EPSITEC SA
Mouette 5 CH – 1092 Belmont
Tél/Fax ++41 (0) 21 728 44 83

Crésus - Facturation : Les calculs

Normalement, les différentes rubriques d'une présentation sont remplies par l'utilisateur. Un calcul permet de remplir une rubrique ou une variable globale automatiquement, en fonction des contenus d'autres rubriques.

Les calculs

Par exemple, les rubriques Prix et Quantité sont remplies normalement par l'utilisateur. En revanche, la rubrique Total contient un calcul: @Prix*@Quantité. Elle apparaît avec un fond gris, et ne peut pas être remplie manuellement. La rubrique Total est enregistrée dans la base de données comme toutes les autres. Elle peut donc être utilisée pour un tri ou une extraction dans un accès.

Les calculs temporaires

Un calcul temporaire ne remplit pas une rubrique normale. Son résultat apparaît, mais n'est pas copié dans une rubrique existante. Le résultat d'un calcul temporaire ne peut donc pas être utilisé pour un tri ou une extraction dans un accès, ni être réutilisé en tant que valeur intermédiaire dans d'autres calculs.

Les calculs temporaires sont principalement utilisés pour l'impression. Si, par exemple, vous voulez faire figurer le prénom et le nom d'un client sur une facture, vous pouvez dessiner les deux rubriques Prénom et Nom côte à côte. Mais vous obtiendrez un espacement fixe entre les deux, même si le prénom est très court. Dans ce cas, il est préférable de remplacer les deux rectangles des rubriques par un seul contenant le calcul temporaire suivant : @Prénom+" "+@Nom

Ce qui signifie: le contenu de la rubrique Prénom, suivi de la chaîne de caractères entre guillemets (un espace), suivi du contenu de la rubrique Nom.

Remarque : Les présentations pour l'imprimante ne peuvent contenir que des calculs temporaires ou des calculs sur des variables globales. Elles peuvent bien entendu utiliser le résultat d'un calcul normal contenu dans une rubrique, mais pas effectuer un calcul normal.

La syntaxe

Un calcul doit respecter une syntaxe très rigoureuse. Un nom de rubrique est toujours précédé du signe @. Une rubrique tableau peut être suivie d'un nombre entre crochets, pour indiquer la ligne concernée (la première a le numéro zéro).

Exemples : @Prénom

@RabaisQuantité[2]

L'opérateur d'addition (par exemple) représenté par le signe + nécessite deux opérandes, placées de part et d'autre du signe. Ces opérandes peuvent être des constantes ou des rubriques.

Exemples : 12 + 45 addition de deux constantes

@Quantité + 10 addition d'une rubrique avec une constante

@Quantité + @Retour addition de deux rubriques

Il est possible d'additionner deux nombres (constantes, rubriques numériques entières ou fractionnaires, etc.), mais pas un nombre avec une chaîne. Il est également possible d'additionner deux chaînes; le résultat de l'addition sera alors la concaténation des deux chaînes.

Il faut également prendre garde au type de la rubrique contenant le calcul. Par exemple, la rubrique alphanumérique Désignation ne peut pas contenir le calcul 12+45 ! Ceci est également valable pour les calculs temporaires.

Les espaces et les fins de lignes entre les opérateurs permettent d'améliorer la lisibilité. Ils n'ont pas d'influence sur le résultat du calcul. // débute un commentaire dans un calcul qui se terminera au bout de la ligne.

Remarque : L'espion permet de comprendre l'interaction des différents calculs entre eux.

La situation des calculs

Les calculs se placent selon le moment où ils sont exécutés.

Fichier

Le Menu *Options*, *Définitions*, *Réglages généraux* permet de préparer des calculs qui seront exécutés à l'ouverture du fichier et/ou à sa fermeture, juste avant son enregistrement. C'est là que l'on placera les actions d'importations automatique par exemple.

Fiche

Dans les paramètres des écrans de saisie il est aussi possible de placer des calculs qui seront exécutés lors de l'entrée dans une fiche existante, lors de la création d'une nouvelle fiche, lors de la suppression d'une fiche, lors de la duplication d'une fiche. Attention, la modification d'une fiche provoque les calculs de suppression de la fiche suivi des calculs de validation.

Ecran

Les calculs dessinés dans l'écran de saisie seront actionnés individuellement pour la fiche active, à chaque fois que l'on retouche cette fiche.

Si un calcul est provoqué par une autre base, par exemple comme l'ajout de stock, ce sont les calculs de l'écran de saisie qui contient les calculs standard qui sont actionnés. C'est pourquoi il est important que l'écran de saisie que l'on utilise contienne les calculs standard.

Ces calculs sont aussi actionnés lors de recalcule à travers un accès ou à travers l'ensemble des fiches d'une base de données.

Présentation pour imprimante

Les calculs sont actionnés lors de la visualisation ou lors de l'impression. Ces présentations ne peuvent contenir que des calculs temporaires.

Les opérations

+ - * /

No comment, si ce n'est que l'on peut appondre 2 chaînes de caractères et que l'on ne peut pas appondre un nombre à du texte. Il faut transformer ce nombre en chaîne par l'opérateur CHAINE()

:=

Assignation, utilisée uniquement pour les objets de type bouton action. La rubrique réceptrice du calcul peut être dans la base de données en cours, ou dans une autre base en fonction d'une relation. Dans ce cas, le calcul ne sera effectué qu'au moment où la fiche sera validée. Si les modifications sont annulées, le calcul n'aura pas lieu.

Le nom de la rubrique réceptrice peut être suivi d'un index entre crochets, si la rubrique est de type tableau. Cet index spécifie la ligne concernée dans le tableau. La première ligne correspond à l'index zéro. L'index peut être une constante, ou un calcul aussi complexe que nécessaire.

Syntaxe : rubrique := calcul

rubrique[index] := calcul

Exemples : @Rabais := 0

@RefClient.Info := @RefClient.Info+@Total

@Liste[2] := @Total

@Liste[COMPTE(@Liste)] := @Total

@RefClient.Liste[@Index] := @Rabais

Le calcul d'un bouton action peut contenir plusieurs opérations d'assignation. Il sera ainsi possible d'effectuer plusieurs calculs en cliquant dans un seul bouton. Les différents calculs doivent être sur des lignes différentes.

Lorsque des calculs ne doivent être exécutés que dans certaines conditions, il est possible d'utiliser les instructions #SI, #SINON et #FIN pour séparer différents groupes de calculs. On peut également faire une série d'assignation dans une boucle, en utilisant les instructions #TANTQUE, #REPETE

: :=
Assignation comme ci-dessus, mais dont le résultat est assigné lorsque la fiche est validée.

MIN( )
Cherche la valeur minimale dans une liste de nombres. En donnant le nom d'une rubrique tableau, ce sont toutes les valeurs contenues qui sont prises en compte.
Syntaxe : MIN(liste1; liste2; etc.)
Exemples : MIN(12; 45; -3; 28) donne -3
MIN(@Prix)

MAX( )
Cherche la valeur maximale dans une liste de nombres.
Syntaxe : MAX(liste1; liste2; etc.)
Exemples : MAX(12; 45; -3; 28) donne 45
MAX(@Prix) où prix est une rubrique tableau

SOMME( )
Calcule la somme d'une liste de nombres. En donnant le nom d'une rubrique tableau, ce sont toutes les valeurs contenues qui sont prises en compte.
Syntaxe : SOMME(liste1; liste2; etc.)
Exemples : SOMME(12; 45; -3; 28) donne 82
SOMME(@Prix) où prix est une rubrique tableau

MOYENNE( )
Calcule la moyenne d'une liste de nombres. En donnant le nom d'une rubrique tableau, ce sont toutes les valeurs contenues qui sont prises en compte. Si une valeur du tableau manque, elle n'est pas considérée comme une valeur nulle !
Syntaxe : MOYENNE(liste1; liste2; etc.)
Exemples : MOYENNE(12; 45; -3; 28) donne 20.5
MOYENNE(@Prix) où prix est une rubrique tableau

RASSEMBLE(;)
Rassemble (en les sommant) tous les nombres d'une liste si un test dans une autre liste est vrai. Supposons deux rubriques tableau côte à côte. La première contient des montants (@Prix), et la deuxième des taux de TVA (@TauxTVA). On désire obtenir la somme des tous les montants situés sur une ligne en regard d'un taux de TVA de 6.5%. Il suffit d''écrire :
RASSEMBLE(@Prix; @TauxTVA=6.5)
Cette commande permet, par exemple, de calculer dans une facture les sommes des articles vendus aux différents taux pratiqués.
Syntaxe : RASSEMBLE(liste; test)
Exemples : RASSEMBLE(@SousTotal; @TauxTVA=2)

COMPTE( )
Compte le nombre de lignes d'une rubrique tableau. Si le tableau contient des lignes intermédiaires vides, elles sont comptées.
Syntaxe : COMPTE(liste)
Exemples : COMPTE(@Désignation)
Un test permet de comparer deux nombres ou deux chaînes. Son résultat est un nombre, toujours égal à 1 si la comparaison est vraie, et à 0 si la comparaison est fausse. Par exemple, l'opérateur = peut être comparé à l'opérateur +. Tout comme le calcul 2+3 retourne la valeur 5, le calcul 2=3 retourne la valeur 0 (faux).
Lorsque deux chaînes sont comparées, c'est l'égalité (ou l'inégalité) stricte qui est prise en compte (pas d'équivalences entre majuscules, minuscles et lettres accentuées). Les comparaisons "plus petit ou plus grand que" tiennent compte de l'ordre alphabétique ("A" est plus petit que "B").

NOTE : Compter le nombre de lignes d'un tableau :

On aimerait connaître le nombre de lignes d'un tableau, le nombre de lignes remplies... Il y a ici 2 notions, le nombre de lignes total, et le nombre de ligne remplie. Supposons un tableau avec articles, une colonne désignation et une colonne de couleur. Il y a bien 6 articles, mais seulement certains ont une indication de couleur.

Exemple	@Désignation	@Couleur	@Quantité	@Prix
Chemise	noir	2	75.60	
Cravate		5	124.10	
Pantalon	rouge	1	89.50	
Polo	noir	5	57.20	
Chapeau		6	125.00	
Chemise			58.90	

COMPTE Compte le nombre de ligne (en regardant le plus grand index non vide):
COMPTE(@Désignation) donne 6
COMPTE(@Couleur) donne 4
puisque les 2 dernières lignes ne sont pas remplies.
COMPTE(@Quantité) donne 5
puisque la dernière ligne n'est pas remplie.

RASSEMBLE Rassemble (en les sommant) tous les nombres d'une liste en testant une valeur dans une autre liste. Pour connaître la quantité d'éléments de couleur, on peut faire :

RASSEMBLE (@Quantité ; @Couleur = "noir")
C'est à dire qu'on ne prend la quantité que si la case @Couleur correspondante contient "noir".
Ici on aura 7 comme résultat (2 + 5), c'est à dire le total des articles noirs.

RASSEMBLE (@Quantité ; NON (VIDE (@Couleur)))
C'est à dire qu'on ne prend la quantité que si la case @Couleur correspondante est remplie d'une valeur quelconque.
Ici on aura 8 comme résultat (2 + 1 + 5), c'est à dire le nombre total d'article de couleur.

RASSEMBLE (1; NON (VIDE (@Couleur)))
Ici on ne prend plus la quantité qu'il y a dans la ligne, mais toujours 1. Et on ne prend ce 1 que si la case @Couleur correspondante est remplie d'une valeur quelconque.
Ici on aura 3 puisque le nombre de lignes remplies avec une couleur est de 3.

CHERCHE(;)

Cherche dans un tableau, la ligne qui contient une valeur.
Syntaxe : CHERCHE(Rubriquede format tableau ; valeur)
Exemple : CHERCHE(@Adésignation ; "**code***"

INDEX()

Donne l'index de la ligne courante du tableau.
Exemple : @Tableau[INDEX() –1]

Les tests

Ce chapitre regroupe toutes les opérations permettant d'effectuer des tests. Un test est une sorte d'aiguillage permettant d'effectuer différents calculs selon le résultat d'une comparaison :

Un test permet de comparer deux nombres ou deux chaînes. Son résultat est un nombre, toujours égal à 1 si la comparaison est vraie, et à 0 si la comparaison est fausse. Par exemple, l'opérateur **=** peut être comparé à l'opérateur **+**. Tout comme le calcul 2+3 retourne la valeur 5, le calcul 2=3 retourne la valeur 0 (faux). Lorsque deux chaînes sont comparées, c'est l'égalité (ou l'inégalité) stricte qui est prise en compte (pas d'équivalences entre majuscules, minuscles et lettres accentuées). Les comparaisons "plus petit ou plus grand que" tiennent compte de l'ordre alphabétique ("A" est plus petit que "B").

=
Comparaison d'égalité entre deux nombres ou chaînes.
Syntaxe : op1 = op2
Exemples : 81=82 donne 0 (faux)
81=81 donne 1 (vrai)
"fenêtre"="fenetre" donne 0 (faux)
"fenêtre"="Fenêtre" donne 0 (faux)
@Total=0
@Ville="Lausanne"

<>
Comparaison d'inégalité entre deux nombres ou chaînes.
Syntaxe : op1 <> op2
Exemples : 81<>82 donne 1 (vrai)
81<>81 donne 0 (faux)
@Titre<>"Monsieur"

<
Comparaison "plus petit que" entre deux nombres ou chaînes.
Syntaxe : op1 < op2
Exemples : 81<80 donne 0 (faux)
81<81 donne 0 (faux)
81<82 donne 1 (vrai)
"Eléphant"<"Souris" donne 1 (vrai, un éléphant est plus petit qu'une souris !)

<=
Comparaison "plus petit ou égal à" entre deux nombres ou chaînes.
Syntaxe : op1 <= op2
Exemples : 81<=80 donne 0 (faux)
81<=81 donne 1 (vrai)
81<=82 donne 1 (vrai)

>
Comparaison "plus grand que" entre deux nombres ou chaînes.
Syntaxe : op1 > op2
Exemples : 81>80 donne 1 (vrai)
81>81 donne 0 (faux)
81>82 donne 0 (faux)

>=
Comparaison "plus grand ou égal à" entre deux nombres ou chaînes.
Syntaxe : op1 >= op2
Exemples : 81>=80 donne 1 (vrai)
81>=81 donne 1 (vrai)
81>=82 donne 0 (faux)

CONTIENT(;)

Comparaison d'égalité entre une chaîne normale et une deuxième pouvant contenir des jokers. L'opérateur retourne 1 (vrai) si les chaînes sont semblables ou 0 (faux) si les chaînes sont différentes. Les minuscules sont équivalentes aux majuscules, et les accents sont ignorés.

Syntaxe : CONTIENT(chaîne; jokers)
Exemples : CONTIENT("béta"; "*a*") donne 1 (vrai)
CONTIENT("béta"; "a*") donne 0 (faux)
CONTIENT(@Ville; "*yverdon*")

SI(; ;)

Evalue une expression ou une autre en fonction du résultat d'une comparaison. Pour combiner plusieurs conditions, utiliser les opérateurs logiques.

Syntaxe : SI(test; vrai; faux)
Exemples : SI(0; 11; 22) donne 22 (le test est faux)
SI(1; 11; 22) donne 11 (le test est vrai)
SI(@Nb=12; "oui"; "non") donne "oui" si la rubrique Nb contient 12
SI(@Rabais=0; "Pas de rabais"; "Rabais accordé")

Remarque : Pour effectuer des tests avec les boutons action, il faut utiliser l'opérateur #SI.
Les opérateurs logiques permettent de combiner logiquement plusieurs tests, par exemple dans un opérateurSI.

ET(;)

ET logique entre deux ou plusieurs tests, qui doivent tous être vrai pour que le résultat soit également vrai.

Syntaxe : ET(test1; test2; ...)
Exemples : ET(1<2; 25<100) donne 1 (vrai)
ET(1<2; 25<8) donne 0 (faux)
ET(@Rabais>5; @Total>100)

OU(;)

OU logique entre deux ou plusieurs tests, qui doivent tous être faux pour que le résultat soit également faux.

Syntaxe : OU(test1; test2; ...)
Exemples : OU(12<50; 45<3) donne 1 (vrai)
OU(12<4; 45<3) donne 0 (faux)
OU(@Payé=1; @SImpression=1)

NON( )

NON logique, pour inverser le résultat d'un test.

Syntaxe : NON(test)
Exemples : NON(1=2) donne 1 (vrai)
NON(ET(@Rabais>5; @Total>100))

#SI, #SINON et #FIN

Utilisé uniquement dans les boutons.
Lorsque des calculs ne doivent être exécutés que dans certaines conditions, il est possible d'utiliser les instructions **#SI, #SINON et #FIN** pour séparer différents groupes de calculs.

Syntaxe :
```
#SI test
#SINON
#FIN
```
Si le calcul spécifié dans le test est vrai, les lignes suivantes sont exécutées, jusqu'au #SINON ou jusqu'au #FIN. Les calculs entre le #SINON et le #FIN sont effectués si le test est faux.

Exemple :
```
#SI @Payé=0
    @Payé = 1
    @Report = @Total
#FIN
```

#TANTQUE, #REPETE

Utilisé uniquement dans les boutons.
On peut également faire une série d'assignation dans une boucle, en utilisant les instructions **#TANTQUE, #REPETE**

Syntaxe : #TANTQUE test
#REPETE

Tant que le calcul spécifié dans le test est vrai, les lignes suivantes sont exécutées, jusqu'au #REPETE. Il faut bien entendu penser à faire varier la valeur testée dans #TANTQUE pour que la boucle se termine un jour. Toutefois, les boucles sont limitées à 200 itérations.

Exemple :
```
@Index := 0
#TANQUE @Index < COMPTE(@Tableau)
    @Copie[@Index] := @Tableau[@Index]
    @Index := @Index + 1
#REPETE
```

Les conversions

ARRONDI(;)

Arrondit au multiple le plus proche, par excès (mode = 1 ou rien) ou par défaut (mode = 0).
Syntaxe : ARRONDI(montant; multiple; mode)
Exemples : ARRONDI(4.22; 0.05) donne 4.20
ARRONDI(4.23; 0.05) donne 4.25
ARRONDI(4.23; 1) donne 4
ARRONDI(2.5; 1) donne 3 (par excès)
ARRONDI(3.175; 0.05; 1) donne 3.20 (par excès)
ARRONDI(3.175; 0.05; 0) donne 3.15 (par défaut)

INT( )

Conserve uniquement la partie entière d'un nombre.
Syntaxe : INT(montant)
Exemples : INT(3.9) donne 3
INT(-4.2) donne -4

FRAC( )

Conserve uniquement la partie fractionnaire d'un nombre.
Syntaxe : FRAC(montant)
Exemples : FRAC(3.9) donne 0.9
FRAC(-4.2) donne -0.2
CHAINE(INT(@Quantité); 0)+"h"+CHAINE(FRAC(@Quantité)*60; 0)
si @Quantité = 2.5, donne la chaîne 2h30

ABS( )

Supprime le signe d'un nombre, afin qu'il soit toujours positif.
Syntaxe : ABS(montant)
Exemples : ABS(3.9) donne 3.9 (inchangé)
ABS(-4.2) donne 4.2

DEFAUT(;)

Il ne faut pas confondre une rubrique numérique inexistante avec une rubrique contenant zéro. Une rubrique inexistante n'a jamais été remplie; son contenu est indéterminé. Si elle est utilisée dans un calcul, CRESUS ne peut pas évaluer l'expression. Par exemple : @Quantité*10 Si la rubrique quantité n'existe pas, ce calcul génère une erreur. Il faut le remplacer par : DEFAUT(@Quantité;0)*10
Remplace une rubrique inexistante par une valeur par défaut.
Syntaxe : DEFAUT(rubrique; défaut)
Exemples : DEFAUT(@Rabais; 10)
DEFAUT(@TauxTAV; 6.5)

VIDE()

Il ne faut pas confondre une rubrique numérique inexistante avec une rubrique contenant zéro. Le résultat d'un calcul donne en principe toujours une valeur. Si un calcul doit rendre la valeur "inexistant", il faut utiliser la constante numérique VIDE().
Remarque : Pour spécifier une chaîne vide, il suffit de donner "" (deux guillemets).
Donne un nombre vide, c'est-à-dire inexistant.
Syntaxe : VIDE()
Exemples : SI(@Rabais=0; VIDE(); @Rabais)

Il est possible de tester si une valeur existe ou non en utilisant la syntaxe suivante :

```
SI( VIDE( @Truc) ; 0 ; 1 )
```

CHAINE( )

Convertit une date, une heure ou un nombre en une chaîne. Cela est nécessaire avant d'appondre une rubrique numérique à une chaîne, par exemple. Le nombre de décimales n'est utile que pour les rubriques numériques fractionnaires. S'il est omis, il est converti à deux décimales. Le nombre de digits permet de forcer le nombre de chiffres pour la partie entière, en ajoutant des zéro non significatifs au début du nombre.
Syntaxe : CHAINE(nombre; décimales; nbdigits)
CHAINE(nombre; décimales)
CHAINE(nombre) donne un nombre à deux décimales
Exemples : "le "+CHAINE(@Date)
"le "+CHAINE(AUJOURDHUI())
CHAINE(@Prix)+" francs"
CHAINE(3.849) donne "3.85"
CHAINE(3.849; 1) donne "3.9"
CHAINE(3.849; 0) donne "4"
CHAINE(23; 0; 5) donne "00023"

CHAINEMOIS( )

Convertit un numéro de mois compris entre 1 et 12 en une chaîne en français (langue = 0), allemand (langue = 1) ou anglais (langue = 2).
Syntaxe : CHAINEMOIS(mois; langue)
CHAINEMOIS(mois) donne un texte en français
Exemples : CHAINEMOIS(2; 0) donne "février"
CHAINEMOIS(2) donne "février"
CHAINEMOIS(3; 1) donne "März"
CHAINEMOIS(MOIS(@Date); 0)
"le "+CHAINE(JOUR(@Date))+" "+CHAINEMOIS(MOIS(@Date))+" "+CHAINE(ANNEE(@Date))

CHAINEJDLS( )

Convertit un numéro de jour de la semaine compris entre 1(lundi) et 7 (dimanche) en une chaîne en français (langue = 0), allemand (langue = 1) ou anglais (langue = 2).
Syntaxe : CHAINEJDLS(jour; langue)
CHAINEJDLS(jour) donne un texte en français
Exemples : CHAINEJDLS(3; 0) donne "mercredi"
CHAINEJDLS(5; 1) donne "Friday"
CHAINEJDLS(JDLS(@Date); 1)

FINLIGNE( )

Cet opérateur retourne simplement une chaîne contenant une fin de ligne. Il permet, par exemple dans un calcul temporaire, de générer un texte de plusieurs lignes.
Syntaxe : FINLIGNE()
Exemples : "Première ligne" + FINLIGNE() + "Deuxième ligne"
@Prénom+" "+@Nom + FINLIGNE() + @Adresse

NOMSTATISTIQUE( )

Cet opérateur retourne simplement une chaîne contenant le nom de la statistique en cours. Il est ainsi possible de réaliser une seule présentation pour toutes les statistiques, tout en imprimant les différents noms de statistiques dans l'en-tête par exemple.
Syntaxe : NOMSTATISTIQUE()
Exemples : "Statistique " + NOMSTATISTIQUE()

NOMACCES( )

Cet opérateur retourne simplement une chaîne contenant le nom de l'accès en cours.
Ce peut être l'accès utilisé pour imprimer (présentations imprimante),
ou l'accès actuel (présentations écran)
Syntaxe : NOMACCES()
Exemples : "Selon accès " + NOMACCES()

ACCESCOMPTE("NomAcces" ; "NomBase")

Donne le nombre de fiches se trouvant dans l'accès spécifié.
Le second paramètre est optionnel (base courante), sinon il doit être l'un des 5 mots suivants: "Factures", "Clients", "Articles", "Commandes", "Fournisseurs".

FORMATESPACE(; ;)

Cet opérateur formate un nombre en ajoutant un espace entre chaque chiffre, pour permettre par exemple l'impression des montants sur les nouveaux bulletins de versement rouge des PTT.
Le caractère de bourrage est utilisé pour remplir les chiffres inutilisés à gauche. La longueur indique le nombre total de cases à remplir.
Le nombre formaté ne contient jamais de point pour séparer les francs des centimes, ni de guillements pour séparer les milliers.
Pour imprimer un bulletin de versement rouge, il ne faut pas oublier d'utiliser une police de caractère à chasse fixe, telle que "Courrier" ou "OCR-B".

Exemple	Résultat obtenu
FORMATESPACE(1234.50; "0"; 11)	0 0 0 0 1 2 3 4 5 0
FORMATESPACE(1234.50; "-"; 11)	- - - - 1 2 3 4 5 0
FORMATESPACE(1234.50; "0"; 9)	0 0 1 2 3 4 5 0

Syntaxe : FORMATESPACE(rubrique; bourrage; lg)
Exemples : FORMATESPACE(@Total; "0", 11)

Les pour-cent (Calcul)

Les calculs de pourcentage peuvent très bien être réalisés à l'aide des quatre opérations de base (addition, soustraction, multiplication et division). Ce chapitre montre cependant une méthode plus simple et plus compacte.

C%(;)

Calcule le pourcentage d'un montant.
Syntaxe : C%(montant; taux)
Formule : résultat = (montant*taux)/100
Exemples : C%(100; 6.5) donne 6.5
C%(20; 6.5) donne 1.3
C%(@Prix; @TauxTVA)

A%(;)

Ajoute un pourcentage à un montant.
Syntaxe : A%(montant; taux)
Formule : résultat = montant+(montant*taux)/100
Exemples : A%(100; 6.5) donne 106.5
A%(20; 6.5) donne 21.3

S%(;)
Calcule la valeur qu'avait un montant avant qu'on ne lui ajoute un pourcentage.
Syntaxe : S%(montant; taux)
Formule : résultat = (montant*100)/(taux+100)
Exemples : S%(106.5; 6.5) donne 100
S%(21.3; 6.5) donne 20

M%(;)
Calcule la valeur ajoutée à un montant augmenté d'un pourcentage.
Syntaxe : M%(montant; taux)
Formule : résultat = montant-((montant*100)/(taux+100))
Exemples : M%(106.5; 6.5) donne 6.5
M%(21.3; 6.5) donne 1.3

T%(;)
Calcule le pourcentage entre deux montants.
Syntaxe : T%(montant1; montant2)
Formule : résultat = ((montant1/montant2)-1)*100
Exemples : T%(106.5; 100) donne 6.5
T%(21.3; 20) donne 6.5
T%(6; 3) donne 100
T%(6; 4) donne 50
T%(4; 8) donne -50

+ et -
Ajoute ou enlève un pourcentage à un montant.
Syntaxe : montant + taux%, montant - taux%
Formule : résultat = montant + (montant*taux)/100, résultat = montant - (montant*taux)/100
Exemples : 100+6.5% donne 106.5
20-5% donne 19
@Prix-@Rabais%

* et /
Multiplie ou divise un montant par un taux.
Syntaxe : montant * taux%, montant / taux%
Formule : résultat = montant*(taux/100), résultat = montant/(taux/100)
Exemples : 45*30% donne 13.5
75/50% donne 150

Les dates et heures

De façon interne, une date est représentée par un nombre entier égal au nombre de jours écoulés depuis le 1 janvier 1900. Ceci permet, par exemple, de soustraire deux dates pur connaître le nombre de jours qui les séparent. Ou encore d'ajouter un nombre entier (nombre de jour) à une date pour obtenir une autre date.
Exemples : "écoulé depuis "+CHAINE(AUJOURDHUI()-@Date)+" jours"
SI(@Date+30 < AUJOURDHUI(); "Rappel"; "Délai")

AUJOURDHUI()
Donne la date du jour.
Syntaxe : AUJOURDHUI()
Exemples : AUJOURDHUI() donne 31.3.96 (au hasard)

DATE(; ;)
Donne une date constante quelconque. Si le jour, le mois ou l'année manquent, c'est la date actuelle qui fait foi.
Syntaxe : DATE(jour; mois; année)
Exemples : DATE(31; 3; 96)
DATE(1; 1) donne le premier janvier de l'année en cours

DATE(1) donne la date 1.6.96 si on est en juin 1996
AUJOURDHUI()-DATE(1; 1; 96)

JOUR( )

Convertit une date en un numéro de jour compris entre 1 et 31. Pour convertir le résultat numérique en une chaîne représentant le nombre, utilisez l'opérateur CHAINE.
Syntaxe : JOUR(date)
Exemples : JOUR(@Date) donne 31 si Date contient 31.3.96
JOUR(@Date+1) donne 1 si Date contient 31.3.96

MOIS( )

Convertit une date en un numéro de mois compris entre 1 et 12. Pour convertir le résultat numérique en une chaîne contenant le nom du mois, utilisez l'opérateur CHAINEMOIS.
Syntaxe : MOIS(date)
Exemples : MOIS(@Date) donne 3 (mars) si Date contient 31.3.96
MOIS(@Date+1) donne 4 (avril) si Date contient 31.3.96

ANNEE( )

Convertit une date en un numéro d'année compris entre 1900 et 2100. Pour convertir le résultat numérique en une chaîne représentant le nombre, utilisez l'opérateur CHAINE.
Syntaxe : ANNEE(date)
Exemples : ANNEE(@Date) donne 1996 si Date contient 31.3.96

JDLS( )

Convertit une date en un numéro de jour de la semaine compris entre 1 (pour lundi) et 7 (pour dimanche). Pour convertir le résultat numérique en une chaîne contenant le nom du jour de la semaine, utilisez l'opérateur CHAINEJDLS.
Syntaxe : JDLS(date)
Exemples : JDLS(@Date) donne 7 (dimanche) si Date contient 31.3.96

SEMAINE( )

Convertit une date en un numéro de semaine compris entre 0 et 53. Une semaine va du lundi au dimanche. Si le premier janvier est compris entre lundi et jeudi, il appartiendra à la semaine un. S'il est compris entre vendredi et dimanche, il appartiendra à la semaine numéro zéro.
Syntaxe : SEMAINE(date)
Exemples : SEMAINE(DATE(31; 3; 96)) donne 13
SEMAINE(DATE(1; 1; 98)) donne 1
SEMAINE(DATE(1; 1; 99)) donne 0

ADDMOIS(;)

Ajoute un ou plusieurs mois à une date, en changeant d'année si nécessaire.
Syntaxe : ADDMOIS(date)
ADDMOIS(date; nbmois)
Exemples : ADDMOIS(DATE(1; 1; 96)) donne le 1.2.96
ADDMOIS(DATE(31; 1; 96)) donne le 29.2.96
ADDMOIS(DATE(15; 7; 96); 6) donne le 15.1.97

DEBUTMOIS( )

Retourne la date au premier jour du mois.
Syntaxe : DEBUTMOIS(date)
Exemples : DEBUTMOIS(DATE(1; 3; 96)) donne le 1.3.96
DEBUTMOIS(DATE(15; 3; 96)) donne le 1.3.96
DEBUTMOIS(DATE(31; 3; 96)) donne le 1.3.96

FINMOIS( )

Retourne la date au dernier jour du mois.
Syntaxe : FINMOIS(date)
Exemples : FINMOIS(DATE(1; 2; 96)) donne le 29.2.96

FINMOIS(DATE(15; 2; 96)) donne le 29.2.96
FINMOIS(DATE(29; 2; 96)) donne le 29.2.96

De façon interne, une heure est représentée par le nombre de secondes écoulées depuis minuit. Ceci permet, par exemple, de soustraire deux heures pour connaître le nombre de secondes qui les séparent.
Exemples : "Durée = "+CHAINE((@Fin-@Début)/60)+" minutes"

MAINTENANT()
Donne l'heure actuelle.
Syntaxe : MAINTENANT()
Exemples : MAINTENANT() donne 12:00 s'il est midi

TEMPS(; ;)
Donne une heure constante quelconque. Les paramètres manquants sont remplacés par l'heure actuelle.
Syntaxe : TEMPS(heures; minutes; secondes)
Exemples : TEMPS(11; 45; 30)
TEMPS(9; 30; 0)
TEMPS(; 0; 0) donne 11:00:00 s'il est 11:34:52
(MAINTENANT()-TEMPS(6; 0; 0))/3600

HEURE( )
Convertit un temps en un nombre d'heures compris entre 0 et 23.
Syntaxe : HEURE(date)
Exemples : HEURE(@heure) donne 12 si Heure contient 12:15:30

MINUTE( )
Convertit un temps en un nombre de minutes compris entre 0 et 59.
Syntaxe : MINUTE(date)
Exemples : MINUTE(@heure) donne 15 si Heure contient 12:15:30

SECONDE( )
Convertit un temps en un nombre de secondes compris entre 0 et 59.
Syntaxe : SECONDE(date)
Exemples : SECONDE(@heure) donne 30 si Heure contient 12:15:30

Les calculs sur les chaînes

MAJUSCULE( )
Transforme tous les caractères de la chaîne en caractères majuscules.
Syntaxe : MAJUSCULE(chaîne)
Exemples : MAJUSCULE("Chaîne") donne CHAINE

MINUSCULE( )
Transforme tous les caractères de la chaîne en caractères minuscules.
Syntaxe : MINUSCULE(chaîne)
Exemples : MINUSCULE("C'EST CA") donne "c'est ca"

NOMPROPRE( )
Transforme chaque mot de la chaîne en nom propre (première lettre majuscule, le reste en minuscule).
Syntaxe : NOMPROPRE(chaîne)
Exemples : NOMPROPRE("c'est un exemple") donne "C'Est Un Exemple"

GAUCHE(;)
Retourne les N premiers caractères de la chaîne.
Syntaxe : GAUCHE(chaîne, longueur)
Exemples : GAUCHE("Bonjour", 3) donne "Bon"
GAUCHE("Bonjour", 100) donne "Bonjour"

DROITE(;)
Retourne les N derniers caractères de la chaîne.
Syntaxe : DROITE(chaîne, longueur)
Exemples : DROITE("Bonjour", 4) donne "jour"
DROITE("Bonjour", 100) donne "Bonjour"

MILIEU(; ;)
Extrait des caractères au centre de la chaîne.
Syntaxe : MILIEU(chaîne, debut, longueur)
Exemples : MILIEU("Bonjour", 2, 3) donne "njo"
MILIEU("Bonjour", 2, 100) donne "njour"
MILIEU("Bonjour", 20, 1) donne ""

LONGUEUR( )
Retourne la longueur de la chaîne (le nombre de caractères).
Syntaxe : LONGUEUR(chaîne)
Exemples : LONGUEUR("Bonjour") donne 7

TROUVE(;)
Cherche la première occurance d'une sous-chaîne dans une chaîne.
Syntaxe : TROUVE(chaîne, souschaîne)
Exemples : TROUVE("Bonjour", "o") donne 1
TROUVE("Bonjour", "ou") donne 4
TROUVE("Bonjour", "B") donne 0
TROUVE("Bonjour", "b") donne -1 (pas trouvé)

NOMBRE( )
Transforme le contenu d'une chaîne de caractère en nombre décimal.
Syntaxe : NOMBRE(chaîne)
Exemples : NOMBRE("-1'234.55") donne le nombre -1234.55

MOT(; ;)
Extrait un mot d'une chaîne.
Syntaxe : MOT(chaîne, position, separateurs)
Exemples : MOT("C'est une phrase", 2, " ") donne "une"
MOT("C'est une phrase", 1, " ") donne "C'est"
MOT("C'est une phrase", 0, " ") donne "C'est" (cas particulier 0 identique à 1)
MOT("C'est une phrase", 5, " ") donne ""
MOT("X/Y-Z+U U/V";4;"/-+") donne "U U"

LIGNE(;)
Retourne une ligne d'une rubrique multi-lignes.
Syntaxe : LIGNE(chaîne, num)
Exemples : LIGNE(@Adresse; 2) donne la seconde ligne de l'adresse.

CODECAR( )
Construit une chaîne à partir des codes ASCII des caractères.
Syntaxe : CODECAR(num)
CODECAR(num1 ; num2 ; num3 ; num4 ; .....)
Exemples : CODECAR(9) retourne un "tabulateur".
CODECAR(65; 66; 67; 68) retourne la chaîne "ABCD"

CANTON()
CANTON("1110") retourne la chaîne "VD".

Les calculs spéciaux

Opérations pour BVR

Les trois opérations BVRAD, BVRREFet BVRCODE permettent de générer tous les numéros demandés par les PTT, pour l'impression de bulletins de versement avec numéro de référence (bulletins bleus). Les numéros produits dépendent des définitions pour BVR.
Les différents objets calculs temporaires doivent utiliser une police de caractères OCR-B. Ils doivent être positionnés très précisément dans le bulletin bleu, en fonction des spécifications des PTT.

BVRAD( )

Donne le numéro d'adhérent BVR. Ce numéro est de type alphanumérique. Le paramètre def indique quel numéro d'adhérent il faut utiliser, par un nombre compris entre 0 et 4. Si ce paramètre est omis, c'est le premier numéro d'adhérent qui est utilisé (0).

Syntaxe : BVRAD(def)
Exemples : BVRAD()
BVRAD(4)
Résultat : 01-162-8

BVRREF(;)

Donne le numéro de référence BVR. Ce numéro est de type alphanumérique. Le numéro de rappel permet de distinguer un numéro de référence imprimé sur une facture, un premier rappel, un deuxième rappel, etc. En fonction de la présentation en cours de dessin, il faut utiliser les numéros suivants :

0 facture
1 premier rappel
2 deuxième rappel
3 troisième rappel

Le paramètre def indique quel numéro d'adhérent il faut utiliser, par un nombre compris entre 0 et 4. Si ce paramètre est omis, c'est le premier numéro d'adhérent qui est utilisé (0).

Syntaxe : BVRREF(rappel; def)
Exemples : BVRREF(0)
BVRREF(1; 4)
Résultat : 0 09600 10001 00506 (16 positions)
33 12340 09600 10000 00001 00502 (27 positions)

Le numéro de référence correspond à la partie centrale du numéro complet BVRCODE, groupé par tranches de 5 chiffres. Lorsqu'il s'agit d'un codage à 16 positions (PTT), il contient le numéro de la facture (7 chiffres), le numéro du client (7 chiffres) et le numéro de rappel (1 chiffre). Par exemple :

0 09600 10001 00538

0096001 facture numéro 96001
0001005 client numéro 1005
3 troisième rappel
8 chiffre de contrôle calculé automatiquement

Lorsqu'il s'agit d'un codage à 27 positions (banque), il contient en plus un numéro fixe propre à votre banque :

33 12340 09600 10000 00001 00502

331234 partie fixe propre à votre banque
0096001 facture numéro 96001
00000 inutilisé
0001005 client numéro 1005
0 facture
2 chiffre de contrôle calculé automatiquement

Le numéro de la facture, le numéro du client et le numéro de rappel se retrouvent sur la disquette fournie par les PTT, lorsque des paiements sont effectués. Ce sont ces trois numéros qui permettent de déterminer la provenance du paiement.

BVRCODE(; ;)

Donne le numéro de codage BVR complet. Le numéro de rappel permet de distinguer un numéro de référence imprimé sur une facture, un premier rappel, un deuxième rappel, etc. Si la commande BVRCODE ne contient pas de montant, le numéro de codage permet de générer des BVR+ (BVR sans montant

préimprimé). Si le montant existe mais est plus petit ou égal à zéro, il s'agit d'un bulletin non destiné à des paiements.
Le paramètre def indique quel numéro d'adhérent il faut utiliser, par un nombre compris entre 0 et 4. Si ce paramètre est omis, c'est le premier numéro d'adhérent qui est utilisé (0).

Syntaxe : BVRCODE(rappel; montant; def)
BVRCODE(rappel)
Exemples : BVRCODE(0; @Total)
BVRCODE(0)
BVRCODE(0; 0))
Résultats : 0100000123458>0096001000100506+ 010001628>
042>0096001000100506+ 010001628> (BVR+ sans montant)
010001628> (montant nul)

Lorsque le numéro est complet, la première partie contient le montant du paiement, la deuxième le numéro de la facture et le numéro du client, et la dernière le numéro d'adhérent. Par exemple :
0100000123458>0096001000100514+ 010001628>

0100000123458>	montant de 123.45 francs
0096001000100514+	facture numéro 96001 pour le client numéro 1005, premier rappel
010001628>	numéro d'adhérent 01-162-8

La partie centrale est plus longue dans le cas d'un codage à 27 positions. Les chiffres supplémentaires sont des signes de contrôle.

BVRNOM( )
Donne le nom d'un compte BVR. Ce nom est défini dans le dialogue de définitions pour le système de paiement BVR. Le numéro correspondant doit être compris entre 0 et 4.
Syntaxe : BVRNOM(numéro)
Exemples : BVRNOM(0)
BVRNOM(4)

Opérations pour DTA / SOG
Ces opérateurs permettent d'adapter une présentation pour les paiements par DTA/SOG.
Remarque : Toutes ces fonctions ne sont disponibles que dans la version Largo :

DTANOM( )
Donne le nom d'une banque pour le SOG/DTA. Ce nom est défini dans le dialogue de définitions pour le système de paiement SOG/DTA. Le numéro correspondant doit être compris entre 0 et 4.
Syntaxe : DTANOM(numéro)
Exemples : DTANOM(0)
DTANOM(4)

DTAADRESSE( )
Donne l'adresse "versé par" d'une banque pour le SOG/DTA. Cette adresse est définie dans le dialogue de définitions pour le système de paiement SOG/DTA. Elle peut comporter jusqu'à quatre lignes. Le numéro correspondant doit être compris entre 0 et 4.
Syntaxe : DTAADRESSE(numéro)
Exemples : DTAADRESSE(0)
DTAADRESSE(4)

Les opérations sur la TVA et Comptes
Crésus-facturation récupère les codes à utiliser pour la TVA (dialogue Comptes et Codes TVA) et les taux qui leur sont associés, à partir de la comptabilité liée.
Les fonctions suivantes permettent de trouver les taux à utiliser selon les codes introduits par l'utilisateur.
Remarque : Pour que cela fonctionne totalement, il faut travailler en mode sécurisé, et avec la version 3.1 (ou suivantes) de Crésus-comptabilité.

TVATAUX( )
Donne le taux associé à un code, selon la table d'association courante.
En dehors des codes définis dans Crésus-comptabilité, il est possible d'utiliser les codes suivants :

« Normal »	taux standard (7.5%)
« Réduit »	taux réduit (2.3%)
« Hébergement »	taux spécifique à l'hôtellerie (3.2%)
« (Aucun) » ou chaîne vide	taux nul (0%).

Il est aussi possible de mettre une chaîne représentant un nombre à la place du code.
Syntaxe : TVATAUX(code)
Exemples : TVATAUX("TVARED")
TVATAUX("Normal")
TVATAUX("3.2")
Résultats : 2.3
7.5
3.2

TVACASE(; ; ;)
Semblable à la fonction précédente, mais à utiliser lorsque les taux ont été mémorisés.
Typiquement, dans une ancienne facture, il ne faut pas utiliser TVATAUX() pour connaître le taux à appliquer, car il faut utiliser les taux qui avaient cours à la création de la facture.
On mettra par exemple :
@TauxNormal = TVATAUX("Normal") par un contenu initial,
@TvaArticles = TVACASE(@AcodeTvaArticle ; @TauxNormal ;@TauxRéduit ;@TauxHéberg)
Syntaxe : TVACASE(code; tauxnormal ; tauxréduit ; tauxhébergement)
Exemples : TVACASE("TVARED" ; 7.5 ;2.3 ;3.2)
TVACASE(@ACodeTvaArticle; @TauxNormal ; @TauxRéduit ;@TauxHébergement)
Résultats : 2.3
7.5

Les autres cas divers

NOMCOMPTE( )
Retourne le nom d'un compte défini dans Crésus-compta à partir de son numéro alphanumérique.
Syntaxe : NOMCOMPTE(numéro)
Exemples : TVACASE("1020")
Résultats : « Banque B »

LANGUE( )
Donne la langue du logiciel
Exemple : LANGUE() rend 0 pour le français

TAUXCHANGE( )
Trouve le taux de change pour une monnaie et une date donnée.
Exemple : TAUCHANGE("EUR" ; AUJOURDHUI())

CREEFICHE( )

Provoque la création d'une nouvelle fiche (par exemple nouvelle facture) dans la base où est définie la création en série.
La fiche créée est initialisée avec la référence à la fiche (client par exemple) qui a servi à la création. La fonction donne en résultat le numéro de référence de la fiche nouvellement créée.
Syntaxe : CREEFICHE()
CREEFICHE(référence, index)

Résultats : CREEFICHE(), CREEFICHE(@Numéro), CREEFICHE(@Numéro ;1) sont tous équivalents. Cela ajoute une fiche dans la base de travail, en initialisant la première rubrique de type référence à la base parcourue avec le numéro de la fiche trouvée.
CREEFICHE(@RefClient) et CREEFICHE(@RefClient ; 1) sont équivalents. Cela ajoute une fiche dans la base de travail, en initialisant la première rubrique de type référence à la base correspondant à « RefClient » avec le contenue de @RefClient.

S'il y a deux rubriques références à la base parcourue dans la présentation principale (avec calculs standard), il peut-être utile de mettre CREEFICHE(@Numéro ; 2) ou CREEFICHE(@RefClient ;2) pour initialiser la seconde rubrique de référence au client et non la première.

CREEFICHE( ) retourne le numéro de référence de la fiche créée (ou zéro en cas d'erreur).
on peut donc faire

```
@RefNouvelle := CREEFICHE( )
    #SI @RefNouvelle = 0
        … en cas de problème
    #FIN
```

La fonction MENU

Cette fonction est particulière. Elle a sa place dans des calculs associés à des boutons. Son utilité consiste à simuler une action telle que l'utilisateur aurait pu faire depuis les menus du programme, comme par exemple changer de base, changer d'accès ou exporter des données. Voyez les notes importantes en pages 23.

Note : la première action d'un MENU va être de terminer l'édition de la fiche s'il y a lieu. Selon les réglages, si la fiche a été modifiée, la question classique « Mettre à jour la fiche ? » va s'afficher, la macro continue que la réponse soit oui ou non. Le premier paramètre de cette fonction est le texte du menu à sélectionner, par exemple "Fichier:Enregistrer". Toutes les fonctions du menu ne sont cependant pas accessible. Voici la liste des fonctions utilisables.

MENU("Edition:Valider les modifications")
Enregistre les modifications apportées à la fiche en édition. Simule donc une pression de la touche F12 par l'utilisateur. Les calculs pour F12 sont évalués, ainsi que les assignations différées en attente.

MENU("Edition:Annuler les modifications")
Ferme l'édition de la fiche sans enregistrer les modifications qui y ont été apportées. Simule donc une pression de la touche Esc.

MENU("Fichier:Enregistrer")
Enregistre le fichier dans son état actuel sur le disque.

MENU("Données:Accès" ; "NomAccès")
Permet de changer d'accès. Si le paramètre « NomAccès » n'est pas donné, c'est le dialogue de sélection des accès qui est ouvert. Les opérations suivantes sont annulée si on quitte ce dialogue avec le bouton « Annuler », ou si le paramètre « NomAccès » ne correspond pas à un nom d'accès existant (dans l'une des langues).

MENU("Données:Présentations" ; "NomPrésentation")
Permet de changer de présentation pour l'écran. Si le paramètre « NomPrésentation » n'est pas donné, c'est le dialogue de choix des présentations qui est ouvert. Si le paramètre est présent, mais ne correspond pas à une présentation écran existante (dans aucune des langues), la macro est stoppée.

MENU("Données:Factures")
MENU("Données:Clients")
MENU("Données:Articles")
MENU("Données:Fournisseurs")
MENU("Données:Commandes")
Passe dans la base de données correspondante. Il faut utiliser les noms tels qu'ils sont là, et pas les noms de bases paramétrés dans le dialogue « Noms des bases ».

MENU("Fichier:Exporter" ; "NomModèle" ; "NomAccès" ; "NomFichier")

Pour exporter des données. Les trois paramètres supplémentaires sont optionnels.
"NomModèle" est le nom du modèle à utiliser pour l'exportation. Si le paramètre est vide, le dialogue de sélection du modèle s'affiche. Si le nom donné n'existe pas, la macro est interrompue. "NomAccès" est le nom de l'accès à utiliser. Si le paramètre n'existe pas, c'est l'accès associé au modèle d'exportation qui sera utliisé. "NomFichier" est le nom du fichier à écrire sur le disque. Le dossier par défaut est celui où se trouve le document facture. Si le fichier existait déjà, il est remplacé sans avertissement. Si le paramètre est absent, le dialogue demandant le nom du fichier s'affiche.

MENU("Fichier:Importer" ; "NomModèle" ; "NomFichier")

Pour importer des données. Les deux paramètres supplémentaires sont optionnels."NomModèle" est le nom du modèle à utiliser pour l'importation. Si le paramètre est vide, le dialogue de sélection du modèle s'affiche. Si le nom donné n'existe pas, la macro est interrompue. "NomFichier" est le nom du fichier à lire sur le disque. Le dossier par défaut est celui où se trouve le document facture. Si le fichier existait déjà, il est remplacé sans avertissement. Si le paramètre est absent, le dialogue demandant le nom du fichier s'affiche.

MENU("Données:Création en série" ; "NomModèle" ; "NomAccès")

Lance une fonction de création en série. "NomModèle" est le nom du modèle à utiliser. Si le paramètre est vide, le dialogue de sélection de la fonction s'affiche. Si le nom donné n'existe pas, la macro est interrompue. "NomAccès" est le nom de l'accès à utiliser (si l'accès n'est pas imposé par le modèle).

MENU("Données:Statistiques" ; "NomStatistique" ; "xx")

Qui permet d'ouvrir une statistique. « NomStatistique » donne le nom de la statistique à utiliser. Si le paramètre est absent, c'est le dialogue du choix des statistiques qui est ouvert. « xx » est un code pour choisir l'opération désirée :

"0" donne l'aperçu rapide,
"1" fait l'impression
"2" fait l'aperçu avant impression (l'impression est possible depuis l'aperçu).
le code peut avoir un second caractère pour choisir entre une fiche ou toutes les fiches
"11" imprime pour la fiche courante
"12" imprime pour toutes les fiches.
"21" aperçu pour la fiche courante
"22" aperçu pour toutes les fiches.

MENU("Fichier:Imprimer" ; "NomPrésentation" ; "NomAccès" ; "NomPrésentation" ; "NomPrésentation" ; "NomPrésentation")

Pour imprimer des documents. Les paramètres supplémentaires sont optionnels. "NomPrésentation" est le nom de la présentation à utiliser pour l'impression. Si le paramètre est vide, le dialogue de sélection d'impression s'affiche. Si le nom donné n'existe pas, la macro est interrompue. "NomAccès" est le nom de l'accès à utiliser. Si le paramètre n'existe pas, c'est l'accès associé à la présentation ci-dessus qui sera utilisé. Les noms de présentation supplémentaires permettent d'imprimer plusieurs documents d'un coup, comme par exemple des copies de factures. **Note** : les éventuelles présentations chaînées associées à la première présentation sont de toutes façons ignorées par cette fonction. En cas d'erreur (pas de fiches à imprimer par exemple) la macro continue.

MENU("Fichier:Imprimer" ; "NomPrésentation" ; "1")
Cas particulier qui force l'impression de la fiche courante seule.

MENU("Fichier:Aperçu avant impression" ; "NomPrésentation" ; "NomAccès" ; "NomPrésentation" ; "NomPrésentation" ; "NomPrésentation")
Idem que ci-dessus, mais affiche l'aperçu à l'écran au lieu d'imprimer. S'il n'y a rien à imprimer, il n'y a pas de message d'erreur (rien à imprimer) et la macro continue.

MENU("Fichier:Imprimer comme précédemment")
Fait l'impression sans aucun dialogue. Cette fonction peut prendre les même paramètres que la fonction MENU("Fichier:Imprimer" ; ...) Par contre l'imprimante ne peut pas être sélectionnée et ce choix sera identique au choix précédent. S'il n'y a rien à imprimer, il n'y a pas de message d'erreur (rien à imprimer) et la macro continue.

Dans les fonctions ci-dessous (**Données:Opération**), pour pouvoir faire l'opération sans les accumulations ou/et sans les calculs selon F12, le nom de l'action peut être complété par l'un de ces 4 modes.

/11 (optionnel, mode par défaut) avec accumulations et avec calculs F12
/01 sans accumulations mais avec calculs F12
/10 avec accumulations, sans calculs F12
/00 sans accumulation, et sans calculs F12

MENU("Données:Opération" ; "mettre/11" ; "accès" ; "rubrique" ; "valeur")
mettre est un mot identifiant l'opération, accès est le nom de l'accès à utiliser, rubrique est le nom de la rubrique (sans le @), valeur est la valeur à mettre dans la rubrique. Ce peut être un calcul exprimé dans la chaîne "valeur" comme s'il était tapé dans le dialogue en question.

MENU("Données:Opération" ; "recalculer/11" ; "accès")
recalculer est un mot identifiant l'opération, accès est le nom de l'accès à utiliser, les fiches de cet accès sont recalculées.

MENU("Données:Opération" ; "actionner/11" ; "accès" ; "X")
actionner est un mot identifiant l'opération, accès est le nom de l'accès à utiliser, X est la lettre du raccourci à actionner Alt + X.

MENU("Données:Opération" ; "vider/11" ; "accès" ; "rubrique")
vider est un mot identifiant l'opération, accès est le nom de l'accès à utiliser, rubrique est le nom de la rubrique (sans le @), cette rubrique est enlevée des fiches.

MENU("Données:Opération" ; "supprimer/11" ; "accès" ; "yes")
supprimer est un mot identifiant l'opération, accès est le nom de l'accès à utiliser, yes fait disparaître le dialogue de confirmation. Attention cette fonction va supprimer une série de fiches de manière irréversible.

MENU("Fiche:Nouvelle fiche")
Crée une nouvelle fiche (simule un Ctrl+N).

MENU("Edition:Met à jour relation")
Simule un Ctrl+R (F2). Le curseur doit se trouver dans une case correspondant à une relation.

Autres fonctions spéciales

FOCUS(@NomRubrique ; ligne ; start ; end)

@NomRubrique est la rubrique correspondant à la case où l'on veut mettre le curseur,
ligne est, pour un tableau, le numéro de la ligne à sélectionner (0 à n-1),
start (optionnel, défaut = 0) est le premier caractère à sélectionner,
end (optionnel, défaut = -1) est le dernier caractère à sélectionner.
Par exemple FOCUS(@ADésignation, COMPTE(@ADésignation)-1, 0, 3) va mettre le curseur dans la dernière désignation en sélectionnant les 3 premiers caractères.

FOCUS est semblable aux fonction MENU, en ce sens qu'elle est également exécutée en différé.

```
FOCUS(@ADésignation ; 1)
        @ADésignation[1] := "Texte"
```

Donne comme résultat le curseur dans la case avec « Texte » déjà mis.

ACTION(Lettre)

Active un bouton de la présentation courante, simule donc un Alt + Lettre.
Par exemple ACTION(P) actionne le bouton **&Proposition**. Cette opération ne stoppe pas la macro en cours, on peut donc enchaîner plusieurs actions. Voir les notes sur ces fonctions (page 23)

PressePapiers

On peut mettre du texte dans le Presse-papiers de Windows à partir d'un bouton, en mettant dans le calcul :

```
@PressePapiers := "Texte à mettre dans le presse-papiers"
```

Il ne faut donc pas qu'une rubrique ou qu'une variable globale soit nommée « PressePapiers ». C'est l'équivalent du raccourcis clavier CTRL+C, sur une sélection, mais permet de copier une adresse complète par exemple.

L'espion

Les espions permettent de fouiller dans les nombreux calculs d'une application, pendant le dessin, et d'ainsi comprendre l'interaction des différents calculs entre eux.
Pour accéder à l'espion, il ne doit pas y avoir d'objet sélectionné (en cas de besoin, il suffit de cliquer un endroit ne contenant rien). Si nécessaire, sélectionnez l'outil flèche dans la palette flottante. Vous pouvez alors cliquer sur les yeux dans la barre d'icône.

Un autre moyen pour accéder à l'espion consiste à cliquer sur un objet avec le bouton de droite de la souris, puis de choisir "Espion..." dans le menu contextuel.

Boîte de dialogue

Le terme DBOX est l'abréviation usuelle de Dialogue Box, donc Boîte de Dialogue. La fonction DBOX permet d'afficher un message lors du déroulement d'un calcul, et d'influencer éventuellement sur le déroulement du calcul.

Tous ces appels retournent une valeur 1 (oui/ok) ou 0 (non/annule), que l'on peut tester ainsi

```
#SI DBOX( "D'accord ?", "OUI/NON" )
    action si oui
#FIN
```

DBOX("Texte à afficher" ; "vu")

Affiche le texte donné dans un dialogue, avec l'icône information et juste un bouton OK. Retourne toujours 1.

DBOX("Texte à afficher" ; "ko")

Affiche le texte donné dans un dialogue, avec l'icône stop et juste un bouton OK. Retourne toujours 1.

DBOX("Texte à afficher" ; "ok")

Affiche le texte donné dans un dialogue, avec l'icône attention, un bouton OK et un bouton Annuler. Retourne 1 pour OK et 0 pour Annuler.

DBOX("Texte à afficher" ; "oui/non")

Affiche le texte donné dans un dialogue avec les boutons OUI et NON. Retourne 1 si l'utilisateur choisi OUI, retourne 0 sinon.

DBOXINPUT("Texte à afficher" ; @NomVariableGlobale1 ; @NomVariableGlobale2 ; @NomVariableGlobale3 ; @NomVariableGlobale4 ; @NomVariableGlobale5)

Cette fonction affiche un dialogue de saisie semblable à un assistant d'une seule page et permet à l'utilisateur de modifier jusqu'à 5 variables globales. "Texte à afficher" est le texte qui s'affiche dans le dialogue pour expliquer ce qui doit être introduit, @NomVariableGlobale1à 5 sont les variables à éditer par ce dialogue. Retourne 0 si l'utilisateur a cliqué sur Annuler.

Action sur des fichiers

Il n'y a pas de fonction pour lire depuis un fichier. Pour la lecture, il faut utiliser l'importation de données.

EXISTEFILE( )

```
#SI EXISTEFILE("Data.txt")
        // opérations si le fichier existe
#FIN
```

Permet de savoir si un fichier existe ou non. Utilisé en général avec #SI

DELETEFILE( )

```
#SI DELETEFILE( "NomDuFichier.txt" )
    // opérations si le fichier ne peut être détruit
#FIN
```

Permet de supprimer un fichier.

WRITELN( )

```
#SI WRITELN("NomDuFichier"; "Texte à rajouter à la fin du fichier")
    // opérations si l'écriture n'a pas pu se faire.
#FIN
```

Ouvre le fichier donné par son nom (le crée si nécessaire) et ajoute la ligne de texte à la suite. Le fichier est refermé immédiatement. Chaque appel WRITELN( ) ajoute une nouvelle ligne au fichier désigné.

Autres commandes

EXECUTE( )

EXECUTE("Ligne de commande")
Permet d'exécuter une commande. La commande peut être un nom d'un fichier exécutable ou le nom d'un fichier à ouvrir ("toto.doc" ouvre le document dans Word), voir une URL. Il s'agit d'une « macro », c'est à dire comme les fonctions MENU(...), elle n'est pas exécutée de suite, mais à la suite des autres menus.

Exemple
```
MENU( "Fichier:Enregistrer" )
#SI DELETEFILE( "toto.txt" ) + WRITELN( "toto.txt" ; "Fichier vidé" )
#FIN
EXECUTE( "toto.txt" )
```
Cette fonction va faire (dans cet ordre)

- Détruire le fichier "toto.txt"
- Recréer le fichier "toto.txt" et y écrire le texte "Fichier vidé"
- Enregistrer le fichier FAC courant (menu Fichier - Enregistrer)
- Ouvrir "toto.txt" dans l'éditeur associé aux extensions .TXT

Autre exemple, importation de stock à partir de fichiers numérotés créés par ailleurs.

```
MENU( "Données:Articles" ) // passe dans la base des articles
#SI DELETEFILE("Renomme.bat")   // vide ou détruit le fichier Renomme.bat qui va renommer les fichiers
#FIN
      // boucle sur les fichiers numérotés
#TANTQUE ( "Stock_" + CHAINE(@VImport) + ".txt" )
      #SI DBOX("Le fichier Stock_" + CHAINE(@VImport) + " existe. Je l'importe"+FINLIGNE(); "VU")
      // Boîte de dialogue qui indique ce qui se passe
      #FIN
      // Prépare une ligne de commande et l'inscrit dans le fichier *.bat
      // par ex. : REN Stock_34.txt Stock_34.old
      #SI WRITELN("Renomme.bat";
                  "REN "+"Stock_" + CHAINE(@VImport) + ".txt "+"Stock_" + CHAINE(@VImport) + ".old")
      #FIN
     // donne l'ordre d'importation (à suivre) du fichier
   MENU( "Fichier:Importer..." ; "StockFiliale" ; "Stock_" + CHAINE(@VImport) + ".txt")
   @VImport := @VImport + 1
#REPETE
      // arrive sur un fichier qui n'existe pas, le signale
#SI DBOX("Le fichier Stock_" + CHAINE(@VImport) + ".txt n'existe pas"; "Vu")
#FIN
      // renomme les fichiers après importation
EXECUTE( "Renomme.bat" )
```

Les fonctions MENU, ACTION et EXECUTE

Elles sont très particulières. Lisez attentivement ces remarques si vous voulez les utiliser.

Ces fonctions n'exécutent pas immédiatement l'opération demandée, mais placent l'action à la suite d'une liste d'exécutions en attente. Les actions sont ensuite effectuées l'une après l'autre, une action ne débutant jamais avant que la précédente ne soit terminée.

Par exemple, si l'on écrit le calcul suivant dans un bouton :

```
@VTest := " Liste"
MENU( "Fichier:Imprimer" ; @VTest ; "acces")
@VTest := "Facture"
MENU( "Fichier:Imprimer" ; @VTest ; "acces" )
```

Les deux demandes d'impression vont être mises dans la liste d'opération à faire, mais elles ne seront exécutées que une fois l'action en cours terminée, c'est à dire lorsqu'on a fini de traiter tout le calcul ci-dessus.

Le résultat sera le même qu'en ayant écrit :

```
@VTest := " Liste"
@VTest := "Facture"
MENU( "Fichier:Imprimer" ; @VTest ; "acces")
MENU( "Fichier:Imprimer" ; @VTest ; "acces" )
```

Donc les deux impressions utiliseront la présentation « Facture » car c'est la valeur finale dans @VTest. Il suffit d'utiliser 2 variables différentes pour résoudre le cas ci-dessus.

Autre exemple, imaginons trois boutons &X, &Y et &Z avec les opérations suivantes :

&X faisant :

```
ACTION( Y )
MENU( "Données:Factures" )
ACTION( Z )
```

&Y faisant

```
MENU( "Données:Articles")
```

&Z faisant

```
MENU( "Données:Nouvelle Fiche" )
```

L'exécution du bouton &X va faire les choses ainsi

1 met "action Y" dans la liste (->a)
2 met "données:factures" dans la liste (->b)
3 met "action Z" dans la liste (->c)

à ce moment là, la fonction étant terminée, le programme va reprendre les actions en attente.

0 exécute "action Y" (<-a)
1 ajoute "données:articles" à la suite de la liste (->d)

2 exécute "données:factures" (<-b)
3 exécute "action Z" (<-c)

4 ajoute "nouvelle fiche" à la liste (->e)
5 exécute "données:articles (<-d)
6 exécute "nouvelle fiche" (<-e)

Ceci montre que la nouvelle fiche sera créée (par &Z) dans la base des *Articles* alors que l'on ne s'attendrait peut-être à la trouver dans la base des factures.

Table des matières