pour une liste de données imprimée.

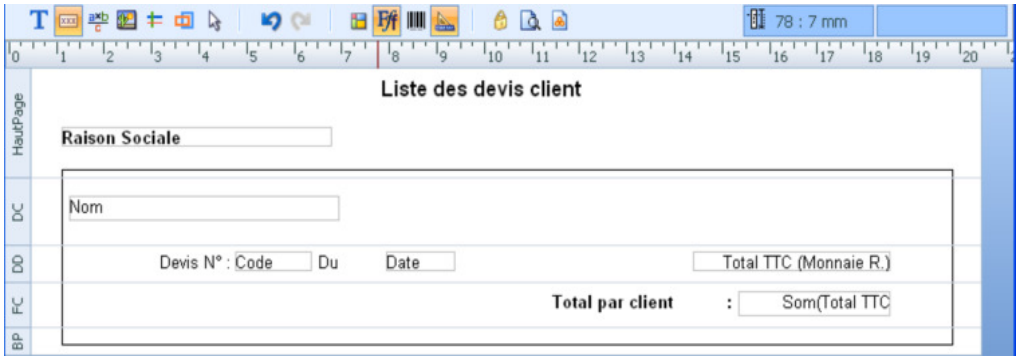
6. **Maximum** : donne la plus grande valeur pour une liste de données imprimée.

Après avoir sélectionné la fonction souhaitée, cliquez et déplacez le curseur jusqu'à atteindre la zone où vous voulez visualiser le résultat (identique au placement d'une donnée)

En l'occurrence, le champ « Somme » doit être inséré dans le bloc 'FinClient', puisqu'à chaque fin d'impression d'un client correspond un total de montant T.T.C. de devis.

Réalisez un clic une fois la position atteinte. Le champ suivant s'affiche alors : 

La maquette finale du paramétrage avec les fonctions statistiques est la suivante :



a) Différence entre la fonction "Somme" et "Cumul"

Les fonctions statistiques 'Somme' et 'Cumul' sont déterminées et différenciées par la remise à zéro ou non de leur calcul.

Dans l'exemple de paramétrage précédent, chaque client édité est associé à la liste des devis qui le concerne et au montant T.T.C. des devis. Par conséquent, un client peut avoir plusieurs devis à son actif.

L'utilisation de la fonction 'Somme' sur le champ « TotalTTC » de la table Devis permet d'obtenir la somme des montants T.T.C. de chaque client.

Pour obtenir le résultat souhaité par client, positionnez la formule sur un bloc de fin de rupture (ici FinClient). Ainsi, pour chaque client, la somme des montants T.T.C. des devis sera imprimée sur ce bloc.

La fonction 'Cumul' permet, sur ce même champ, de cumuler les montants T.T.C. des devis de tous les clients. En effet, cette fonction « stocke » les sommes des montants T.T.C. des devis de chaque client pour obtenir un total général.

Afin d'afficher la somme générale des montants T.T.C. des devis (ici le bloc BasPage), placez la formule 'Cumul' sur un bloc de fin de rupture générale (c'est-à-dire un bloc ne s'imprimant qu'une seule fois).

Donc, la somme réalise une addition de champs et en affiche le total pour un critère donné (le client), équivalent d'une remise à zéro de l'addition à chaque nouveau client.

A l'inverse, le cumul **incrémente** les données à chaque édition de client.

Ces fonctions statistiques ne sont cependant pas réutilisables dans les formules de calculs.

3. Réaliser un cumul via les variables globales

Cette méthode décrit la saisie d'une formule calculant la somme des totaux TTC des devis.
Le principe est de déclarer une variable globale à l'édition remise à 0 lors du DebutClient, additionnant le total TTC pour chaque devis imprimé, pour s'imprimer en FinClient.

Chaque formule peut être constituée d'un ensemble de lignes, lesquelles produiront un résultat unique. Lors de l'impression d'un paramétrage, chaque zone formule sera remplacée par le résultat qu'elle calcule.

a) Généralités

Constituants des formules

Les formules sont constituées d'opérations (les symboles), d'opérandes (les nombres) et de tests de conditions (Si...Alors...Sinon).

Autres constituants

Les parenthèses permettent de combiner les opérations.

 ***Exemple** : **Exemple : 2 × (3 + 4)***

Le signe « **:** » peut être utilisé pour séparer plusieurs instructions dans une ligne.

 ***Exemple** : **Exemple : C = 2 : D = 3***

Des commentaires peuvent être saisis au sein des formules, précédés d'une **apostrophe**.

 ***Exemple : Valeur1 = 10 ' affectation de la variable Valeur1***

Valeur2 = 20 ' affectation de la variable Valeur2

Total = Valeur1 + Valeur2 ' calcul du total

Résultat d'une formule

Pour qu'une valeur soit imprimée à l'emplacement d'une formule, au moment de l'impression, il faut que cette formule produise un résultat.

Toutes les opérations produisent un résultat, sauf les affectations de variables.

 ***Exemple : 2 × (3 + 4) Cette formule produit le résultat 14, qui sera imprimé à l'emplacement de la formule.***

A = 2 × (3 + 4) Cette formule réalise le même calcul que la précédente, l'affecte à la variable A mais ne produit aucun résultat imprimable.

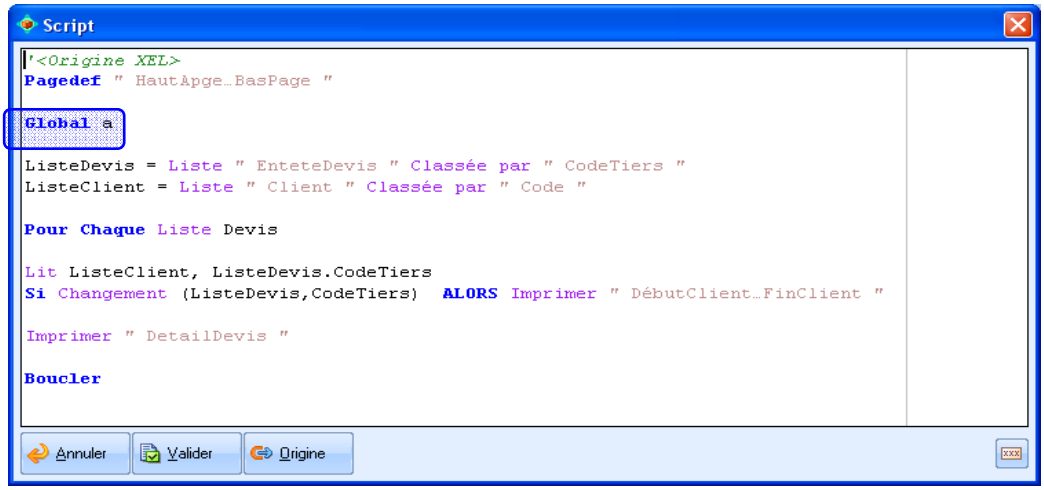
Néanmoins, la variable A contient bien la valeur 14 et pourra être employée dans une autre formule.

b) Utilisation des variables

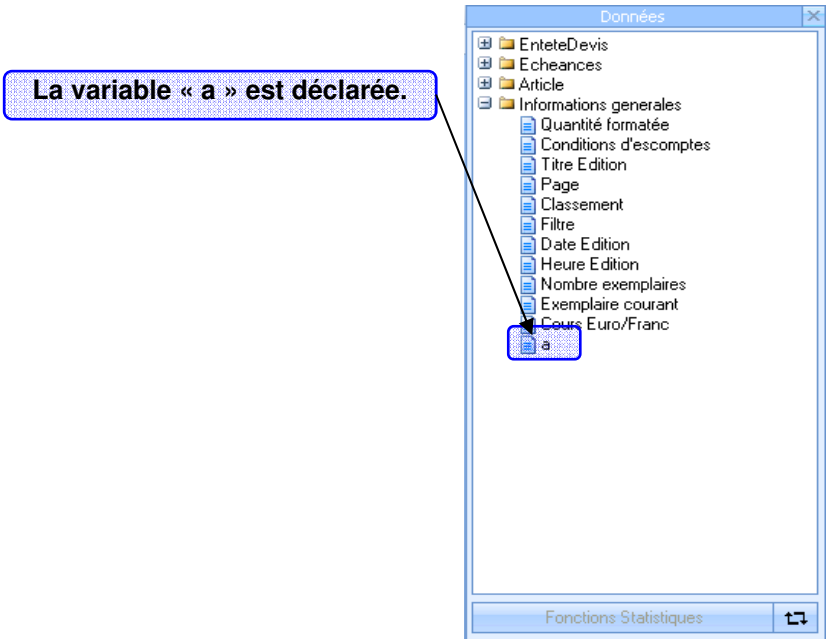
Pour utiliser les variables, il est nécessaire de les déclarer en tant que globales afin de les rendre accessibles.

La syntaxe générale du script à utiliser est la suivante : *Global (Nom variable)*

Le nom de la variable est entièrement libre. Dans l'écran ci-dessous, il se nomme 'a' et apparaît sous la forme suivante :



Après déclaration de la variable, celle-ci est récupérable en tant que champ dans les *Informations générales* de la liste des rubriques :



Note :Si vous aviez inséré le champ « a » sur le paramétrage, aucune donnée n'aurait été imprimée, « a » n'ayant reçu aucune valeur.

c) Etapes de construction de la formule

Affectation d'une valeur à la variable

Sur le paramétrage, le total T.T.C. sera affecté à la variable 'a', ce champ regroupant les calculs.

Aussi, rédigez la formule de calcul suivante :

a = [LISTEDEVIS].[MonnaieRef_TotalTTC]

a

Insérez-la dans le bloc 'DetailDevis' en remplacement du champ « TotalTTC ». Elle vous permettra ainsi d'imprimer le montant T.T.C. du devis et sera intégrée dans le calcul final.

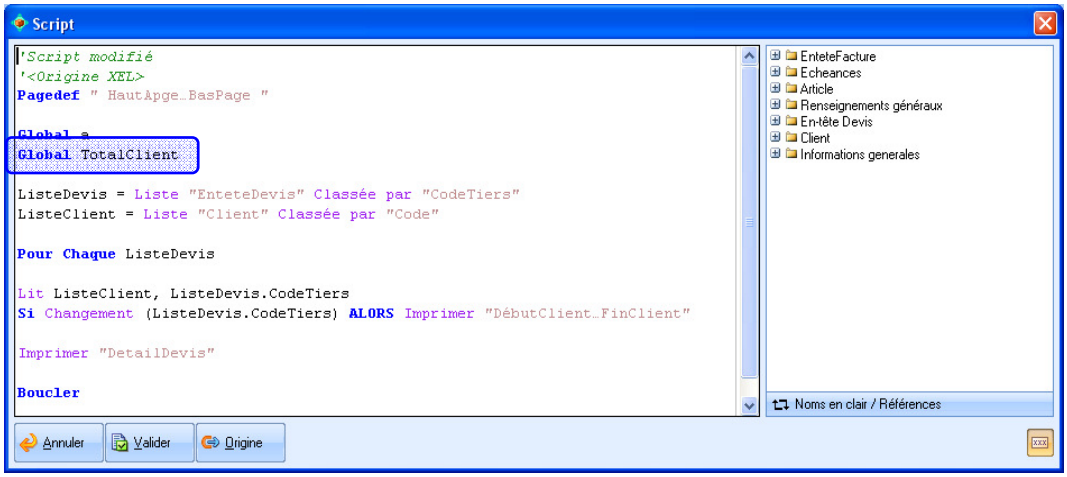
Afin d'obtenir la somme des montants T.T.C. des devis, réalisez une totalisation.

Lors de chaque impression des montants T.T.C. des devis, le résultat viendra incrémenter une nouvelle variable nommée 'TotalClient'.

Déclaration de la variable 'TotalClient'

Comme pour la déclaration de 'a', déclarez la variable '**TotalClient**' au niveau du script.

Le script se définit ainsi :



Initialisation de la variable 'TotalClient'

Initialisez maintenant la variable 'TotalClient' pour obtenir sa remise à zéro lors de chaque nouvelle impression d'un client.

Cette initialisation est réalisée via la formule de calcul suivante :

TotalClient = 0

La place de cette formule (qui ne s'imprime pas) est primordiale. En effet, si elle est placée dans le mauvais bloc, son traitement peut alors être entièrement faussé.

La formule d'initialisation doit être placée dans le bloc 'DébutClient' afin d'être remise à zéro lors de chaque impression d'un nouveau client.

Création de la formule 'TotalClient'

Cette formule reprendra au fur et à mesure du traitement les valeurs des montants T.T.C. (variable 'a') des devis d'un client par secteur géographique et les stockera dans la variable 'TotalClient'.

La formule à créer est la suivante :

TotalClient = TotalClient + a

A chaque passage sur un devis, **les valeurs de 'a' sont incrémentées à la variable 'TotalClient'**.

Note :La place de la formule est une nouvelle fois importante. En effet, si celle-ci est située dans le mauvais bloc, le traitement qui en découle pourra être entièrement faussé.

Par conséquent, la formule est à placer dans le bloc 'Devis', les calculs concernant le paramétrage et plus particulièrement le traitement à réaliser, s'effectuant à ce niveau.

Résultat de la formule 'Compteur'

Une fois les initialisations et les créations de formules concernant les totaux des montants T.T.C. des devis réalisés, vous devrez afficher sur le paramétrage, le résultat global.

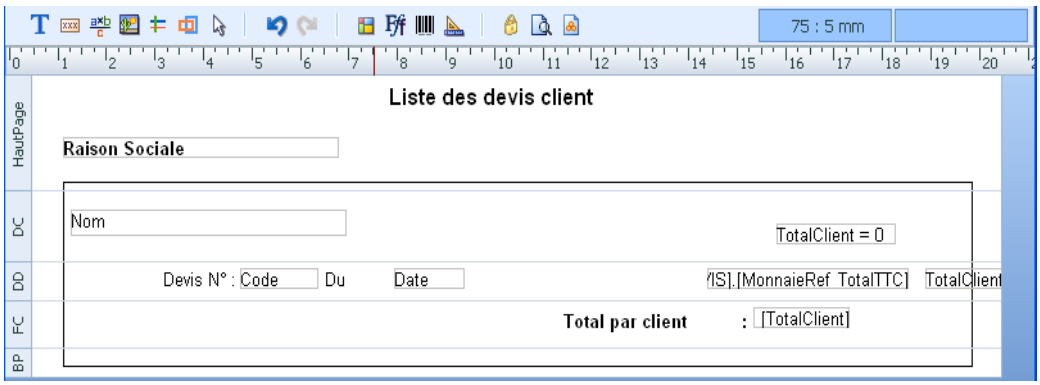
Pour cela, vous disposez de 2 options :

- La récupération du champ « TotalClient » de la table 'Informations générales'.
- La création d'une formule de calcul 'TotalClient'.

Cette dernière devra être insérée dans le bloc 'FinClient' ; son traitement est identique à celui des fonctions statistiques.

La variable « TotalClient » est déclarée.

Vous obtenez alors la maquette avec les formules de calcul suivantes :



IV. Exemples pratiques

Cette section a pour but de décrire, à travers deux exemples, la méthode utilisée pour répondre aux demandes de scripts personnalisés les plus courantes de nos utilisateurs.

1. Élaboration d'un état libre : Liste des devis client par client par secteur géographique

A travers cette demande, nous abordons les étapes nécessaires à l'élaboration d'un état à partir d'une édition libre vide.

En suivant les étapes d'analyse décrites lors de l'introduction de cet assistant, il apparaît que la demande "Je souhaite imprimer une liste des devis clients, par client, par secteur géographique" nécessite les étapes suivantes :

- Créer une liste
Créer une liste des clients
- Identifier le nom d'un champ en vu d'un classement
Identifier le champ correspondant au secteur géographique
- Classer une liste suivant un champ
Créer une liste des clients classée par secteur géographique
- Imprimer le détail des éléments d'une liste dans un bloc
Imprimer les informations du client
- Accéder aux détails d'une fiche en connaissant uniquement son code
Avoir le détail du secteur géographique à partir du code présent sur le client
- Réaliser une rupture lors du changement d'une valeur
Imprimer les informations du secteur géographique lors du changement de celui-ci
(*rupture Par Secteur géographique)*
- Imprimer une liste liée à une fiche
Imprimer le détail des devis du client en cours
- Imprimer les cumuls par rupture
Imprimer le total TTC des devis par secteur géographique
- Imprimer un cumul en fin d'édition
Imprimer le total TTC de tous les devis en fin d'édition
- Saisie et application de paramètres utilisateur
Filtrer l'édition sur un secteur géographique

Chaque étape abordée est facilement transposable à une autre demande, simplement en modifiant la ou les tables utilisées.

a) Création d'une liste

La première étape consiste à créer la liste des clients.
Deux méthodes sont possibles, soit par construction d'une requête SQL, soit par la syntaxe simple de l'instruction Liste.
Ainsi ces deux lignes ont le même résultat :

ListeClient = Liste "Client"

ListeClient = Liste "SELECT * FROM Client"

L'utilisation d'une requête SQL est beaucoup plus souple mais nécessite une connaissance de ce langage. Cette méthode permet notamment de choisir les champs de la table des clients qui nous intéressent.

Exemple : Vous souhaitez récupérer uniquement les champs « Code » et « Nom ».

Entrez la requête suivante :

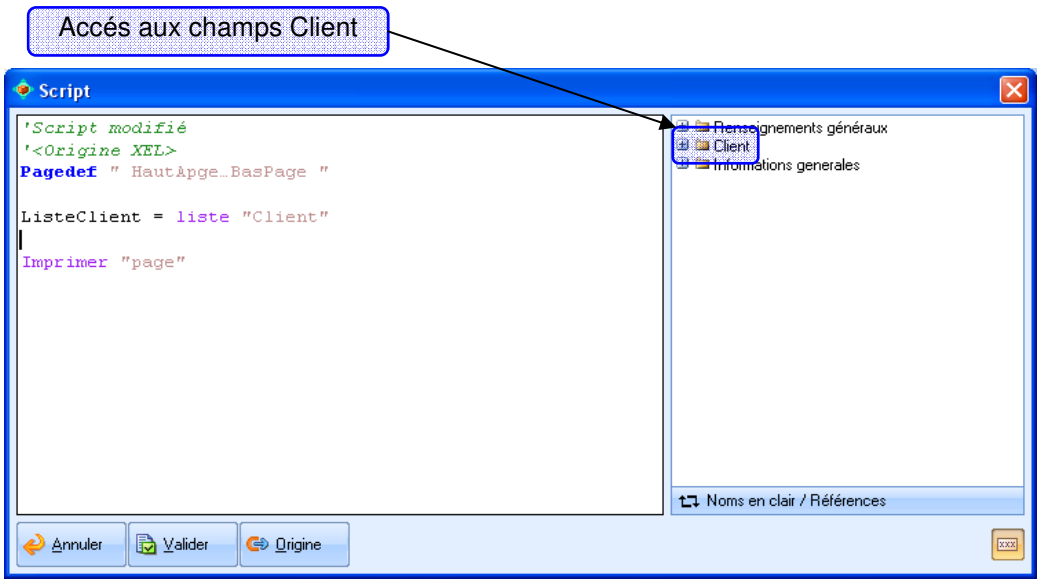
ListeClient = Liste « SELECT Code, Nom FROM Client ORDER BY Code »

La liste des rubriques affichera alors la table 'Clients' en incluant seulement les champs « Code » et « Nom ».

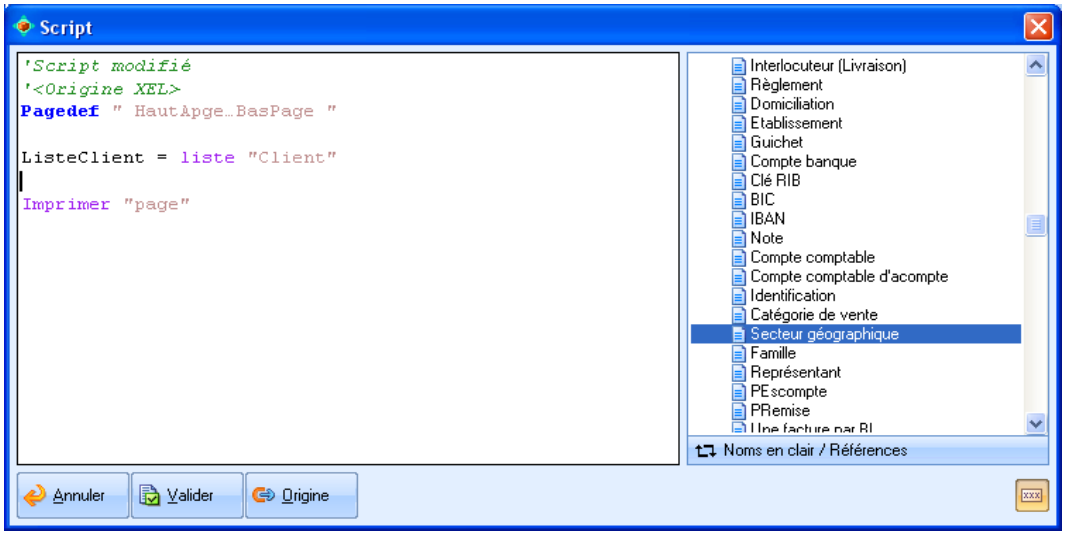
b) Identifier le nom d'un champ en vue d'un classement

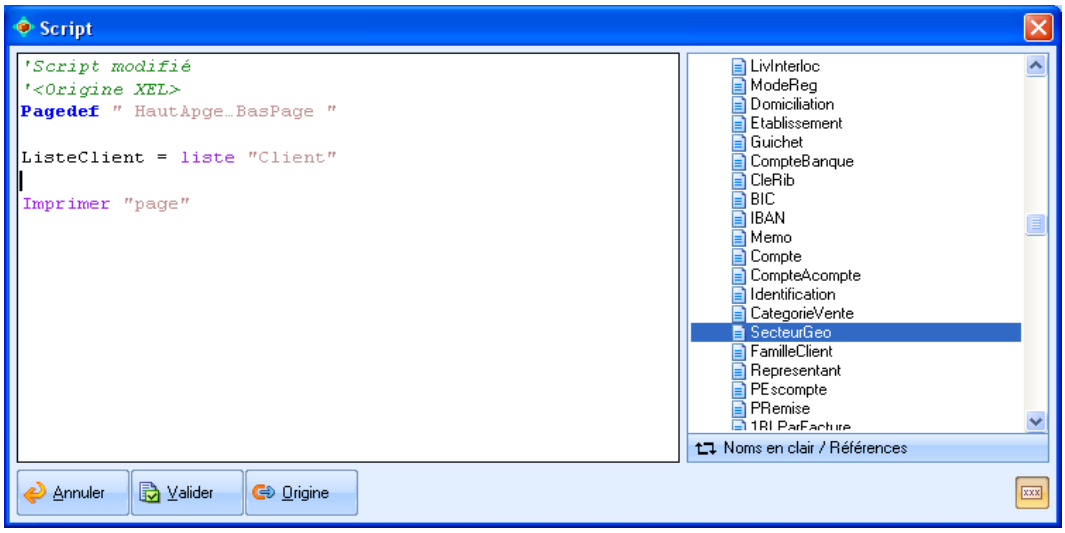
Après avoir validé l'une des deux lignes précédentes, les champs relatifs à la table "Client" apparaissent dans les données.

Cliquez sur le bouton afin d'y accéder :



Dans la liste des champs apparaît "Secteur Géographique", ce qui correspond au "nom en clair" du champ, cliquez sur "Noms en clair / Références" afin d'obtenir le nom réel du champ.





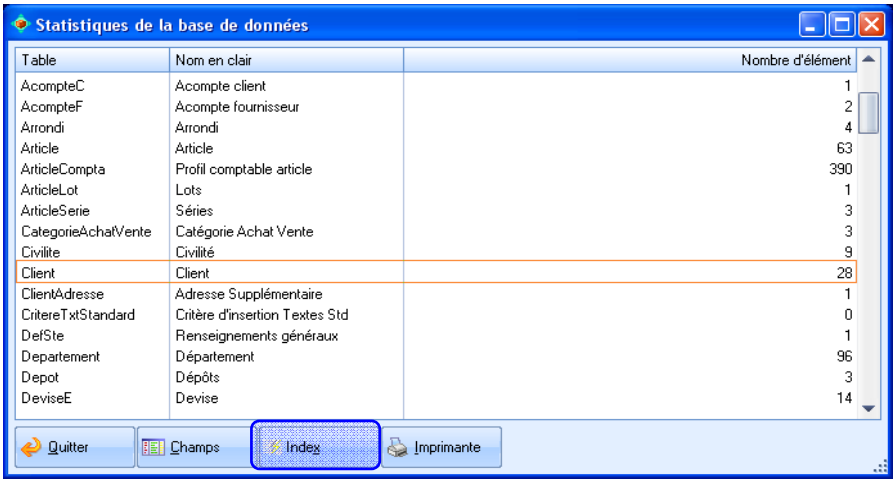
Ainsi, le nom du champ du classement est "SecteurGeo".

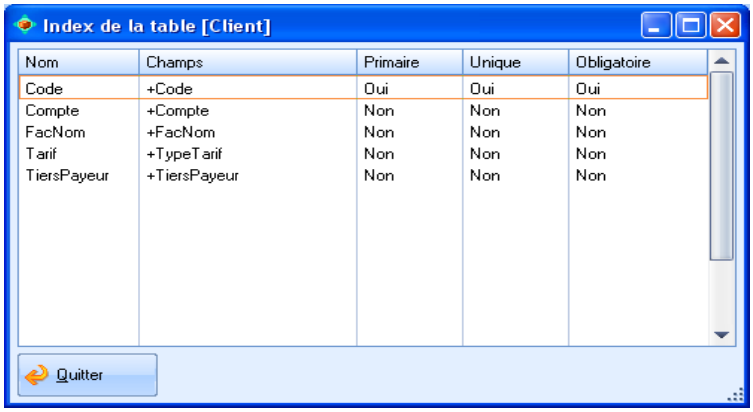
c) Classer une liste suivant un champ

Deux méthodes de classement sont possibles, soit par un index, soit par l'instruction "ORDER BY" d'une requête SQL.

Classer une liste par un Index

Vous obtenez la liste des index d'une table via le menu "Utilitaires / Statistique de la base" puis le bouton "Index"

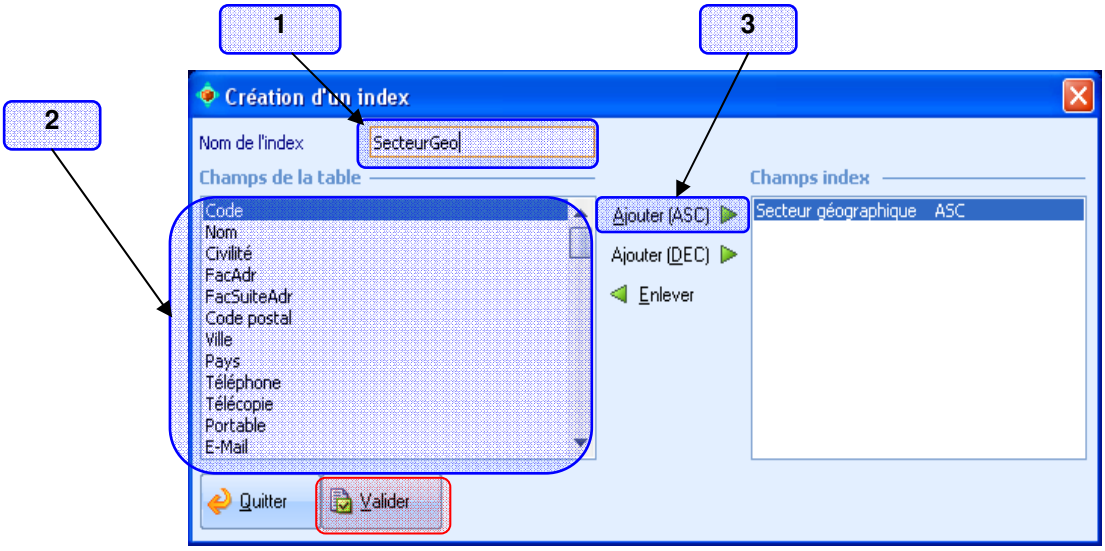




Vous constatez qu'aucun index n'est présent sur le champ "SecteurGeo" de la table "Client".

Afin de le créer rendez-vous dans le menu *"Utilitaires / Configuration avancée / Optimisation des recherches"*

Cliquez sur "Créer" :



1. Donnez un nom à l'index que vous souhaitez créer
Dans notre cas nous l'appellerons "SecteurGeo".

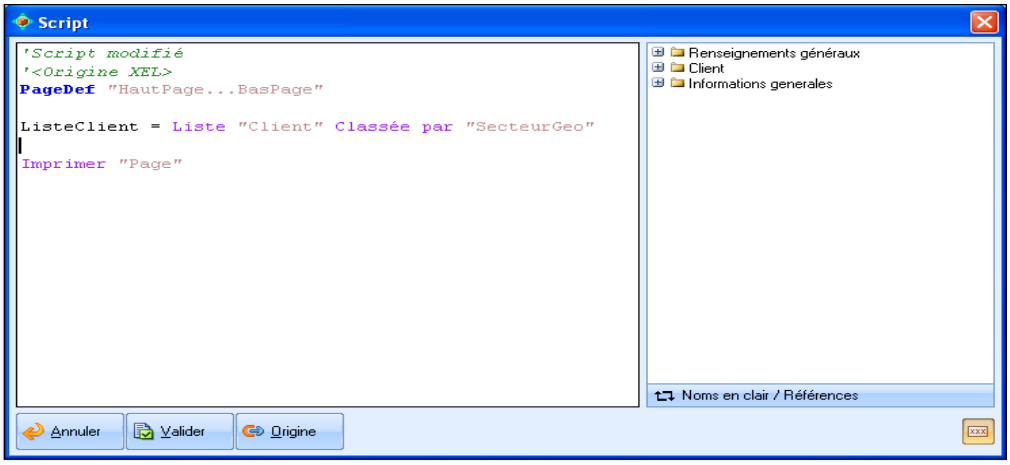
2. Sélectionnez le champ correspondant au classement voulu.
Le champ "Secteur Géographique" dans notre cas.

3. Cliquez sur le bouton correspondant au sens du classement voulu.
Dans notre cas, cliquez sur "Ajouter ASC" afin d'obtenir un classement ascendant.

En validant ces manipulations, l'index "SecteurGeo" apparaît dans la liste des index de la table client :

Nous pouvons désormais utiliser cet index afin de classer la liste des clients avec l'instruction :

ListeClient = Liste "Client" classée par "SecteurGeo"



Classer une liste suivant un champ

Nous avons précédemment vu qu'une liste peut être le résultat d'une requête SQL, ainsi l'utilisationde l'instruction "ORDER BY" vous permet de classer votre liste suivant le ou les champs désirés.

Dans notre exemple, nous souhaitons classer la table client suivant le champ "SecteurGeo", l'instruction correspondante serait :

ListeClient = Liste "SELECT * FROM Client ORDER BY SecteurGeo"

d) Imprimer le détail des éléments d'une liste dans un bloc

La liste des clients étant créée, vous souhaitez imprimer le code et nom de chaque client.

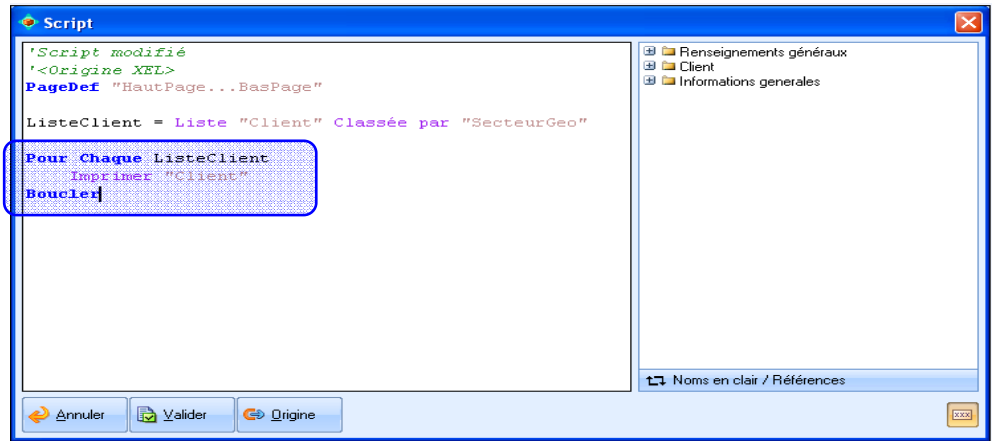
L'instruction **'Pour Chaque ... Boucler'** correspond à une boucle assurant la répétition pour l'impression des enregistrements d'une table.

La requête est alors la suivante :

Pour Chaque ListeClient
*** Imprimer "Client"***
Boucler

Cela signifie que pour chaque élément qui constitue la liste "ListeClient" le bloc d'impression "Client" sera imprimé.

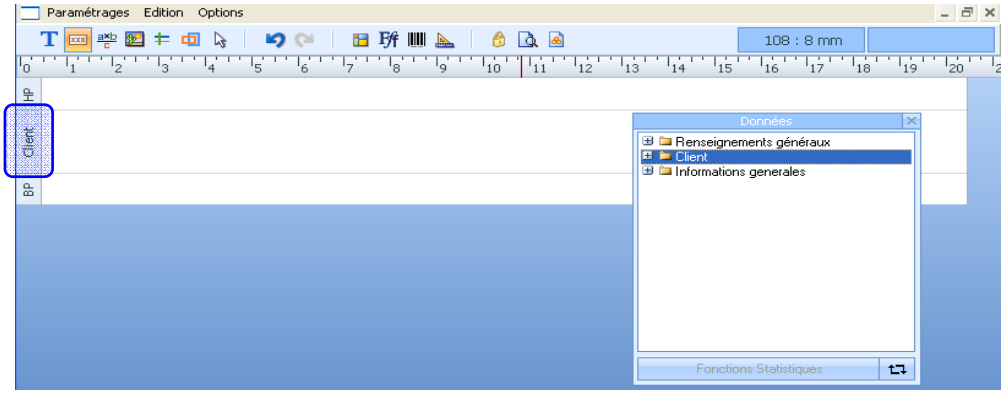
Le script devient donc :



En validant ce script, deux modifications sont apportées sur la partie graphique :

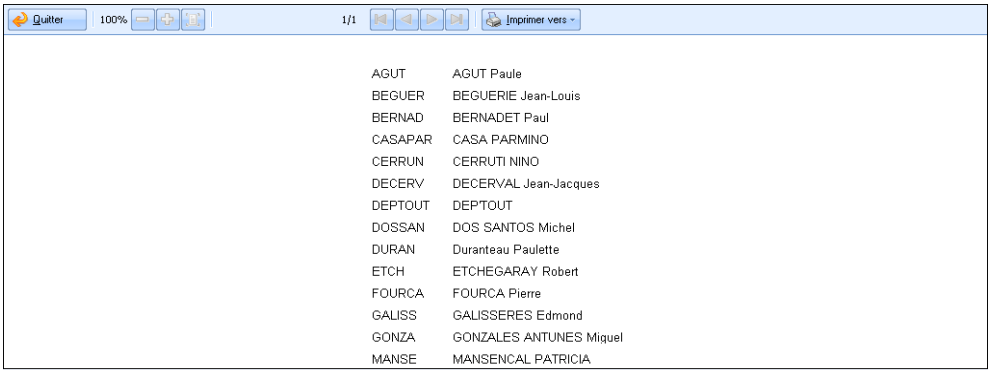
1. L'ajout du bloc d'impression "Client".
2. L'accès aux champs de la table client dans les données via le bouton  de la barre d'icônes.

En validant ce script, deux modifications sont apportées sur la partie graphique :



1. L'ajout du bloc d'impression "Client".

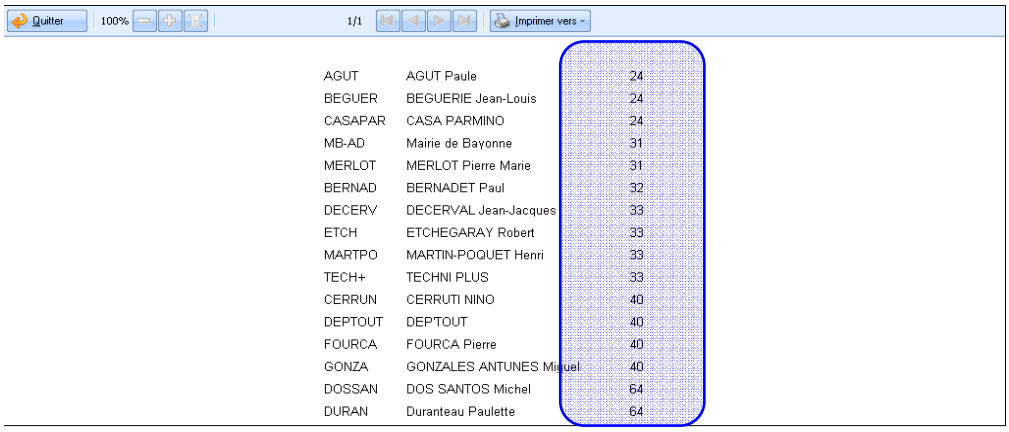
2. L'accès aux champs de la table client dans les données via le bouton  de la barre d'icônes.
Glissez les champs "Nom" et "Code" dans le bloc Client, vous obtenez ce résultat sur l'aperçu :



e) Accéder aux détails d'une fiche en connaissant uniquement son code

Le but de l'édition étant d'obtenir la liste des devis client par secteur géographique, vous souhaitez imprimer le nom du secteur géographique .

Dans les données client, apparaît le champ 'SecteurGeo', voici ce que vous obtenez en l'ajoutant sur l'édition précédente :



Comme vous pouvez le constater, ce champ ne contient que le code du secteur géographique du client en cours d'impression, or vous souhaitez également imprimer le nom de celui-ci.

Il est donc nécessaire de :

- créer une liste contenant les secteurs géographiques
- de la positionner sur le secteur dont le code correspond au champ du client

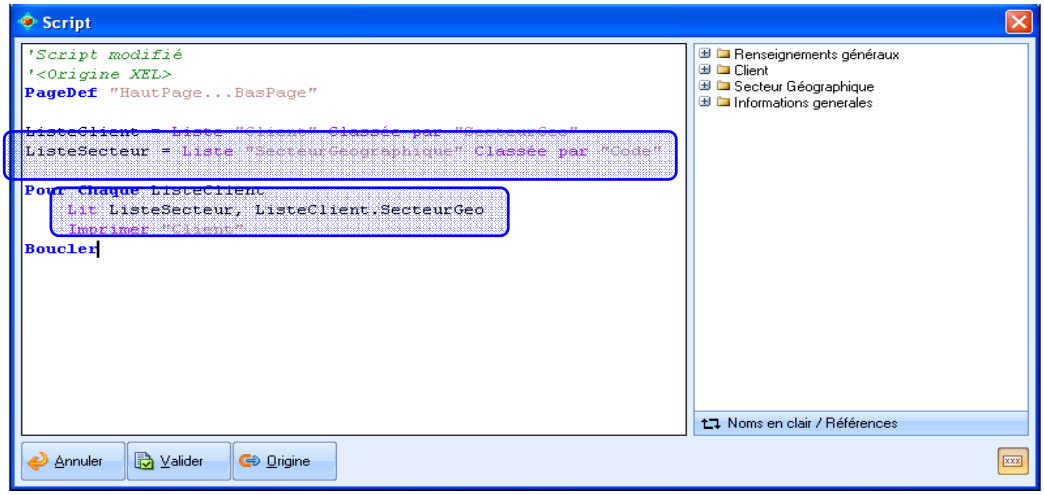
La création de la liste des secteurs géographique se fait par l'instruction "Liste" :

ListeSecteur = Liste "SecteurGeo" Classée par "Code"

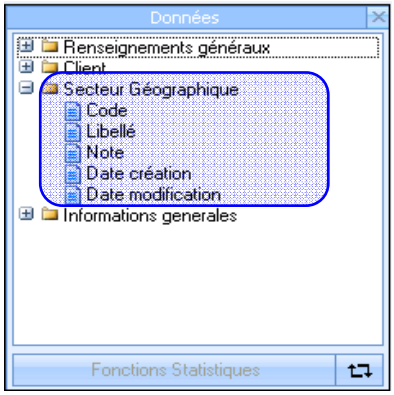
Son positionnement par rapport au champ "SecteurGeo" de la liste "ListeClient" se réalise par l'instruction "Lit" :

Lit ListeSecteur, ListeClient.SecteurGeo

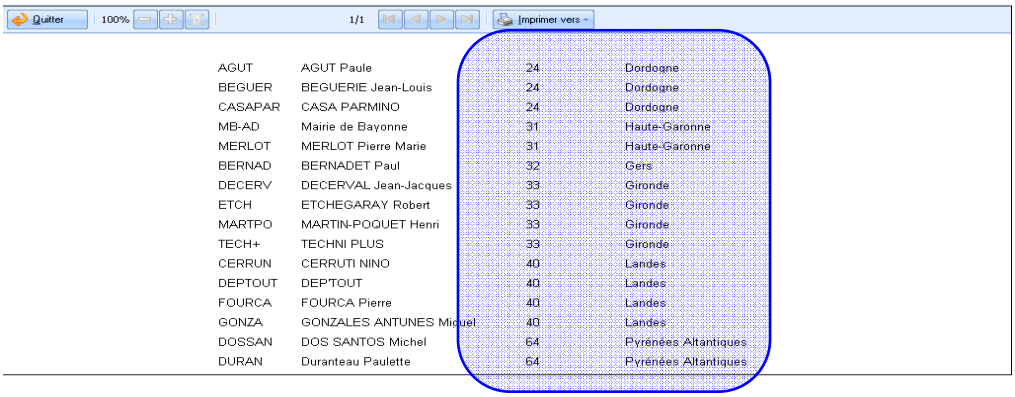
Important :Cette instruction doit obligatoirement être placée dans la boucle imprimant le détail de chaque client afin de repositionner la liste des secteurs géographique sur celui du client en cours d'impression :



A la validation de ce script, apparaissent les informations du secteur géographique dans les données disponibles pour l'édition :



En ajoutant le champ "Libellé" dans le bloc "Client", vous obtenez l'édition suivante :



AGUT	AGUT Paule	24	Dordogne
BEGUER	BEGUERIE Jean-Louis	24	Dordogne
CASAPAR	CASA PARMINO	24	Dordogne
MB-AD	Mairie de Bayonne	31	Haute-Garonne
MERLOT	MERLOT Pierre Marie	31	Haute-Garonne
BERNAD	BERNADET Paul	32	Gers
DECERV	DECERVAL Jean-Jacques	33	Gironde
ETCH	ETCHEGARAY Robert	33	Gironde
MARTPO	MARTIN-POQUET Henri	33	Gironde
TECH+	TECHNI PLUS	33	Gironde
CERRUN	CERRUTI NINO	40	Landes
DEPTOUT	DEPTOUT	40	Landes
FOURCA	FOURCA Pierre	40	Landes
GONZA	GONZALES ANTUNES Miguel	40	Landes
DOSSAN	DOS SANTOS Michel	64	Pyrénées Atlantiques
DURAN	Duranteau Paulette	64	Pyrénées Atlantiques

Le libellé imprimé correspond bien au code du secteur géographique stocké sur le client.

f) Réaliser un regroupement suivant une valeur (notion de rupture)

Vous souhaitez maintenant regrouper tous les clients possédant le même secteur géographique.

Note :Réaliser un regroupement nécessite au préalable d'avoir construit une liste classée suivant le regroupement voulu. Dans notre cas, la liste des clients est classée par secteur géographique.

En étudiant l'édition précédente, cela revient au lieu d'imprimer le détail du secteur géographique pour chaque client, à imprimer un nouveau bloc avec le code et le nom du secteur géographique avant la liste des clients de ce secteur.

L'instruction à utiliser est la suivante :

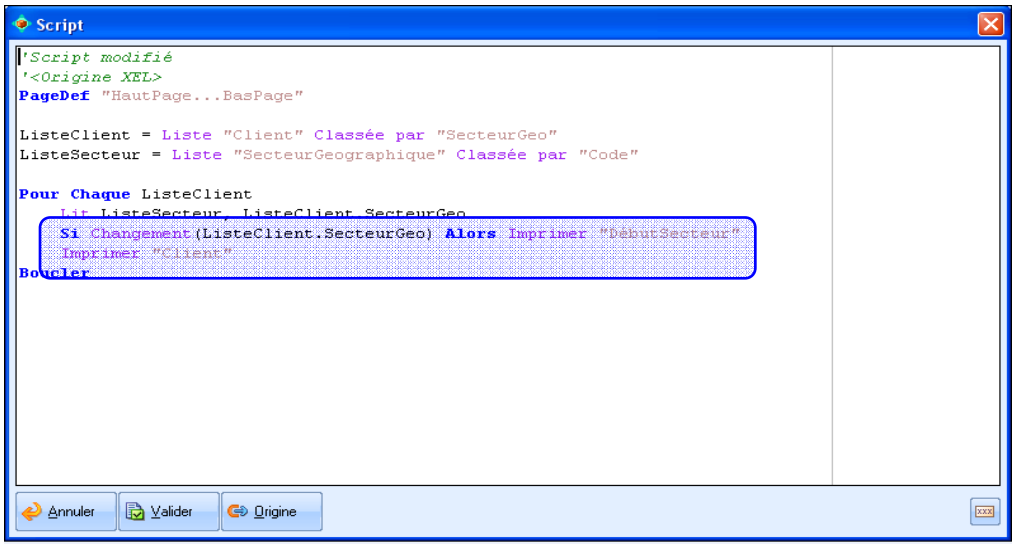
Si Changement(ValeurTestée) Alors Imprimer "Bloc."

Ainsi, l'institution correspondant au besoin ".....Par Secteur Géographique" serait dans notre script :

Si Changement(ListeClient.SecteurGeo) Alors Imprimer "DébutSecteur"

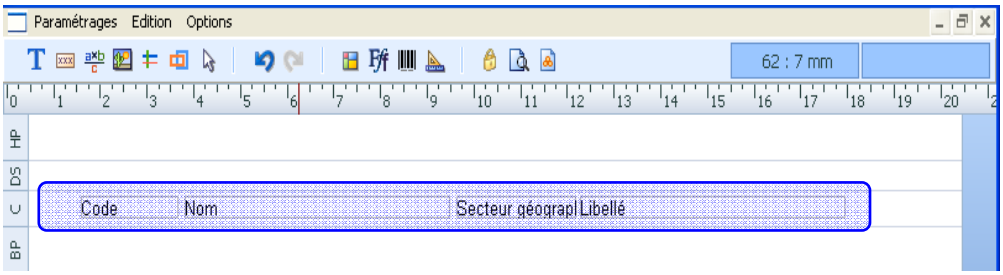
Si le code du secteur géographique d'un client est différent du client précédent, les blocs de début et de fin seront imprimés : il s'agit d'une rupture.

Le script d'édition se présente ainsi :

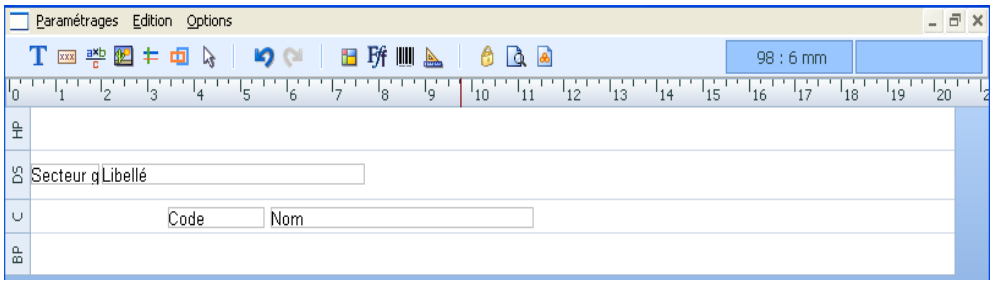


Important :Cette instruction doit impérativement être placée avant l'instruction "Imprimer Client" afin que le bloc "DébutSecteur" soit imprimé avant le détail des clients.

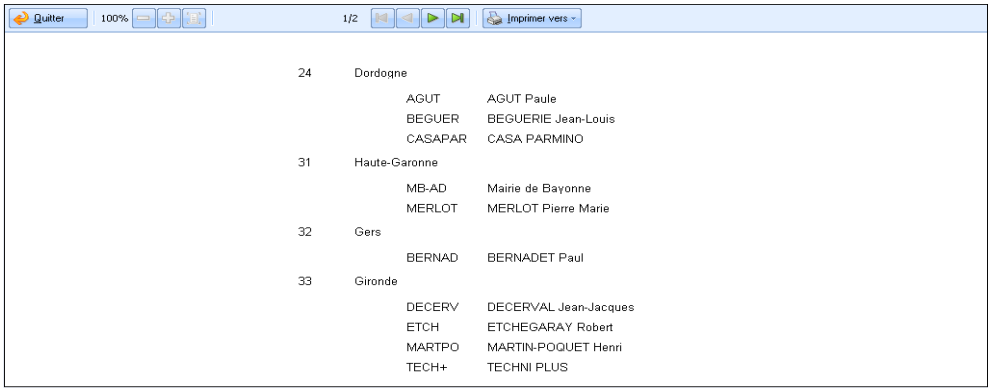
Le paramétrage est alors modifié. Le nouveau bloc "DebutSecteur" vient d'être inséré via l'instruction *Imprimer "DébutSecteur"*, pour permettre l'impression de la rupture.



Le regroupement des clients se faisant par secteur géographique, vous glissez les informations code et libellé du secteur géographique dans le bloc "DébutSecteur" :



Ainsi l'indication du secteur géographique ne sera plus imprimée pour chaque client mais uniquement pour tous les clients rattachés au même secteur. Ce qui donne l'édition suivante :



g) Imprimer une liste liée à une fiche

Vous souhaitez éditer, pour chacun des clients, la liste des devis le concernant.

La méthode utilisée consiste a :

- Créer une liste contenant tous les devis
- Pour chaque client, réaliser une boucle de l'intégralité de la liste des devis
- Imprimer un nouveau bloc "Devis" si le code du client correspond au client en cours.

Pour cela, nous utiliserons les instructions :

'LitPremier()', 'LitDernier()', 'LitSuivant()', 'LitPrécédent()' permettant de se positionner dans une table et d'analyser un par un les enregistrements de celle-ci en fonction du code du client.

Créer une liste contenant tous les devis

L'instruction ci-dessous crée la liste de tous les devis classée par date :

ListeDevis = Liste "EnteteDevis" Classée par "Date"

Elle est à placer en début de script.

Pour chaque client, réaliser une boucle lisant l'intégralité de la liste des devis

Pour cela :
Positionnez la liste des devis sur le premier de la liste avec l'instruction LitPremier() :

```
LitPremier(ListeDevis)
```

Vous créez une boucle parcourant tous les devis grâce aux instructions "Faire...Boucler" et "LitSuivant()" :

```
LitPremier(ListeDevis)
Faire

  LitSuivant(ListeDevis)
```

Boucler

Il est nécessaire d'arrêter la boucle "Faire...Boucler" par une instruction "Arreter".
Vous arrêtez donc de parcourir la liste des devis lorsque vous êtes positionné sur le dernier grâce à l'instruction "Fin()" . Ce qui donne la ligne suivante :

```
LitPremier(ListeDevis)
Faire

  SI Fin(ListeDevis) ALORS ARRETER
  LitSuivant(ListeDevis)

Boucler
```

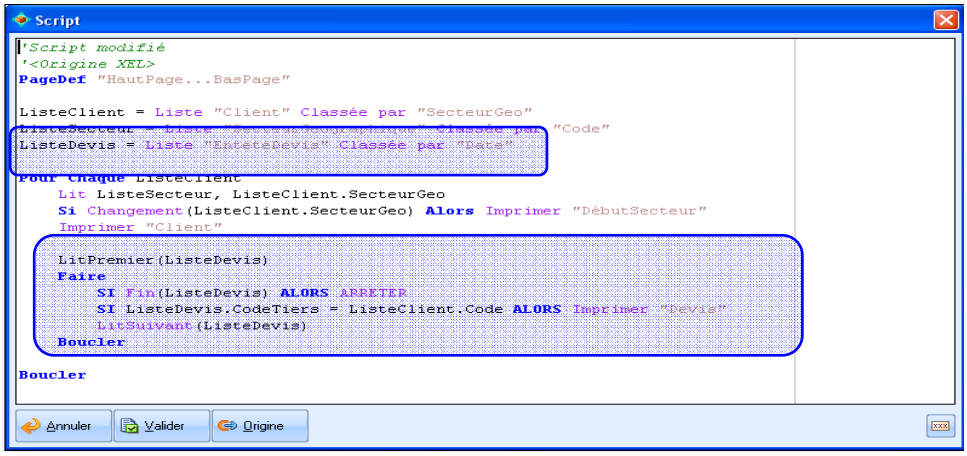
Imprimer un nouveau bloc "Devis" si le code du client correspond au client en cours.

Pour cela, il faut comparer la valeur du champ "Code" du client par rapport au champ "CodeTiers" du devis afin de n'imprimer le bloc "Devis" que dans le cas où ils sont égaux.

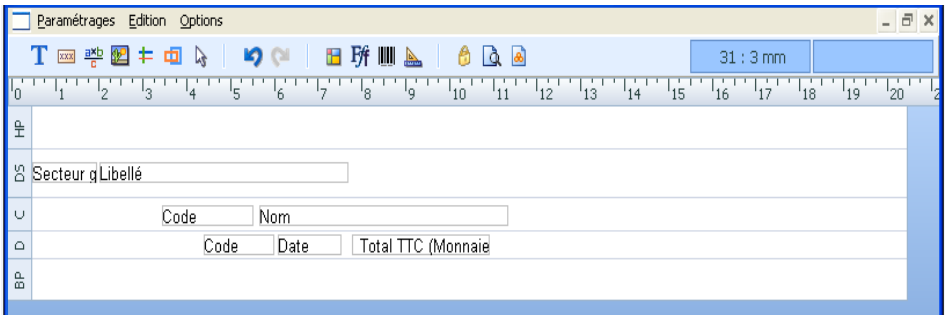
```
LitPremier(ListeDevis)
Faire

        SI Fin(ListeDevis) ALORS ARRETER
        Si ListeDevis.CodeTiers = ListeClient.Code ALORS Imprimer "Devis"
        LitSuivant(ListeDevis)

Boucler
```

Cette boucle est à placer après l'instruction imprimant le détail du client afin que la liste soit imprimée après.

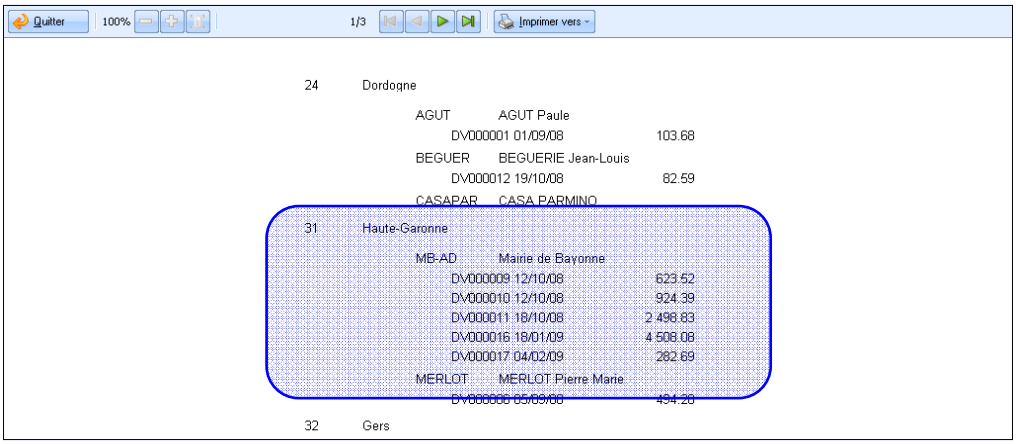
Le script précédent devient donc :



L'image ci-dessous vous donne un aperçu du résultat dans le paramétrage d'édition, et met en évidence le positionnement des rubriques souhaitées dans le bloc devis :



Le résultat dans l'aperçu écran de l'édition est le suivant :



Comme vous le constatez, les devis sont désormais regroupés par client.

Note : La boucle aurait également pu s'écrire :

Pour Chaque ListeDevis

Si ListeDevis.CodeClient=ListeClient.Code Alors Imprimer Devis

Boucler

ou si une relation existait entre la table devis et la table client

Listedevis = Liste "EnteteDevis" de ListeClient

Pour Chaque ListeDevis

Imprimer Devis

Boucler

h) Imprimer des cumuls par rupture

Vous souhaitez réaliser un cumul des montants TTC des devis clients par secteur géographique.

Vous devez pour cela créer un nouveau bloc d'impression s'imprimant après la liste des devis de tous les clients possédant le même secteur géographique.

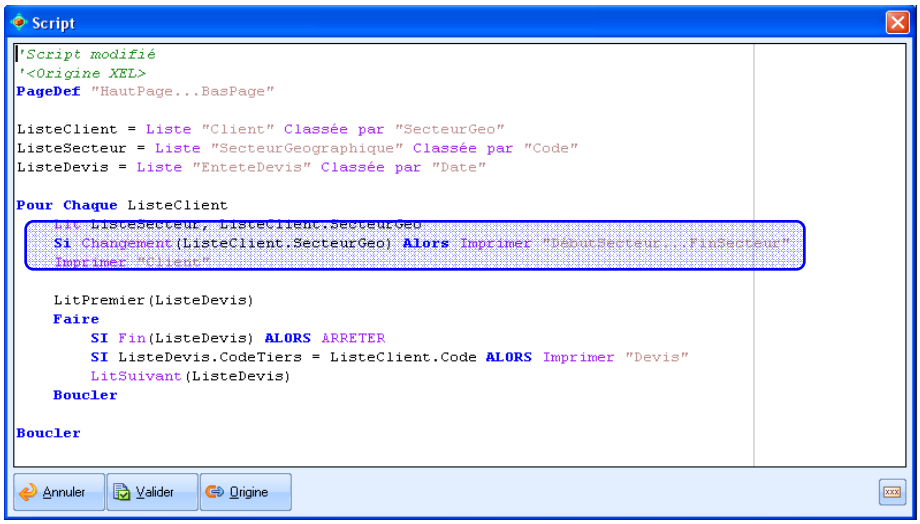
Cela nécessite au préalable d'avoir créé la rupture "Par secteur Géographique" et la liste des devis clients (voir les chapitres précédents).

Dans notre script, l'instruction :
Si Changement(ListeClient.SecteurGeo) Alors Imprimer "DébutSecteur"
Indique au programme d'imprimer le bloc "DébutSecteur" lors d'un changement de secteur

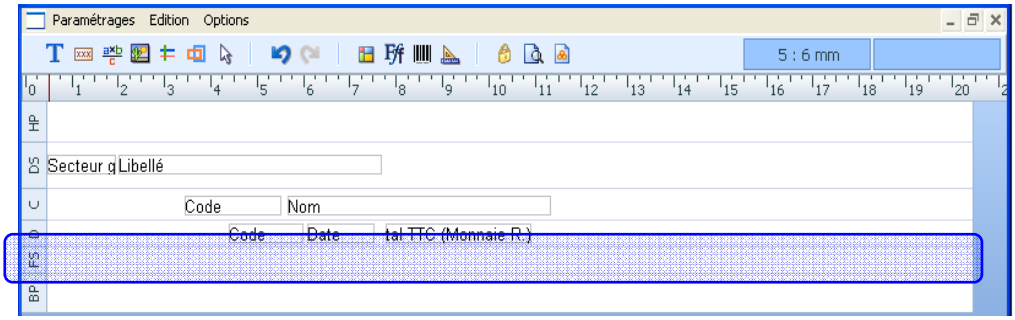
Il est possible de modifier cette instruction pour indiquer au programme d'imprimer un autre bloc à la fin, de la manière suivante :

Si Changement(ListeClient.SecteurGeo) ALORS Imprimer "DébutSecteur...FinSecteur"

Le script précédent devient :



A la validation, le bloc "FinSecteur" vient s'insérer automatiquement après le bloc "Devis"

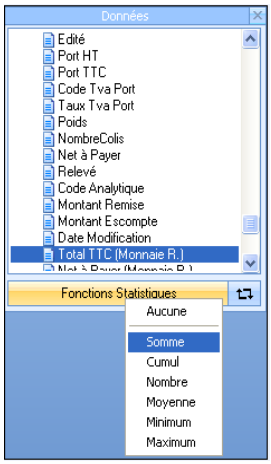


Il s'imprimera donc après le détail de tous les devis des clients du secteur en cours.

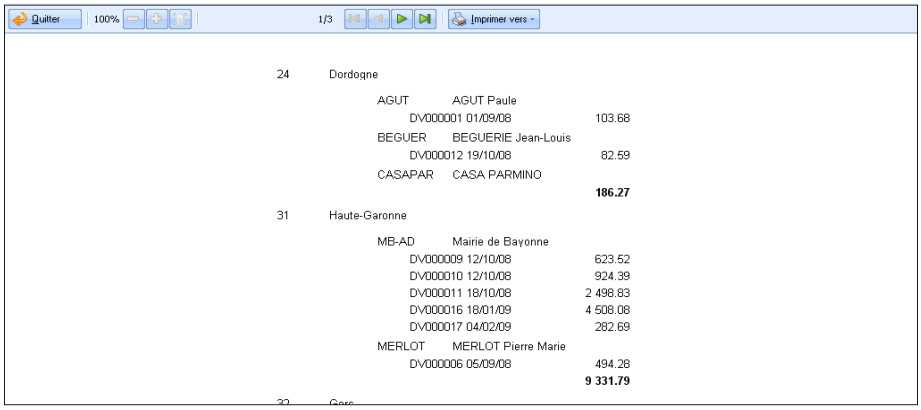
Il faut maintenant placer la somme de tous les montants TTC des devis imprimés depuis le dernier changement de secteur.

La méthode utilisée pour réaliser ce cumul est la création d'une formule de calcul via la fonction statistique "Somme" (Chapitre III.2).

Note :Pour rappel, la fonction "Somme" correspond exactement à notre besoin, réalisée sur le champ "Total TTC" des devis, elle réalisera la somme de ce champ depuis sa dernière impression.



Après l'ajout de cette fonction dans le bloc FinSecteur, la somme des totaux TTC imprimés depuis
le dernier secteur s'affichera sur l'édition (ici en gras) :



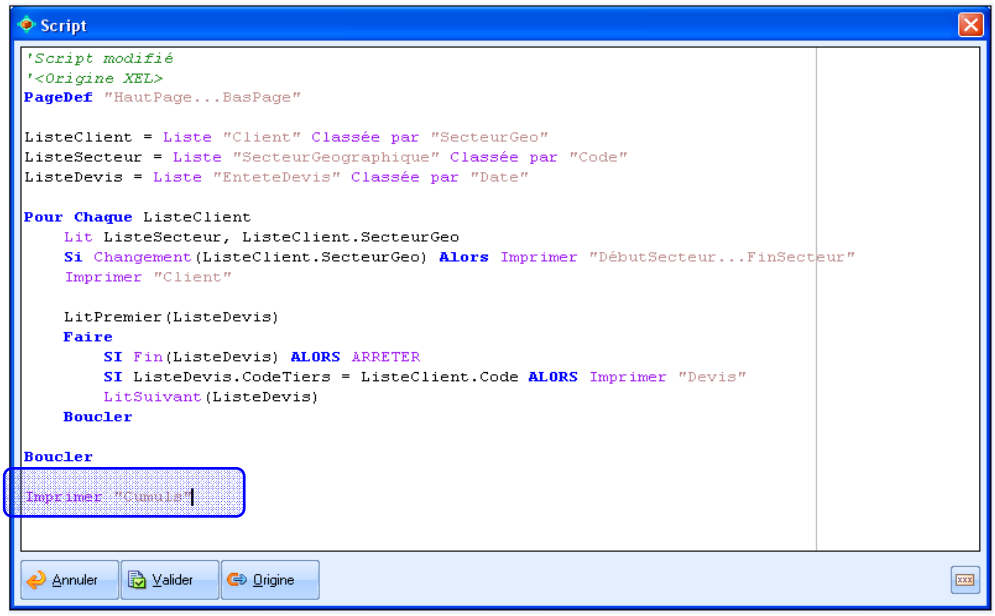
i) Réaliser un cumul en fin d'édition

Vous souhaitez réaliser un cumul des montants TTC de tous les devis clients imprimés.

Il vous est donc nécessaire d'insérer un nouveau bloc en fin d'édition, celui-ci contiendra le cumul de tous les montants TTC imprimés.

Pour cela, ajoutez une fonction "Imprimer(Bloc)" en fin de script.

Le script devient donc :

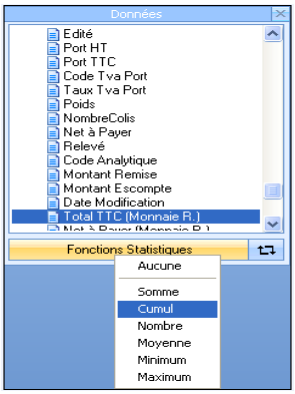


Ce qui a pour effet de rajouter le bloc "Cumul" dans le dessin graphique de l'édition, qui s'imprimera après tous les autres blocs.

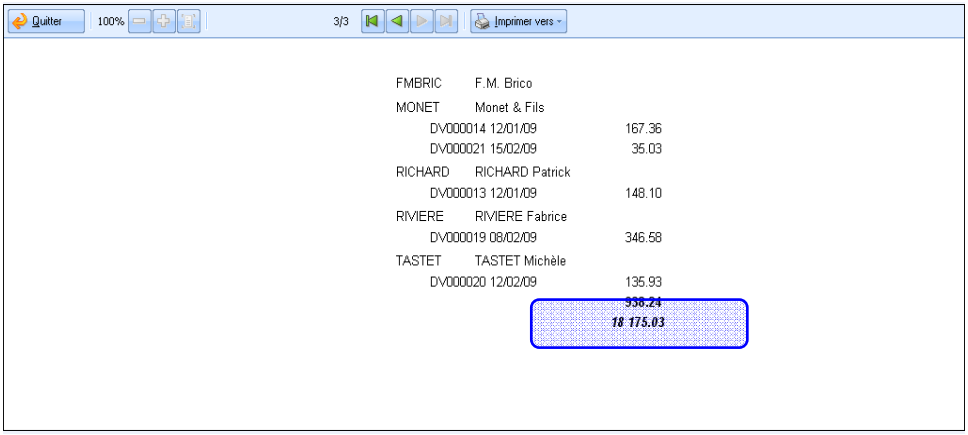
Il faut maintenant y placer la somme de tous les montants TTC des devis imprimés depuis le début de l'édition.

La méthode utilisée pour réaliser ce cumul est la création d'une formule de calcul via la fonction statistique "Cumul" (Chapitre III.2).

Note :Pour rappel, la fonction "Cumul" correspond exactement à notre besoin, réalisée sur le champ "Total TTC" des devis,elle réalisera le Cumul de ce champ de toute l'édition



En plaçant cette fonction dans le bloc "Cumul", le cumul de tous les montants TTC des devis apparaît bien en fin d'édition (ici en gras en italique):



j) Saisie et application de paramètres utilisateur.

L'édition précédente ne laisse aucun choix à l'utilisateur.

La procédure suivante permet d'offrir à l'utilisateur le choix du secteur géographique sur lequel sera ciblé l'édition :
- Apparition d'une fenêtre permettant de saisir le secteur géographique souhaité.
- Edition filtrée pour ne faire apparaître que celui sélectionné.

Faire apparaître la fenêtre de saisie du secteur géographique.

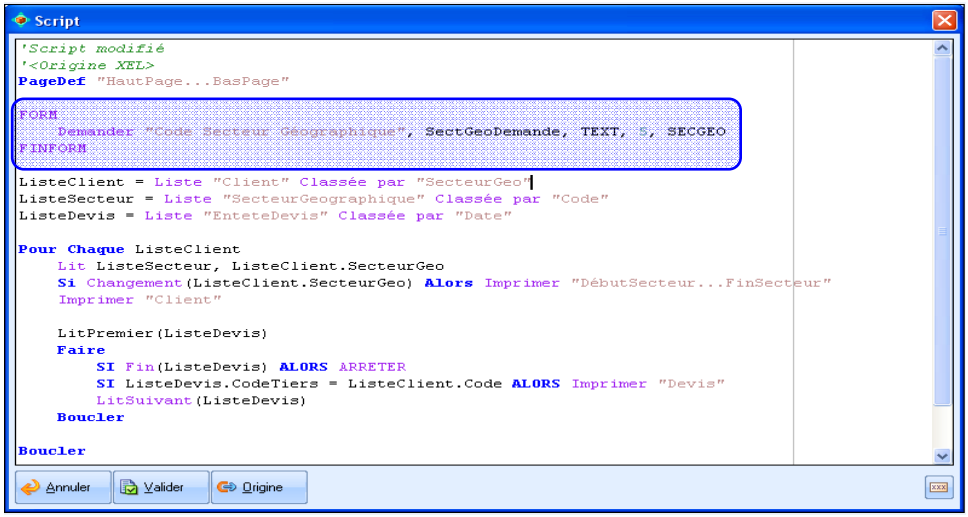
L'instruction FORM....FINFORM telle que documentée dans le chapitre II.H, permet l'affichage une fenêtre permettant à l'utilisateur de saisir des paramètres qui pourront ensuite être utilisés dans le script.

La syntaxe permettant d'afficher une demande intitulée "Code Secteur Géographique" dont la valeur saisie sera stockée dans la variable "SectGeoDemande" de type texte, de longueur 5 liée à la liste des secteurs géographique est la suivante :

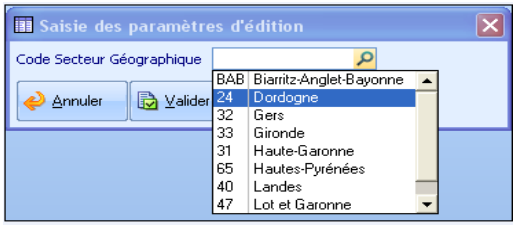
```
FORM
Demander "Code Secteur Géographique", SectGeoDemande, TEXT,5,SECGEO
FINFORM
```

Afin que la fenêtre de saisie s'affiche avant tout traitement, ces lignes sont à placer en début d'édition.

Le script précédent devient donc :



le résultat est l'apparition de la fenêtre ci-dessous lors du lancement de l'édition :



L'utilisateur peut ainsi choisir le secteur géographique sur lequel il souhaite filtrer l'édition.

Filtrer l'édition pour ne faire apparaître que celui sélectionné

Il est maintenant nécessaire de modifier le script pour n'imprimer que les clients dont le code du secteur géographique correspond à celui choisi par l'utilisateur.

L'instruction est la suivante :

Si SectGeoDemande = ListeClient.SecteurGeo ALORS
.....
FinSI

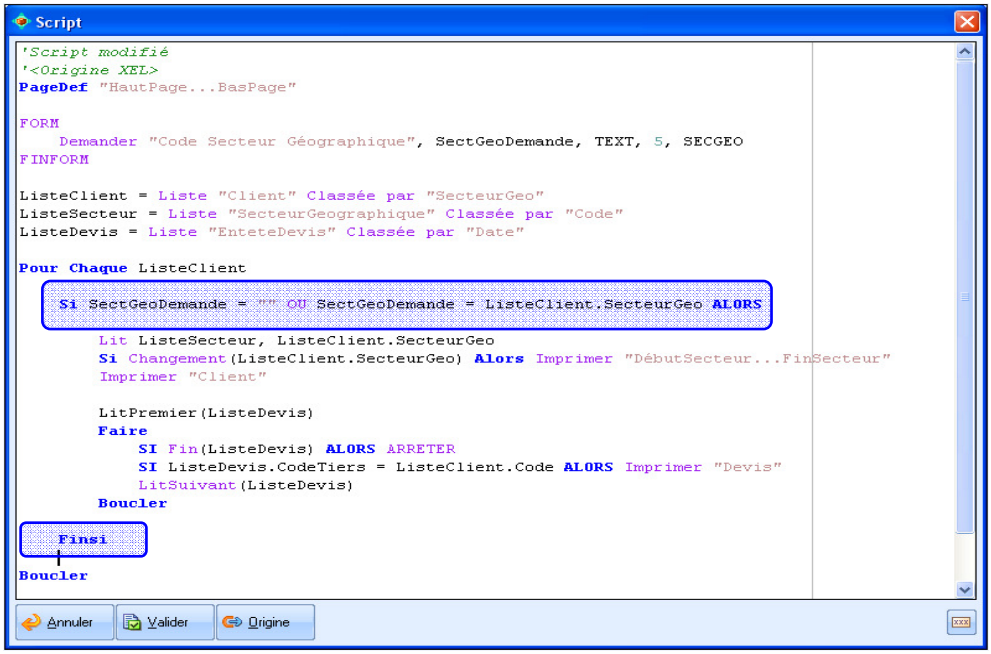
Elle est à ajouter au début de la boucle "Pour chaque Client" afin que la comparaison s'effectue sur chaque client.
Il est également nécessaire de gérer le cas où l'utilisateur ne choisit pas de secteur géographique,dans ce cas SectGeoDemande sera vide.

Il faudra donc réaliser les traitements si SectGeoDemande est vide ou si SectGeoDemande correspond au code du secteur géographique (champ SecteurGeo) du client en cours

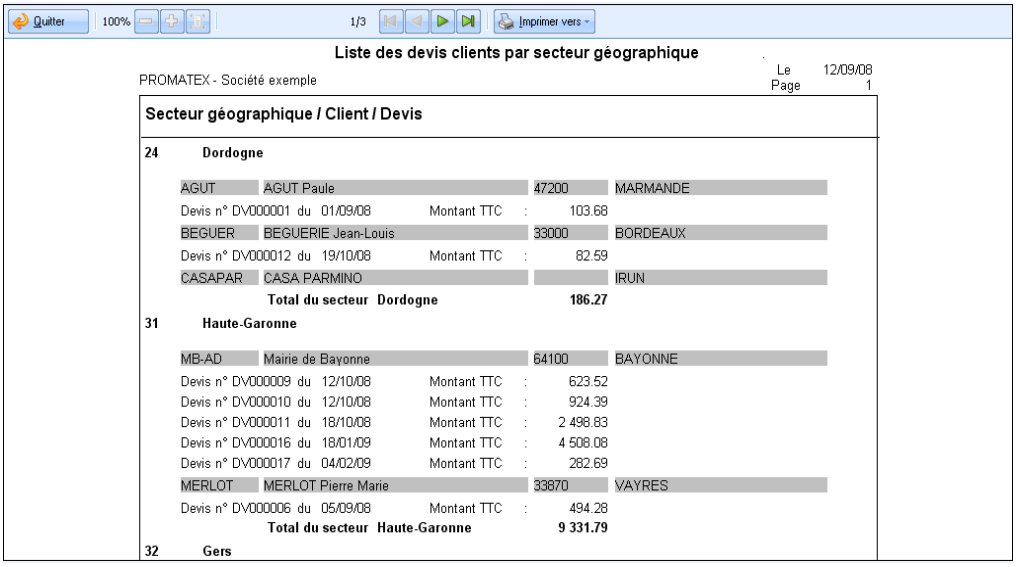
L'instruction précédente devient :

Si SectGeoDemande = "" OU SectGeoDemande = ListeClient.SecteurGeo ALORS
.....
FinSi

Le script devient :

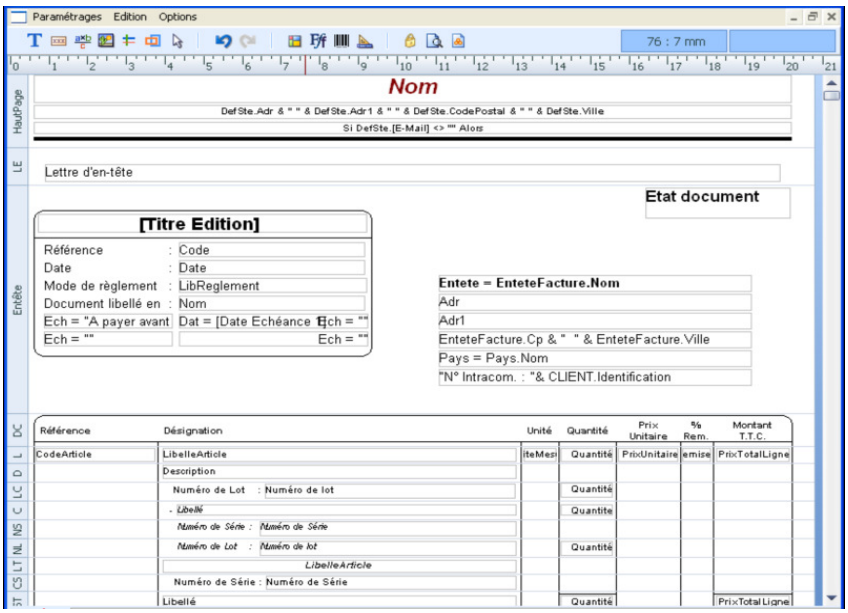


Après quelques modifications de présentation, l'édition finale pourrait ressembler à celle-ci :



2. Modification d'un état existant : La facture client

Dans cette partie, vous trouverez quelques méthodes vous aidant à la personnalisation d'un état existant, la facture client TTC standard.



a) Ajout d'une colonne manquante, le prix unitaire remisé

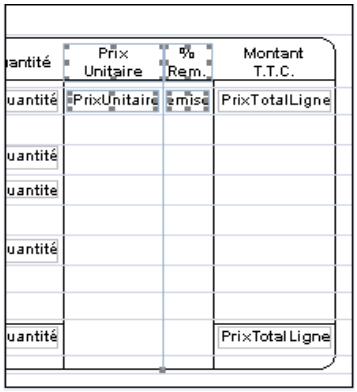
La demande formulée par utilisateur consiste à ajouter une nouvelle colonne contenant le prix unitaire remisé, elle remplacera la colonne "Prix Unitaire" et "% Remise".

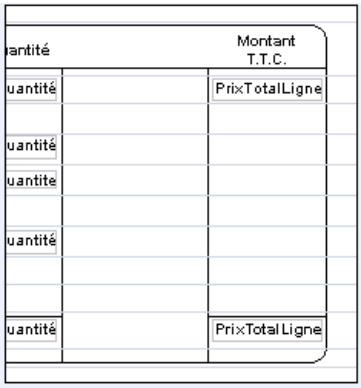
Remplacement des informations graphiques

La première étape est graphique, elle consiste à supprimer les deux colonnes "Prix unitaire" et ""%Rem"' .

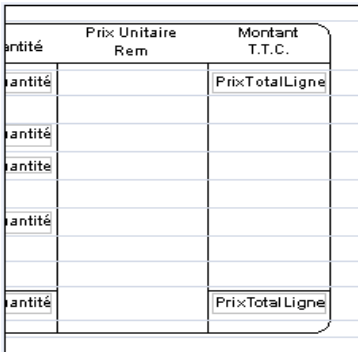
Vous sélectionnez les éléments que vous souhaitez supprimer, puis utilisez la touche "Suppr"

Note :Il est possible de sélectionner plusieurs éléments du paramétrage en maintenant la touche MAJ enfoncée





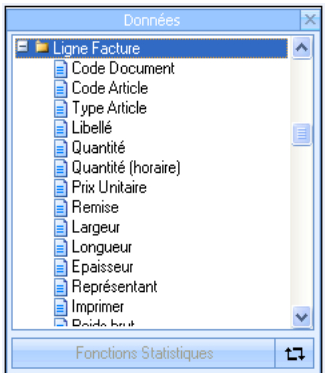
Vous cliquez sur le bouton TXT pour placer le libellé "Prix Unitaire Rem" sur la colonne ainsi libérée :



Il ne reste plus qu'à placer l'information "Prix Unitaire Remisé" dans cette colonne.

Placement des données demandées

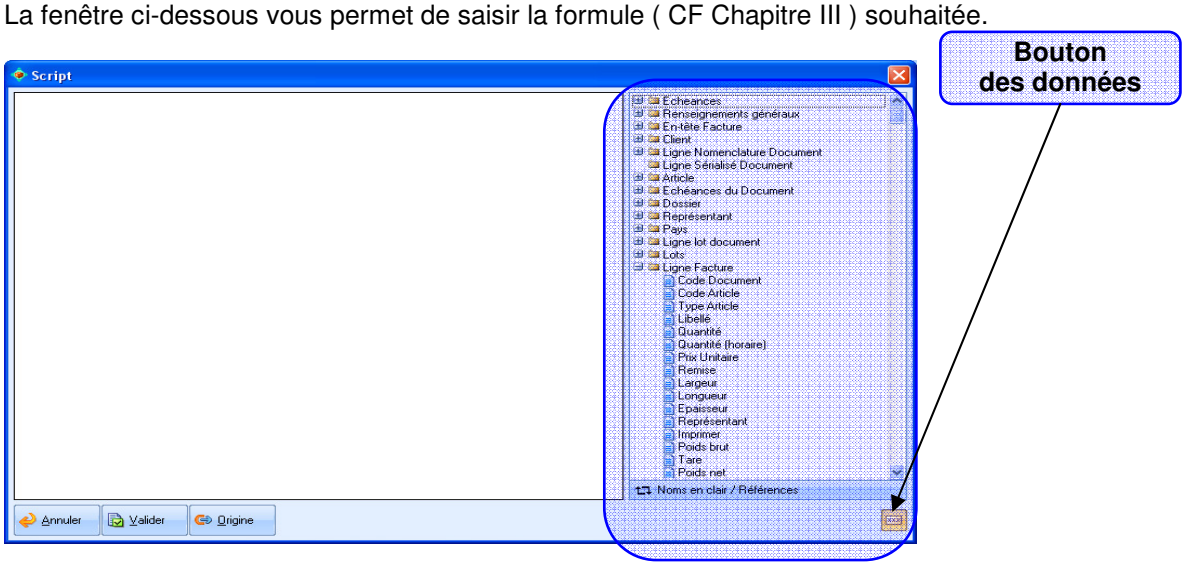
En consultant les données disponibles dans les lignes Factures, il apparaît que l'information demandée, le prix unitaire remisé n'est pas un champ directement exploitable de la base de données.



Il vous faut donc créer une formule de calcul dans laquelle nous diviserons le prix total ligne par la quantité.

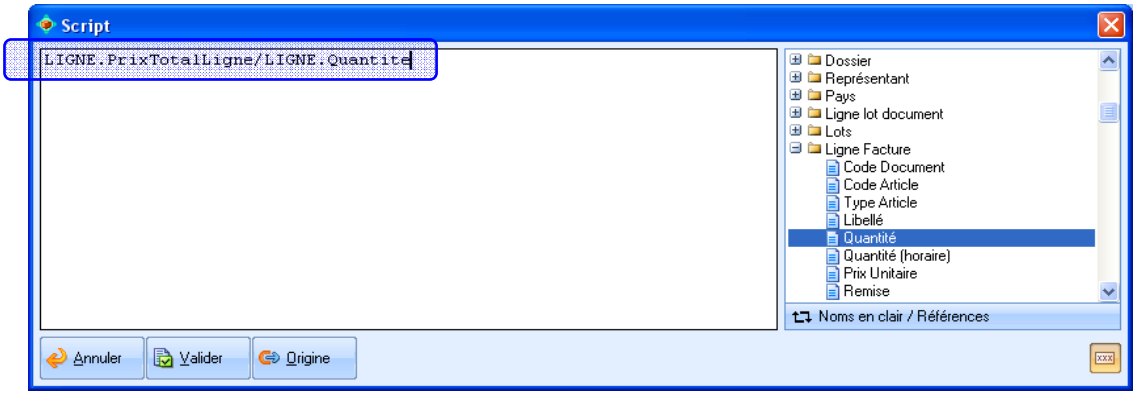
Cliquez sur le bouton de formule puis à l'endroit où vous souhaitez l'insérer

La fenêtre ci-dessous vous permet de saisir la formule (CF Chapitre III) souhaitée.

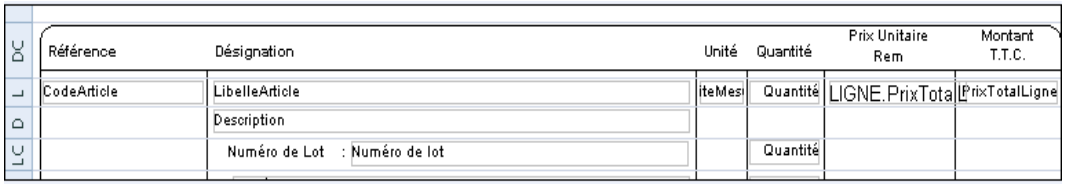


Cliquez sur le bouton des données afin d'avoir la possibilité de glisser les champs dans la formule.
Glissez les champs "Prix Total Ligne" et "Quantité" en précisant que vous divisez le prix total par la quantité.

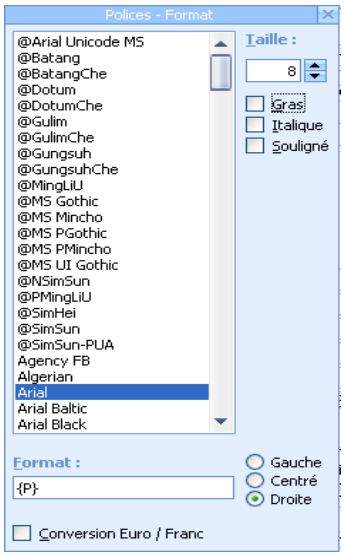
La formule devient :



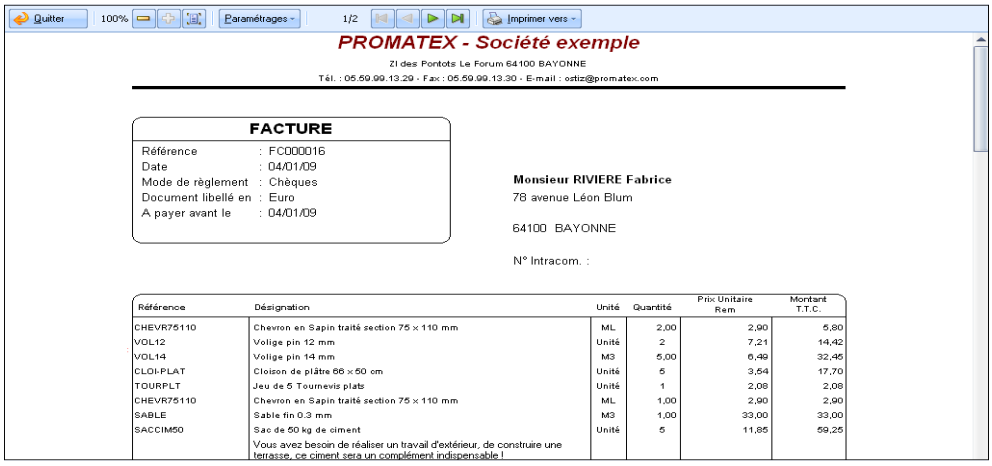
Une fois validée la formule apparaît à l'endroit où vous avez cliqué :



Cliquez sur le bouton "Police et format" afin de lui donner la même apparence que le montant TTC :



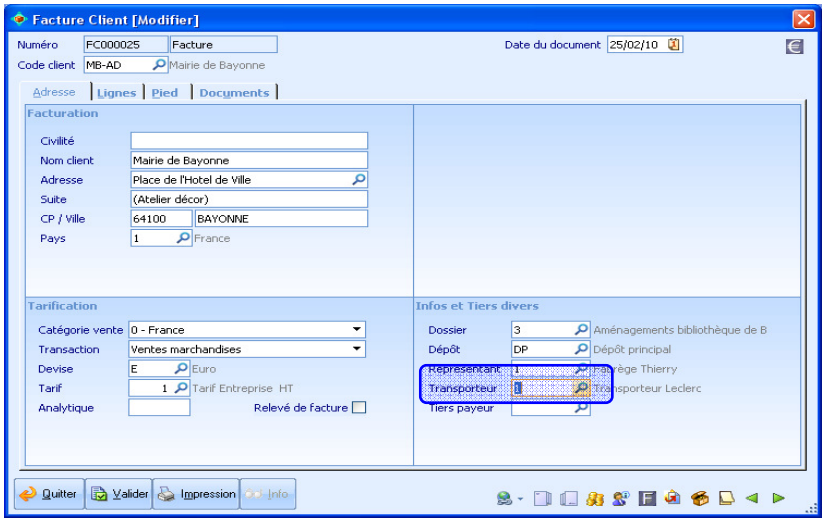
La modification est terminée, lors de l'impression de nouvelle facture, la formule de calcul du prix unitaire remisé apparaîtra à la place du prix unitaire et du % de remise, tel que ci-dessous :



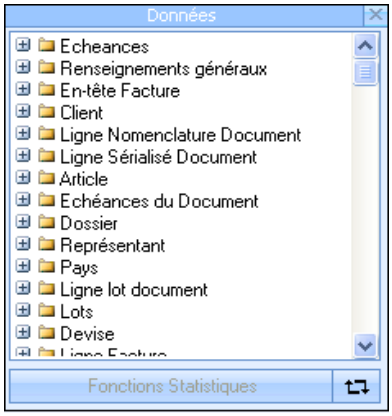
b) Ajout de rubrique non disponible à l'origine

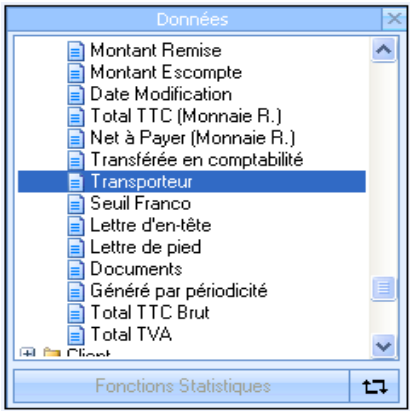
Lors de la réalisation d'un paramétrage d'édition personnalisé, il se peut que les informations demandées ne soient par directement disponibles dans les données.

Dans cet exemple, nous souhaitons imprimer le détail du transporteur saisi en entête de la facture.



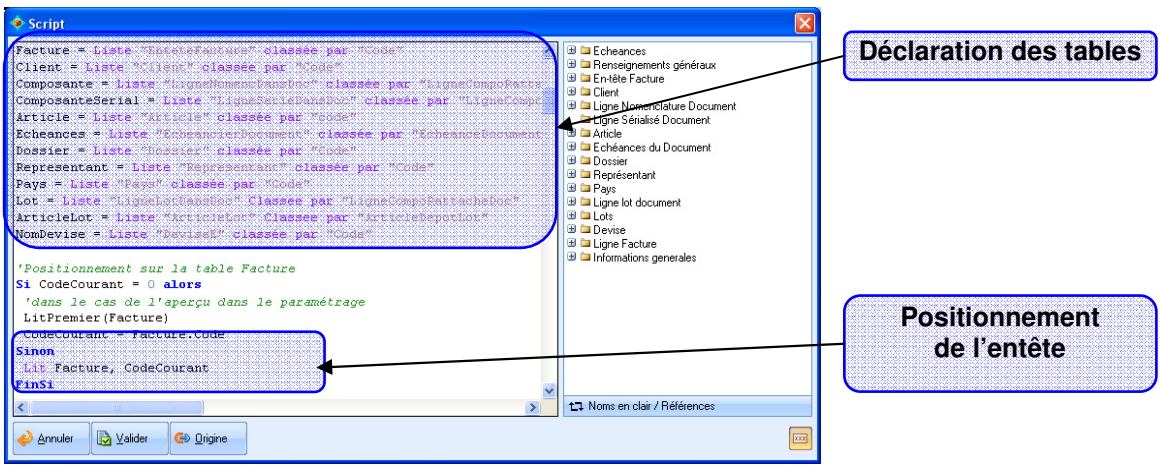
Aucune donnée 'Transporteur' n'est disponible pour l'édition, seul le champ 'Transporteur' est disponible sur l'entête du document :





Dès lors une modification du script est nécessaire, il faut créer une liste des transporteurs et positionner celle-ci par rapport à la valeur 'Transporteur' de l'entête.

En analysant le script d'origine, il faut déterminer à quel endroit insérer les instructions nécessaires.



Au début du script, la partie « Déclaration des tables » ouvre les données disponibles pour l'édition. Ajoutez les données des transporteurs par l'instruction :

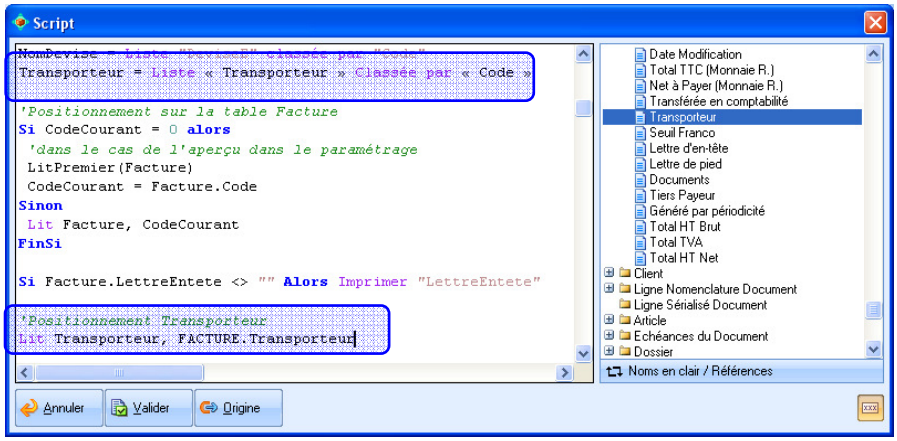
Transporteur = Liste « Transporteur » Classée par « Code »

L'instruction « Lit Facture, CodeCourant » positionne les informations de la liste entête sur la facture en cours, de fait, ce n'est qu'après cette instruction qu'il est possible d'exploiter les données présentes sur la facture en cours.

L'instruction de positionnement de la liste transporteur par rapport au champ 'Transporteur' de l'entête est :

Lit Transporteur, FACTURE.Transporteur

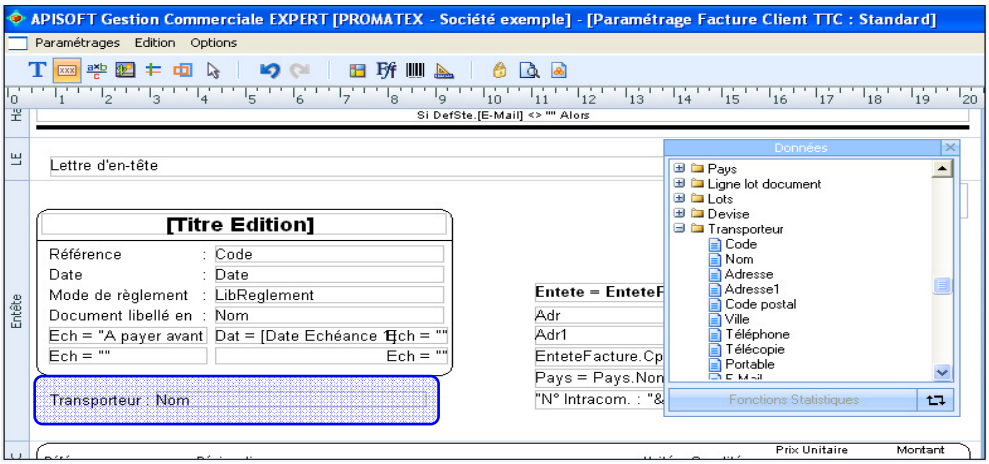
Après ajout des instructions, le script devient :



Note :Afin d'éviter des erreurs de frappe, il est recommandé de glisser/déposer le champ 'Transporteur' de 'entête facture' disponible dans les données (partie droite).

En validant ce script, les informations liées au transporteur indiqué dans l'entête du document seront disponibles dans toute l'édition.

Ici sur l'entête du document :



Un inconvénient à cette méthode : le texte « Transporteur » apparaîtra sur toutes les factures, même dans le cas où aucun transporteur n'est présent.

Imprimer les informations dans un nouveau bloc

Il est également possible de créer un nouveau bloc d'impression contenant toutes les informations du transporteur et d'ajouter une condition afin de n'imprimer ce nouveau bloc que si un transporteur est présent dans l'entête.

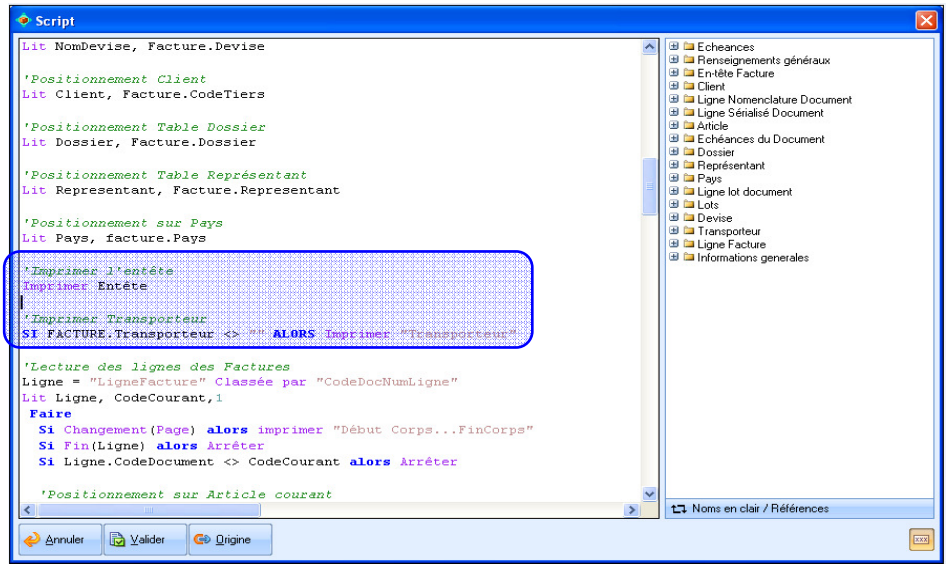
Cela donne l'instruction :

SI FACTURE.Transporteur <> "" ALORS Imprimer "Transporteur"

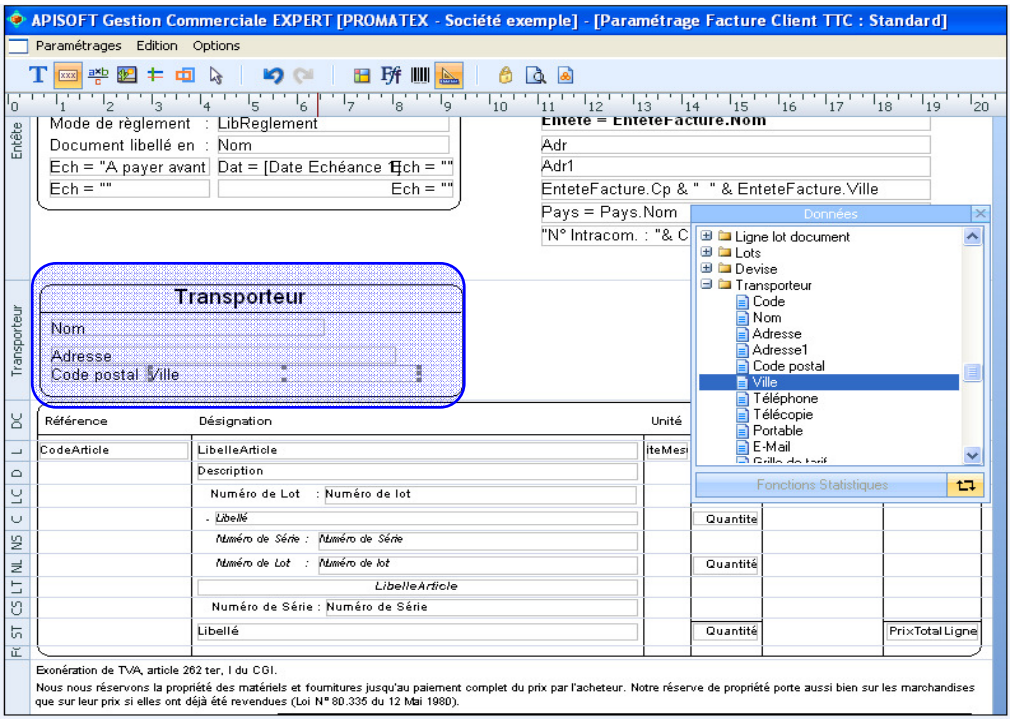
Le placement de cette instruction dans le script déterminera à quel endroit vous souhaitez imprimer les informations du transporteur.

Par exemple, vous souhaitez les faire apparaître juste après l'entête, il faut la placer après l'instruction « Imprimer Entête » :

Cela donne au niveau du script :



A la validation du script, le bloc « Transporteur » apparaît dans lequel vous placez les données que vous souhaitez faire apparaître :



c) Ajout d'un « Total Par » en fin de document

Dans ce type de cas, il est nécessaire de procéder à une analyse de la faisabilité de la demande suivant les informations dont vous disposez.
Dans notre exemple, il vous a été demandé de réaliser une totalisation des lignes de la facture imprimée par famille article en fin d'édition.

Phase d'analyse

Les données liées aux familles articles n'étant pas disponibles, il est premièrement nécessaire de créer une liste contenant toutes les familles articles pour faire un cumul « Par Famille »

En fin d'édition, pour chaque famille d'articles, il faut parcourir les lignes de la facture pour cumuler celles la concernant, or la famille de l'article n'est pas stockée sur la ligne mais sur l'article. Cela nécessite donc de récupérer cette information sur l'article.

Cela donnerait les phases suivantes :

Construire la liste des familles articles
Pour chaque famille

Remettre le cumul à 0

Pour chaque ligne de la facture

Lire le code famille de l'article de la ligne

Si le code famille correspond au code de la famille alors Cumuler le Total Ligne

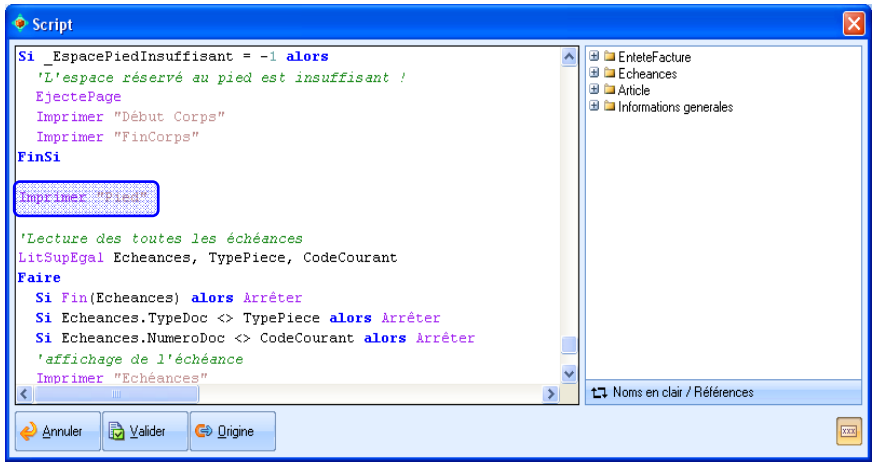
Boucler

Imprimer le cumul

Boucler

Phase d'écriture

En premier lieu, il faut déterminer dans quel endroit du script placer vos instructions.
Dans notre exemple, nous le placerons après le pied, il faut donc rechercher l'instruction « Imprimer Pied »

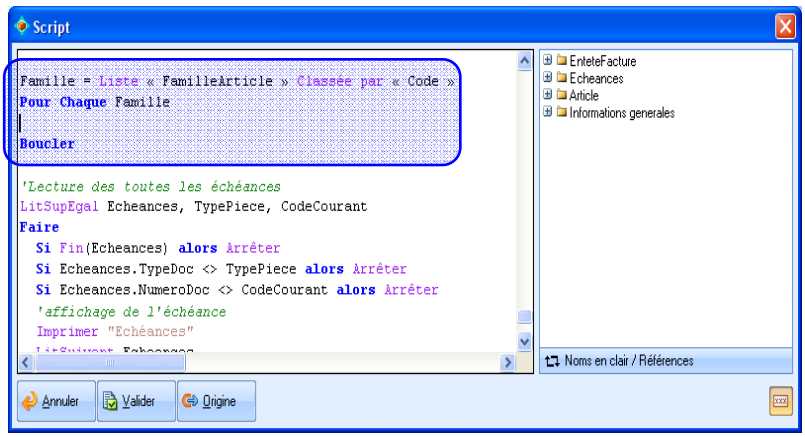


Construction de la boucle « Par Famille »

La construction de la boucle pour chaque famille article se réalise par l'instruction

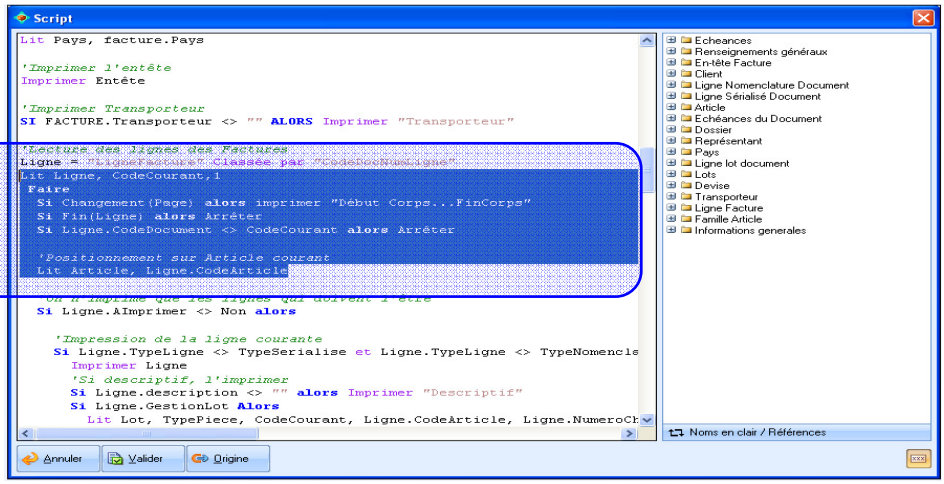
Famille = Liste « FamilleArticle » Classée par « Code »
Pour Chaque Famille

Boucler



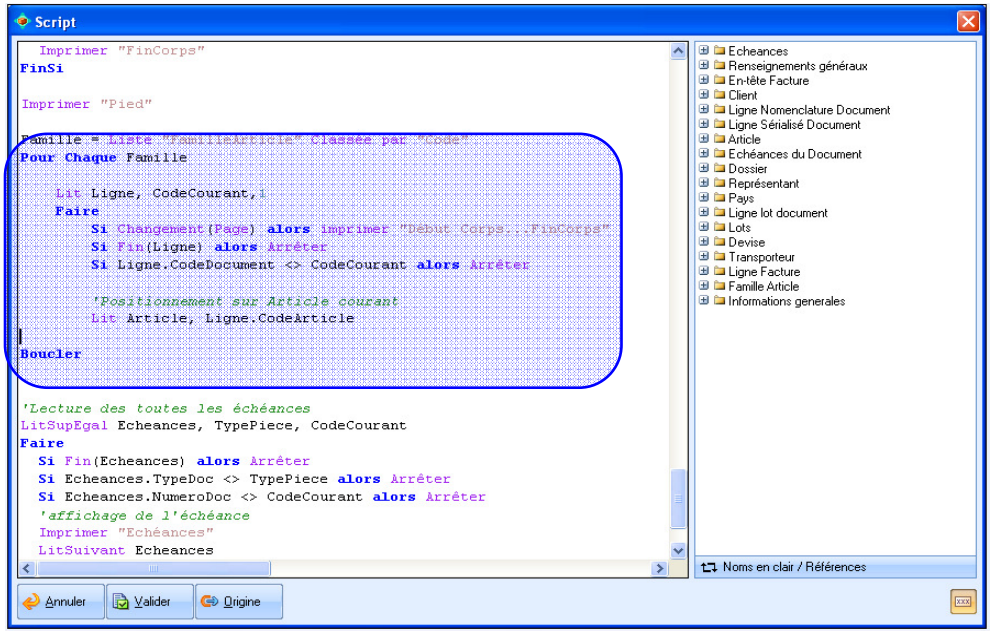
Construction de la boucle «Pour chaque ligne facture »

Le script d'origine des factures client possède une telle boucle.
En début d'édition, le script positionne les lignes facture sur la première de celle-ci et réalise une boucle jusqu'à ce que soit le code facture ait changé soit la liste est arrivée à la fin.



Vous remarquez que l'instruction suivante lit l'article suivant le code article de la ligne ce qui vousest nécessaire afin de lire le code famille de l'article.

Par un copier/coller, vous récupérez les lignes qui vous intéressent afin de les placer dans votre boucle « Pour chaque famille »



Il ne reste plus qu'a passer à la ligne suivante avec l'instruction « LitSuivant Ligne » et à fermer la boucle « pour chaque ligne » avec l'instruction « Boucler » et supprimer l'instruction liée au changement de page qui n'est pas utile pour notre cumul.

Voici la boucle finalisée :

```
Si _EspacePiedInsuffisant = -1 alors
  'L'espace réservé au pied est insuffisant !
  EjectePage
  Imprimer "Début Corps"
  Imprimer "FinCorps"
FinSi

Imprimer "Pied"

Famille = Liste "FamilleArticle" Classée par "Code"
Pour Chaque Famille

    Lit Ligne, CodeCourant,1
    Faire
        Si Fin(Ligne) alors Arrêter
        Si Ligne.CodeDocument <> CodeCourant alors Arrêter

        'Positionnement sur Article courant
        Lit Article, Ligne.CodeArticle


        LitSuivant Ligne
    Boucler

Boucler
```

Construction du cumul pour la famille en cours

En premier lieu, il est nécessaire de déclarer le cumul en tant que variable globale afin d'être utilisable sur le paramétrage d'édition.

L'instruction est :
Global CumulFamille

```
  LitSuivant Ligne
Boucler

Si _EspacePiedInsuffisant = -1 alors
  'L'espace réservé au pied est insuffisant !
  EjectePage
  Imprimer "Début Corps"
  Imprimer "FinCorps"
FinSi

Imprimer "Pied"

Global CumulFamille
Famille = Liste "FamilleArticle" Classée par "Code"
Pour Chaque Famille

    Lit Ligne, CodeCourant,1
    Faire
        Si Fin(Ligne) alors Arrêter
        Si Ligne.CodeDocument <> CodeCourant alors Arrêter

        'Positionnement sur Article courant
        Lit Article, Ligne.CodeArticle


        LitSuivant Ligne
    Boucler
```

A chaque changement de famille, il est nécessaire de commencer le cumul à 0 avant de parcourir les lignes :
CumulFamille = 0

```
Imprimer "Pied"

Global CumulFamille
Famille = Liste "FamilleArticle" Classée par "Code"
Pour Chaque Famille

    CumulFamille = 0

    Lit Ligne, CodeCourant,1
    Faire
        Si Fin(Ligne) alors Arrêter
        Si Ligne.CodeDocument <> CodeCourant alors Arrêter

        'Positionnement sur Article courant
        Lit Article, Ligne.CodeArticle


        LitSuivant Ligne
    Boucler


Boucler
```

Lors de la lecture des lignes de facture, après avoir positionné l'article par rapport au code article de la ligne, ajoutez le total ligne au Cumulfamille dans le cas où le code de la famille de l'article est identique à celui de la famille en cours :

SI ARTICLE.Famille = FAMILLE.Code ALORS

CumulFamille = CumulFamille + [LIGNE.MonnaieRef_PrixTotalLigne]

FINSI

```
    CumulFamille = 0

    Lit Ligne, CodeCourant,1
    Faire
        Si Fin(Ligne) alors Arrêter
        Si Ligne.CodeDocument <> CodeCourant alors Arrêter

        'Positionnement sur Article courant
        Lit Article, Ligne.CodeArticle

        SI ARTICLE.Famille = FAMILLE.Code ALORS

          CumulFamille = CumulFamille + [LIGNE.MonnaieRef_PrixTotalLigne]

        Finsi

        LitSuivant Ligne
    Boucler


Boucler
```

Note :La saisie de cette formule se réalise facilement en faisant glisser/déposer les champs souhaités depuis la partie droite de la fenêtre.

Impression du cumul et mise en page

Une fois les lignes parcourues, vous ne souhaitez imprimer le cumul que s'il est différent de 0 soit l'instruction suivante :

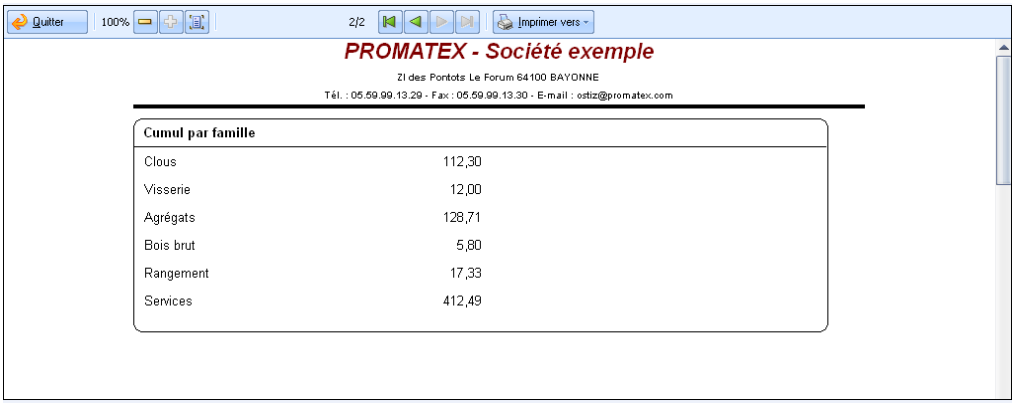
Si CumulFamille <> 0 Alors Imprimer CumulParFamille

Et ajouter un bloc d'impression d'entête de cumul et de fin de cumul.

Le script finalisé se présente comme suit :

```
Global CumulFamille
Famille = Liste "FamilleArticle" Classée par "Code"
Imprimer "DebutFamille"
Pour Chaque Famille
    CumulFamille = 0
    Lit Ligne, CodeCourant,1
    Faire
        Si Fin(Ligne) alors Arrêter
        Si Ligne.CodeDocument <> CodeCourant alors Arrêter

        'Positionnement sur Article courant
        Lit Article, Ligne.CodeArticle

        SI ARTICLE.Famille = FAMILLE.Code ALORS

          CumulFamille = CumulFamille + [LIGNE.MonnaieRef_PrixTotalLigne]

        Finsi

        LitSuivant Ligne
    Boucler
    Si CumulFamille <> 0 Alors Imprimer CumulParFamille
Boucler
Imprimer "FinFamille"

'Lecture des toutes les échéances
LitSupEgal Echeances, TypePiece, CodeCourant
```

Après mise en page, un cumul de ce type apparaîtra en fin d'édition de facture :



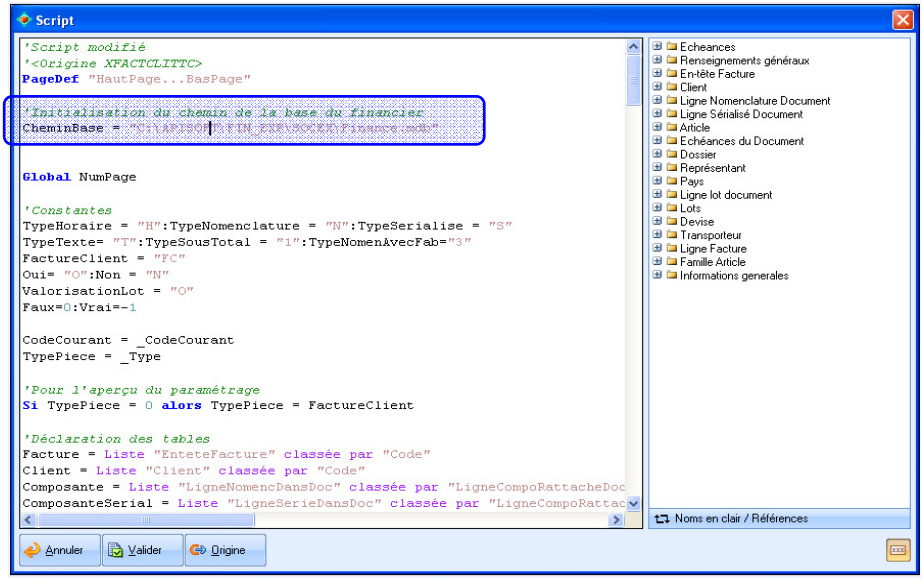
d) Accéder aux données financières de la facture

Ce paramétrage permet de modifier le script à l'aide d'une liaison de base, afin de récupérer des informations du logiciel Financier.

Le but étant d'imprimer la liste des échéances de la facture avec leur reste dû.

Modification de script, base externe

La première modification consiste à mentionner le chemin d'accès aux données de Financier à l'aide de la syntaxe '**CheminBase**'.

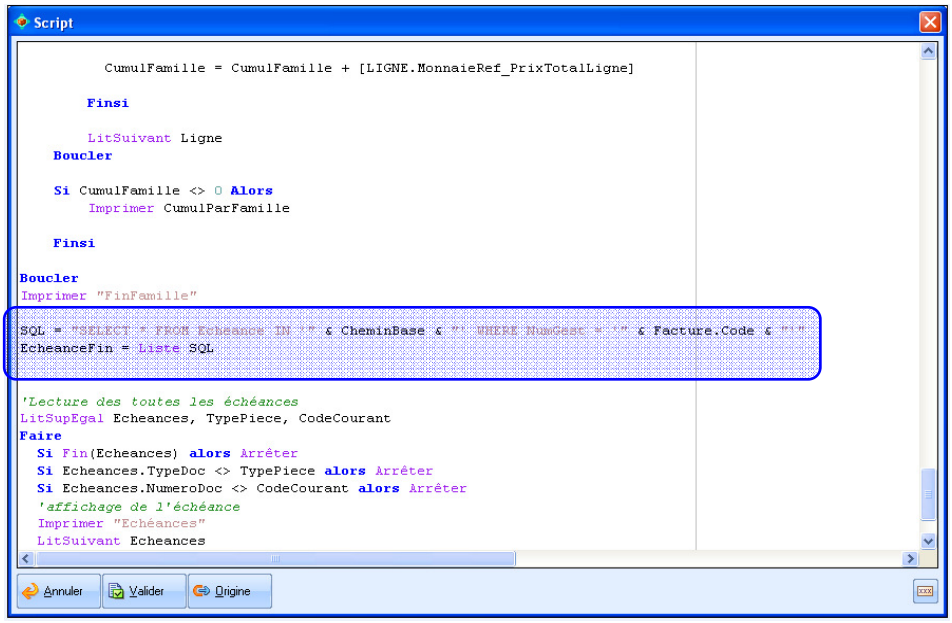


Note :Le chemin d'accès doit correspondre à la base finance.mdb du dossier lié avec le dossier de gestion. Il est donc à modifier en conséquence.

Déclaration de la table Échéance

La gestion de la base de donnée Finance permet de récupérer n'importe quelle information de celle-ci mais se doit d'être saisie en langage SQL.

Vous souhaitez récupérer les informations de la table Echéance lorsqu'elles proviennent de la facture en cours d'impression, utilisez la syntaxe suivante :



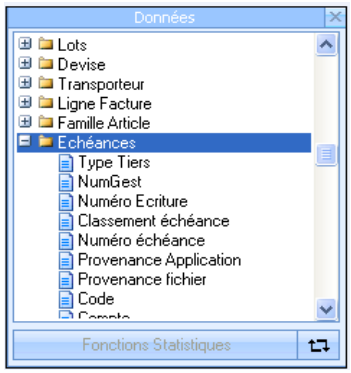
La décomposition de la requête est la suivante :

Select * FROM Echeance :
Indique d'accéder à tous les champs de la table Echeance

IN '" & CheminBase & "' :
Spécifie d'ouvrir la table Echéance dans la base de données spécifiée par CheminBase.

WHERE NumGest = '" & Facture.Code & "'"
La condition est que le champ NumGest soit identique au code de la facture.

A la validation du script, les champs correspondants aux échéances du financier apparaissent dans les données.



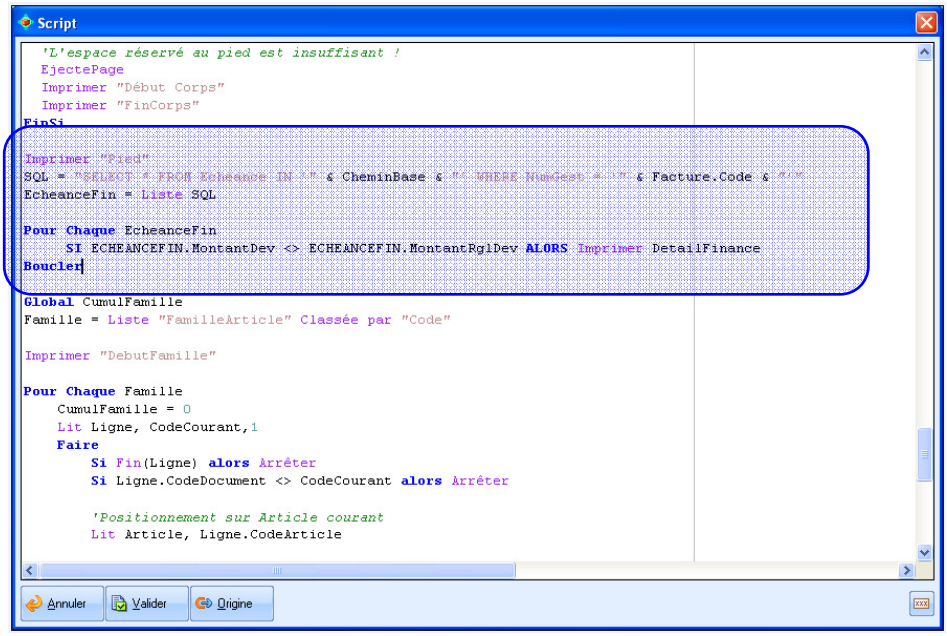
Validation et mise en place des données

Comme vu précédemment, le placement dans le script de la déclaration de la table Echeance est primordial car ce n'est qu'après cette exécution de ces instructions que les données seront exploitables.

Le seul impératif est que la table d'entête facture soit positionnée sur la facture que vous souhaiter imprimer donc après l'instruction « Lit Facture, CodeCourant »

Pour des questions de présentation, la liste des échéances dues sera imprimée après le pied

Afin de n'imprimer que les échéances dues, une condition est posée entre le montant de l'échéance et le montant réglé, d'où le script :



Le bloc DetailFinance est ainsi créé, il est possible d'insérer une formule de calcul calculant le reste dû de l'échéance soit :

« *ECHEANCEFIN.MontantDev - ECHEANCEFIN.MontantRglDev* »

L'édition de la facture pourrait donner ce résultat :

Référence	Désignation	Unité	Quantité	Prix Unitaire Rem	Montant T.T.C.
	Travaux cour intérieure				
GRAVE	Grave à Beton	M3	XXXXXXX	29,35	29,35
SABLE	Sable fin 0.3 mm	M3	XXXXXXX	33,00	33,00
SABLEM	Sable moyen 0.6	M3	XXXXXXX	33,18	66,36
MO	Main d'oeuvre	Unité	XXXXXXX	31,73	190,38
	Sous-Total		XXXXXXX		319,09
	Travaux dans dépendance				
TAB100	Tablette acier laqué blanc 100 x 40 cm	Unité	XXXXXXX	10,11	10,11
CL100	Clou 100 mm à tête plate	KG	XXXXXXX	11,23	112,30
	Clou d'une longueur de **100 mm** idéal pour tous vos travaux de fixations sur du bois.				
COR200	Cornière acier laqué blanc H200 cm	Unité	XXXXXXX	3,61	7,22
KITVIS	Kit visserie tablette	Unité	XXXXXXX	1,20	12,00
CHEVR75110	Chevron en Sapin traité section 75 x 110 mm	ML	XXXXXXX	2,90	5,80
MO	Main d'oeuvre	Unité	XXXXXXX	31,73	222,11
	Sous-Total		XXXXXXX		369,54

Exonération de TVA, article 262 ter, I du CGI.

Nous nous réservons la propriété des matériels et fournitures jusqu'au paiement complet du prix par l'acheteur. Notre réserve de propriété porte aussi bien sur les marchandises que sur leur prix si elles ont déjà été revendues (Loi N° 80.335 du 12 Mai 1980).

Total Brut TTC		: 000 000 000 000,00	Total HT Net	:	
Remise TTC	2,00 %	: 13,77	TVA	:	000 000 000 000 000,00
Escompte TTC		:	Total TTC	:	680,86
Frais de Port TTC		: 3,00	Acompte	:	
Frais de Facturation TTC		: 3,00	**Net à Payer**	**:**	**680,86 EUR**

(Montant indicatif en Francs : 4 466,15)

Taux des pénalités exigibles à compter du lendemain de la date de réglement en l'absence de paiement : 0, %

XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

Echeance dû : 680,86

S.A. au capital de 1 000 000 Euros - N° Siret 11256707400044 - Code APE 312A - N° TVA Intracom. FR42354951901 - RCS

V. Annexes

Nous vous rappelons que le support de formation de niveau III traitant de la syntaxe des scripts d'édition est également disponible dans votre Extranet API.