> Numéro 23

> ALLO SUPPORT

> Dans ce numéro

Cliquez sur un article du sommaire pour vous y rendre directement… et revenez au sommaire à tout moment !

> Enterprise Guide version 4 : des nouveautés au niveau des applications stockées > 1

> SAS/IML® Workshop 2.1 > 7

> Quelques nouveautés SAS®9 de la procédure LOGISTIC > 8

> A la découverte de la procédure INFOMAPS… > 10

> L'ARM pour contrôler vos applications de SAS/BASE A OLAP SERVER > 12

> Ftecnews-L

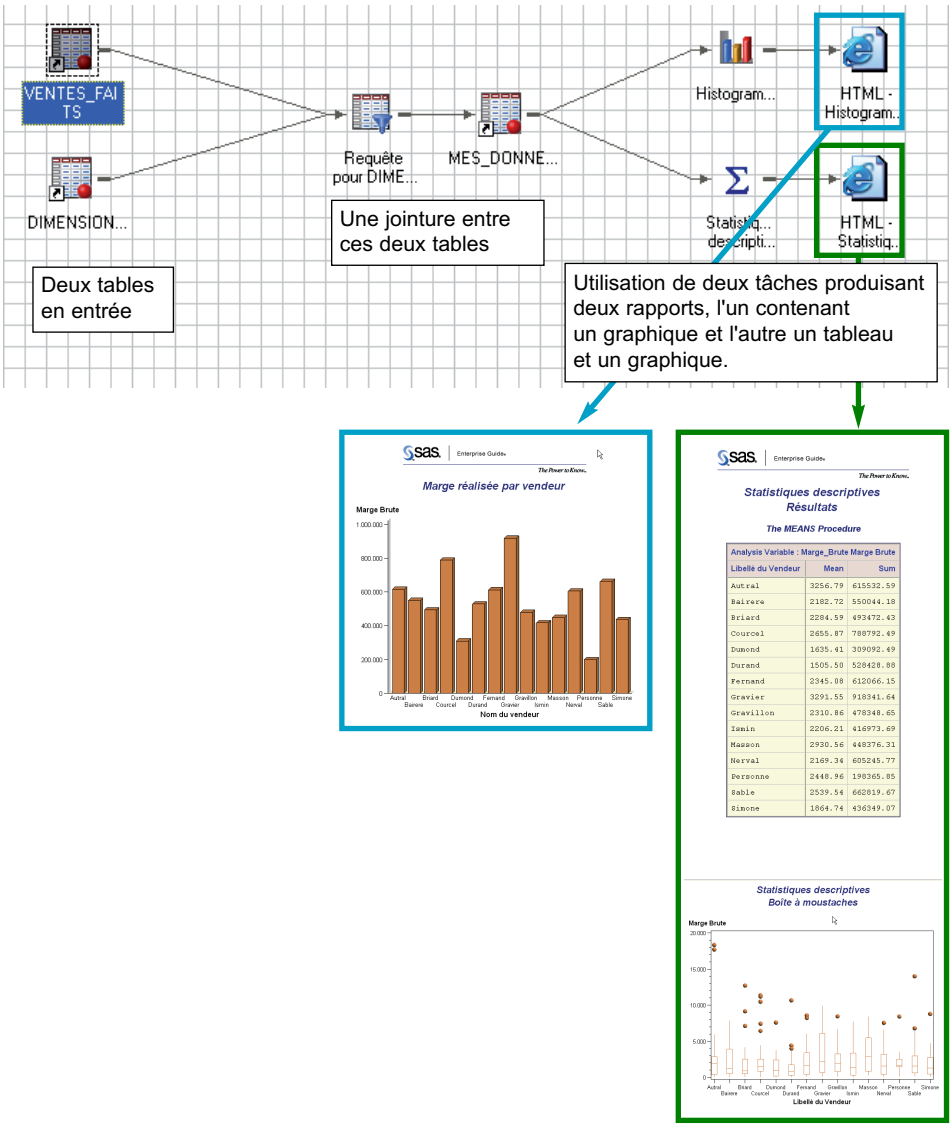
Ftecnews-L, la lettre électronique de l'équipe du Support Clients SAS, permet de diffuser rapidement toute nouvelle information technique vers les utilisateurs des solutions SAS ou/et aux clients francophones.
Avec ce mode de diffusion privilégié, vous serez informés en avant-première de la disponibilité des nouvelles versions, des correctifs logiciels et des nouvelles documentations techniques (TS).
L'abonnement est simple et gratuit : il vous suffit d'envoyer un message électronique à listserv@vm.sas.com en précisant dans le texte du message l'instruction subscribe ftecnews-l suivie de votre prénom et de votre nom.
Ex : subscribe ftecnews-l Jean Dupond.
Cette liste de distribution vient en complément des listes déjà existantes sur notre site Internet (TSNEWS-L, SNOTES-L).
Si vous souhaitez recevoir les éditions futures uniquement via ce support, merci d'en informer Fabienne Vermast : frafav@fra.sas.com.

The Power to Know®

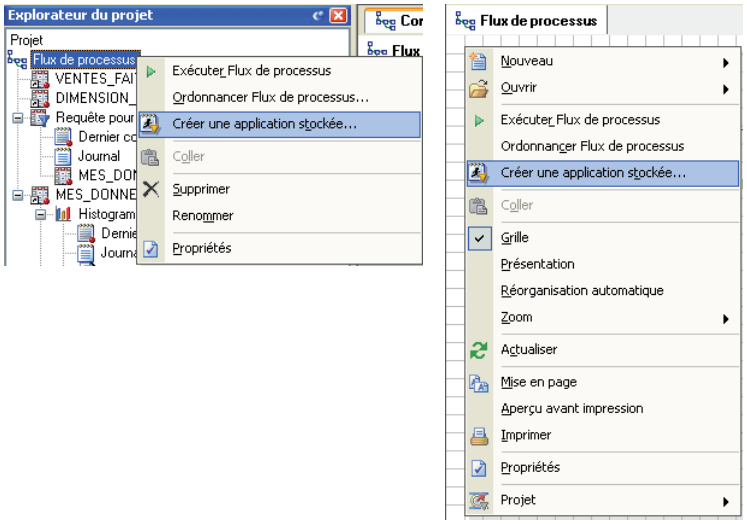
Enterprise Guide version 4 : des nouveautés au niveau des applications stockées

Avec la version 4.1 d'Enterprise Guide, de nombreuses améliorations sont disponibles au niveau de la gestion des applications stockées.
Nous allons décrire ces nouveautés via la mise en œuvre d'un exemple.

1- Prenons l'exemple du flux de processus, créé dans Enterprise Guide et présenté ci-dessous.



Ce processus simple va nous permettre d'illustrer l'un des apports majeurs de la version 4, la possibilité de créer une application stockée à partir de plusieurs tâches. Jusqu'à présent, ce n'était en effet possible qu'à partir d'une tâche. Si vous aviez cinq tâches et souhaitiez créer une seule application stockée, une opération manuelle était nécessaire. Ce n'est donc plus le cas. Pour que toutes les tâches du flux soient intégrées dans votre application stockée, il suffit de cliquer soit dans l'explorateur de projet, soit dans la fenêtre conception du projet.

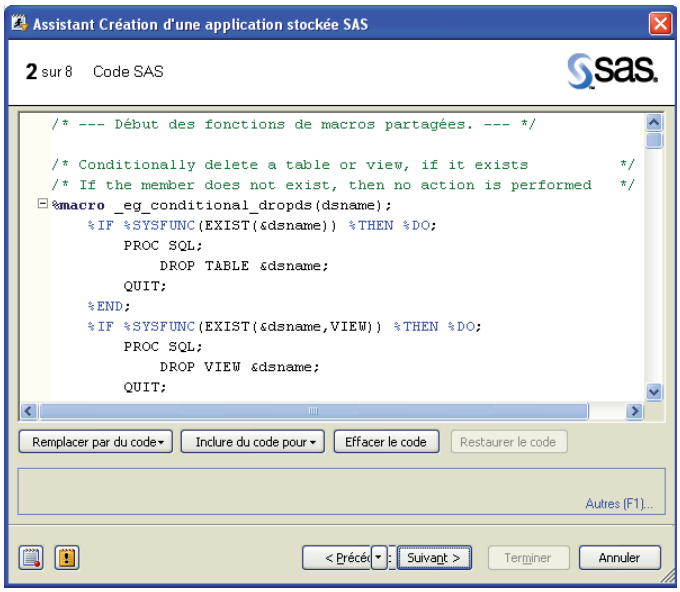


2- Cliquer sur « Créer une application stockée » démarre un assistant qui, en huit étapes, permet de renseigner toutes les informations nécessaires.
Ces étapes sont sensiblement identiques à celles présentes dans la version précédente à l'exception de la deuxième et de la quatrième. L'interface utilisateur a ici été simplifiée et améliorée sur les deux points gérés par ces deux étapes. Nous allons maintenant revoir le déroulement des huit étapes, en nous attardant davantage sur les deux citées précédemment.

1ère étape

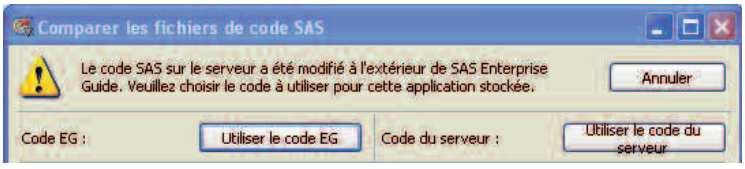
Le nom donné à l'application stockée sera repris pour générer le nom du programme SAS associé, avec des _ à la place des espaces.

2ème étape



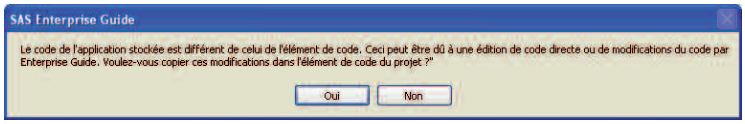
Cet écran est nouveau. Il permet de personnaliser plus facilement qu'auparavant le code SAS associé à l'application stockée et de le modifier. Le bouton « inclure du code pour » permet d'intégrer au code des informations :
- macros de l'application stockée. Il s'agit de l'appel des macro-procédures %stpbegin et %stpend. Leur insertion dans le code est nécessaire pour que l'application stockée puisse s'exécuter correctement. Donc si elle est créée à partir d'une tâche Enterprise Guide, il faut conserver cette option cochée. Si elle est basée sur un code SAS, tout dépend si l'appel de ces macros a déjà été prévu ou non.
- macro-variables globales. Il s'agit des macro-variables définies et utilisées dans les tâches Enterprise Guide, telles que _SASSERVERNAME (nom du serveur SAS), apparaissant par défaut dans les pieds de page.
- instruction libname (voir l'étape 5 pour plus de détails).
- définition de l'application stockée. Des informations sont ajoutées à la fin de l'application stockée dont la définition au niveau des métadonnées. Cette option est intéressante pour déplacer ce code vers une autre machine et pour ainsi recréer l'application stockée. Cette approche est à considérer dans des cas ponctuels, pour migrer une application stockée d'un environnement vers un autre. Dans le cas d'une migration complète, l'utilisation du BI Manager est conseillée.

Lorsque l'application stockée est créée à partir d'un code SAS, un mécanisme est mis en place pour répercuter les changements faits au niveau du programme, que ce soit dans le nœud code ou dans l'application stockée. Ainsi, si le code est modifié directement dans le programme SAS, ce message est affiché à l'ouverture de l'application stockée :



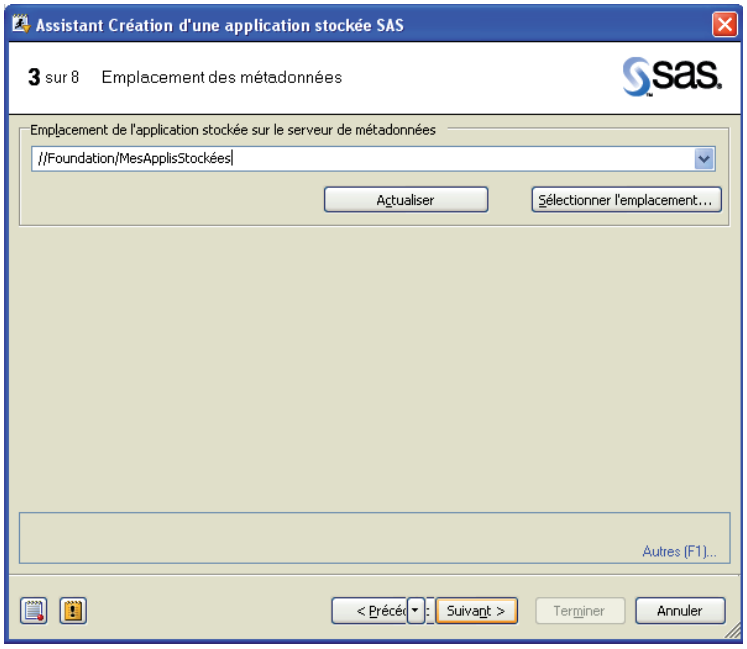
RETOUR SOMMAIRE

Et si l'application stockée est ouverte et le code modifié (donc via le gestionnaire des applications stockées), à son enregistrement, il vous est proposé de modifier l'élément code (pointant sur le même programme SAS) présent dans le projet :



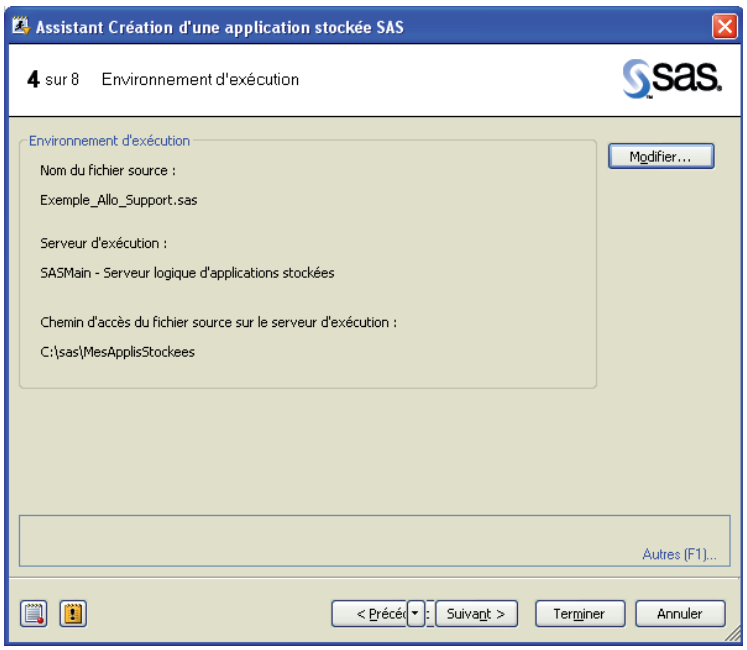
Ce mécanisme vous assure de conserver une cohérence entre l'élément code du projet et le code de l'application stockée.

3ème étape



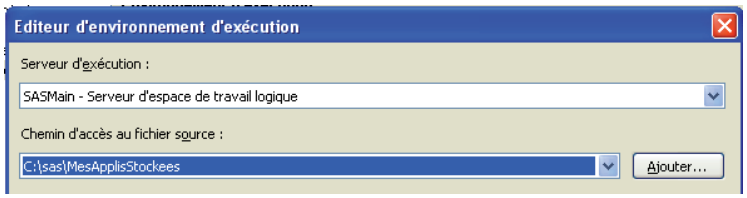
Cet écran permet de choisir où l'application stockée sera enregistrée au niveau des métadonnées. Nous retrouvons ici les dossiers définis dans le plug-in BI Manager de la console d'administration SAS. Dans le cas où vous souhaitez exécuter votre application stockée depuis Web Report Studio, il faut l'enregistrer dans le dossier BIP Tree\ReportStudio\Shared\Reports\StoredProcess (directement ou dans l'un de ses sous-répertoires, que vous devrez créer au préalable dans la console d'administration).

4ème étape



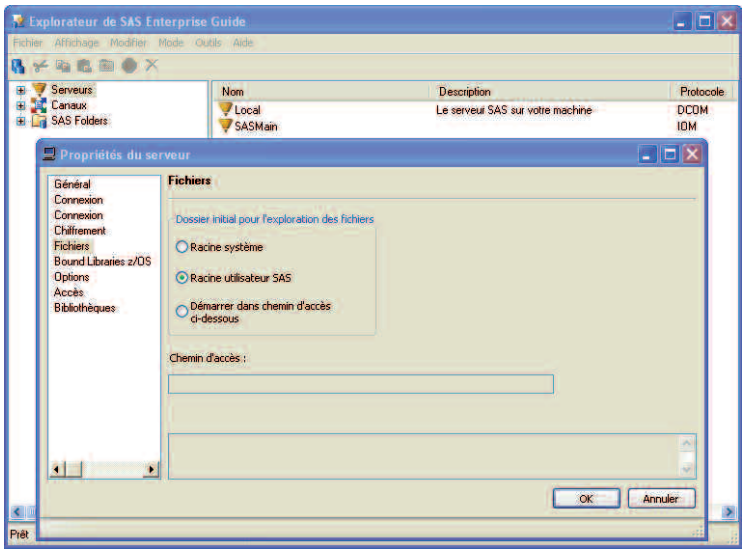
A la différence de la version précédente, vous pouvez constater que les trois informations présentes dans cet écran sont pré-remplies. Enterprise Guide garde en mémoire les informations fournies la première fois que vous créez une application stockée concernant le serveur d'exécution (généralement, c'est le serveur d'applications stockées qui est choisi) et le répertoire physique sur le serveur où sera enregistré le programme SAS. Pour information, ces éléments sont stockés dans un fichier localisé au niveau du profil utilisateur, dans le sous-répertoire Application Data\SAS\Enterprise Guide\4 et s'appelle EGOptions.xml. Pour le nom du programme SAS, nous avons vu dans l'étape 1 qu'il est généré automatiquement. Ces trois paramètres peuvent néanmoins être changés aisément en cliquant sur le bouton « Modifier… ».

Il est également intéressant de noter qu'il est désormais possible de créer un nouveau répertoire physique, où sont stockés les programmes SAS créés, directement depuis l'écran accessible en cliquant sur le bouton « Modifier… » puis sur le bouton « Ajouter… » au niveau du « chemin d'accès au fichier source ». Pour rappel, dans la version précédente d'Enterprise Guide, cette action n'était pas possible et il fallait obligatoirement procéder à cette manipulation dans la console d'administration SAS. Voici une illustration de l'interface :



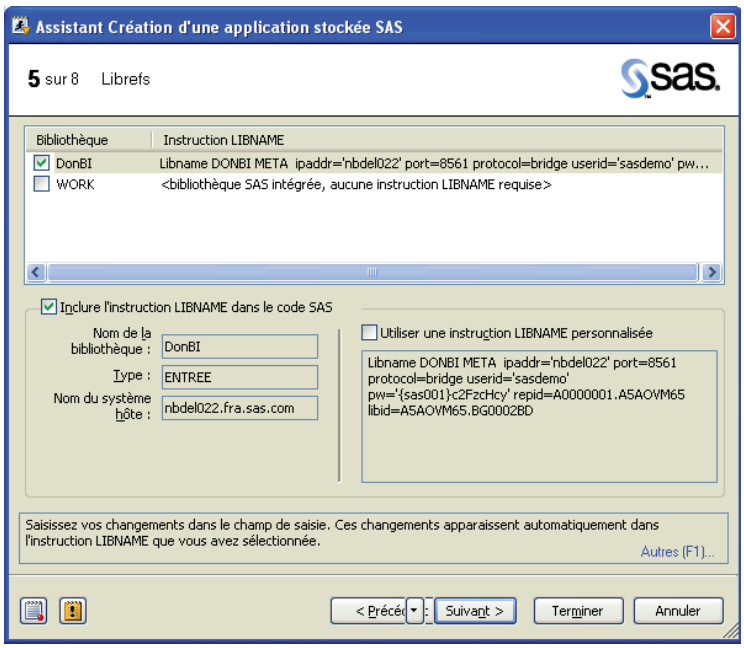
RETOUR SOMMAIRE

L'utilisateur peut uniquement choisir une arborescence de répertoires autorisés par l'administrateur SAS.
Ce dernier effectue cette action dans l'explorateur d'Enterprise Guide (interface remplaçant l'administrateur d'Enterprise Guide et accessible via le menu Outils) :
- il se place au niveau du serveur SASMain et en édite les propriétés.
- l'onglet Fichiers permet de sélectionner le dossier initial pour l'exploration des fichiers.



Par défaut, cette option est positionnée à « Racine utilisateur SAS », donc au répertoire correspondant à la « sasuser ». Par conséquent, l'utilisateur ne verra que ce répertoire et ses sous-répertoires.

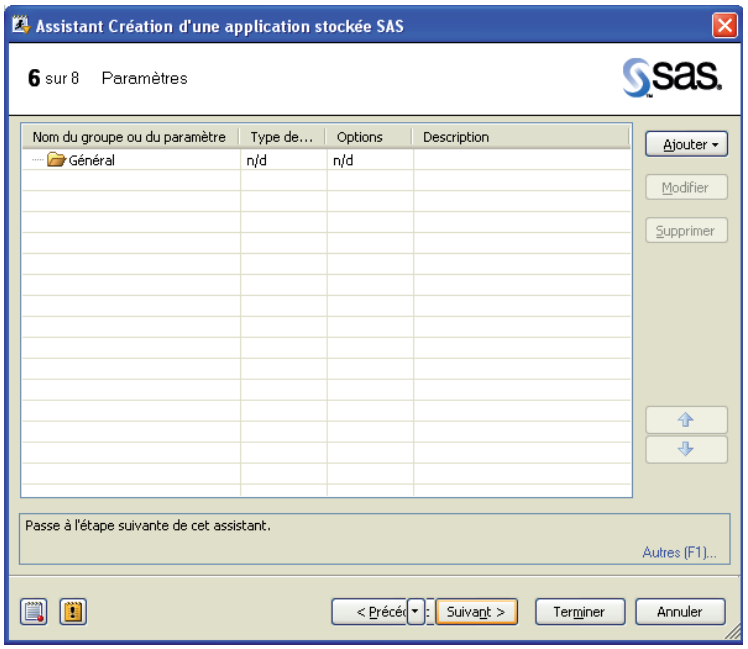
5ème étape



Dans cet écran sont listées les bibliothèques utilisées dans l'application stockée, avec leur définition.
Deux approches sont possibles :
- intégrer cette définition au code de l'application stockée, en cochant les cases sur l'écran ci-dessus.

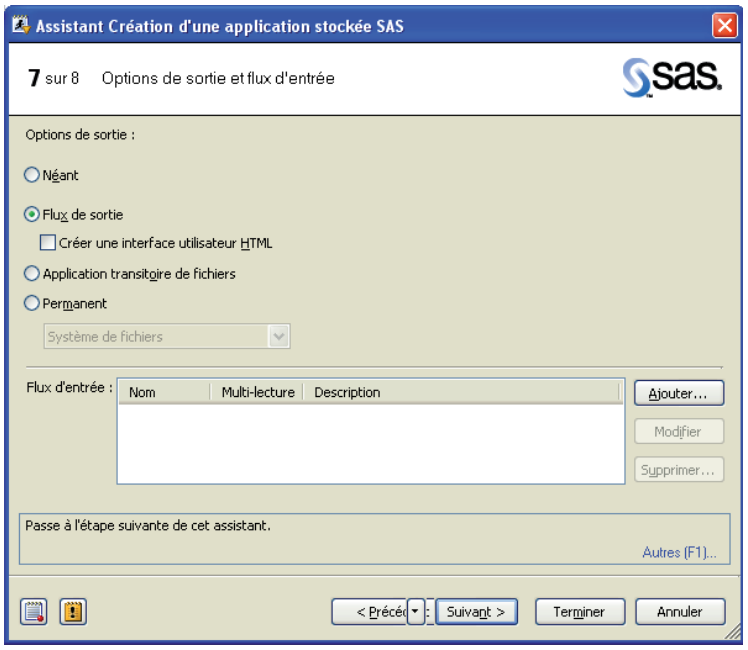
Cette opération sera nécessaire dans toutes les applications stockées.
- paramétrer le serveur d'application stockée pour que les bibliothèques soient toujours connues. Pour ce faire, il faut utiliser un fichier à exécuter au démarrage du serveur d'applications stockées (option -autoexec au niveau de la définition du serveur dans la console d'administration) où les instructions libname adéquates (les mêmes que présentées dans cet écran, donc utilisant le moteur META) seront renseignées.

6ème étape



Cet écran permet toujours de définir des paramètres.

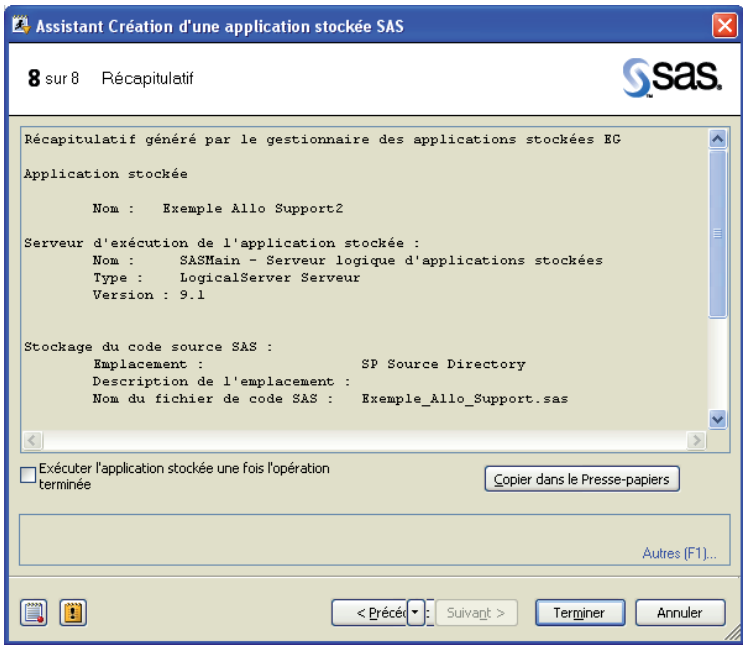
7ème étape



Celui-ci permet de choisir le type de sortie.

RETOUR SOMMAIRE

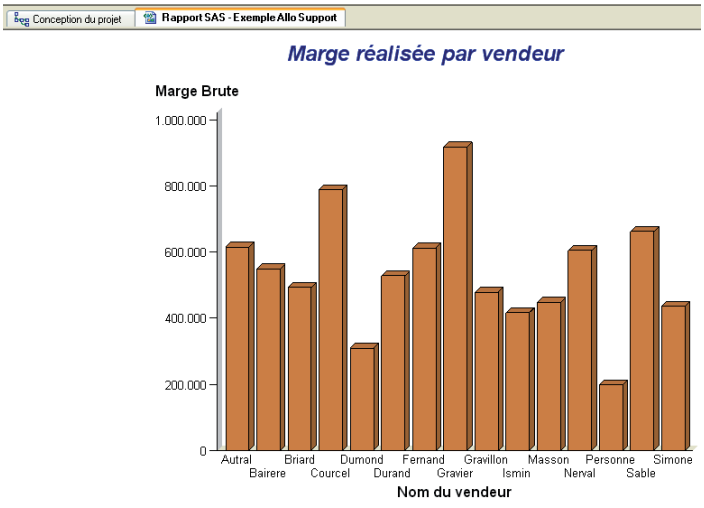
8ème étape



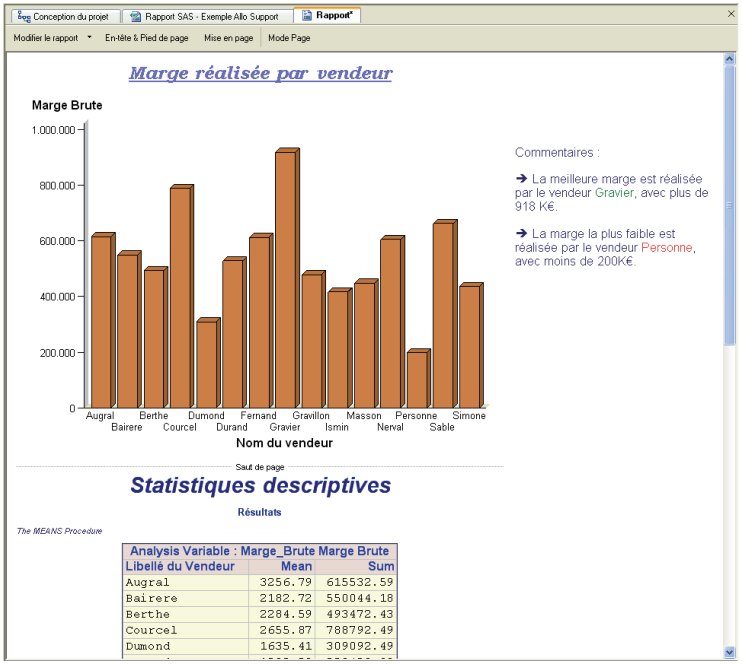
Enfin, tous les éléments renseignés sont résumés dans le dernier écran.

3- Une fois l'application stockée définie et exécutée, un rapport est affiché. Il est désormais possible d'opérer des modifications au niveau de son formatage, polices et couleurs entre autres, et de les conserver, grâce au nouveau type de format de sortie « Rapport SAS » (ou « SAS Report »).

Prenons l'exemple du rapport généré par notre application stockée :

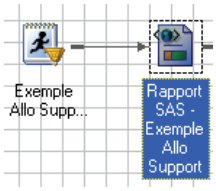


Après des modifications manuelles sur le rapport même, voici un exemple de ce qu'il est possible d'obtenir :

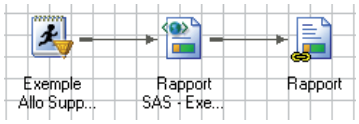


- La couleur du titre du graphique a été modifiée et le texte a été souligné.
- Au niveau des titres du tableau, la taille de la police a été diminuée, la mise en forme italique du second titre a été supprimée et le troisième titre a été aligné à gauche.
- Des commentaires ont été ajoutés dans la partie droite du rapport.

Voyons comment procéder pratiquement.
Dans l'écran de conception du projet, l'application stockée crée un rapport SAS.

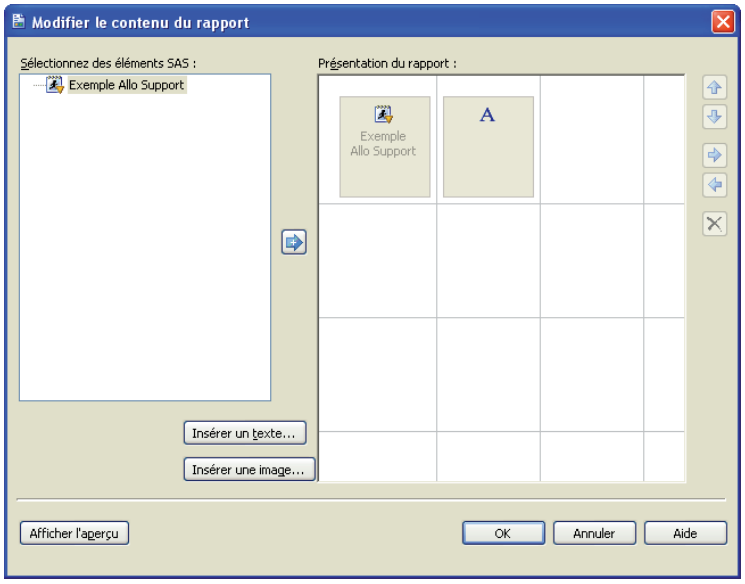


L'étape suivante consiste à créer un nouveau rapport à partir du précédent. L'action à faire est un clic droit sur le premier rapport (Rapport SAS - Exemple Allo Support) et de choisir « créer un rapport ».

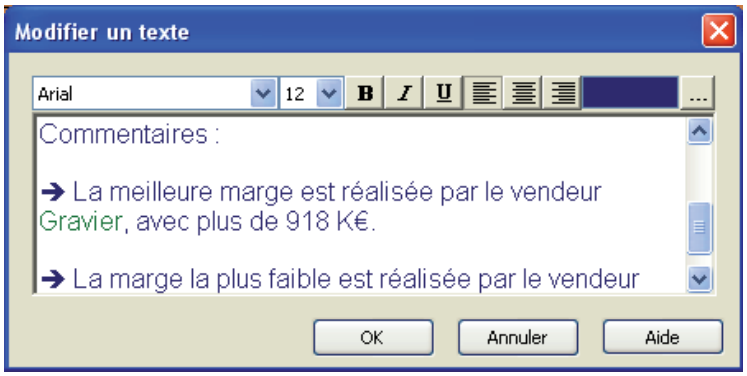


Une fois ce nouveau rapport ouvert, le menu contextuel affiché avec le bouton droit de la souris permet de choisir « Modifier le rapport » puis « Modifier le contenu du rapport

RETOUR SOMMAIRE

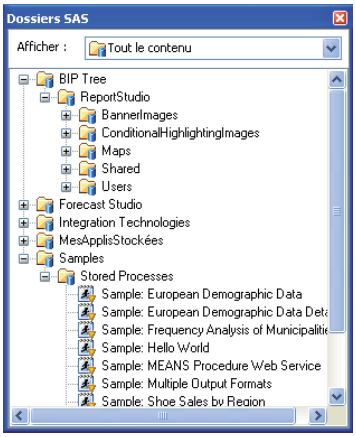


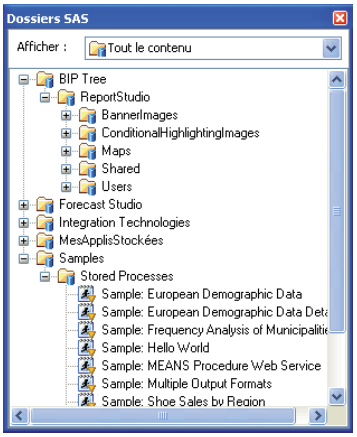
C'est ainsi que des images ou des zones de texte peuvent être ajoutées au rapport. Un double-clic sur une zone de texte initie la fenêtre permettant de saisir ce texte et de le mettre en forme.



Enfin, signalons que lorsque l'application stockée est ré-exécutée, ce rapport est mis à jour, avec les données modifiées (si c'est le cas). Vous pourrez constater, sur les copies d'écran présentées au tout début de ce point, que quelques données ont été changées entre l'exécution du rapport initial, et celle du rapport modifié (deux noms de vendeur ont été modifiés : « Autral » en « Augral » et « Briard » en « Berthe »).

4- La fenêtre utilisée pour lister les applications stockées a changé par rapport à la version 3. Elle s'appelait « Liste des applications stockées ». Désormais, elle porte comme nom « Dossiers SAS » et permet en plus de visualiser les « information maps » et projets Enterprise Guide enregistrés dans les métadonnées. Dans les deux versions, l'insertion de cette fenêtre se déclenche via le menu « Affichage ». Pour ajouter une application stockée à un projet, il suffit de double-cliquer dessus depuis la fenêtre « Dossiers SAS ».





En résumé, les nouveautés au niveau des applications stockées avec la version 4.1 d'Enterprise Guide évoquées dans cet article sont :
- la possibilité de les créer à partir d'un processus complet (donc de plusieurs tâches)
- la modification de l'interface utilisateur pour simplifier certaines étapes de l'assistant
- les améliorations au niveau de l'édition du code et de sa modification
- la possibilité de rafraichir le rapport produit, grâce au format « SAS Report », sans perdre les changements opérés au niveau du formatage (polices, couleurs, justification, ajout de texte)
- la modification de la fenêtre permettant de naviguer dans les métadonnées pour lister les applications stockées (fenêtre « Dossiers SAS »)

Karine Chrillesen

RETOUR SOMMAIRE

SAS/IML® Workshop 2.1

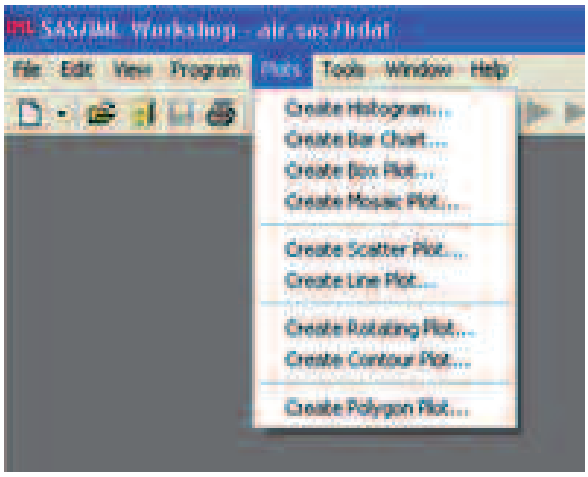
SAS/IML® Workshop est une interface graphique permettant d'explorer les données de façon interactive, d'écrire, d'exécuter et de déboguer des programmes IML et IMLPlus. IML Plus est un langage de programmation plus complet que le langage IML. Il permet, contrairement au langage IML, de faire appel à des procédures SAS et à des fonctions de langages externes tels que C, Fortran ou Java. SAS/IML® workshop s'installe sur Windows.

Installation
Cette interface est disponible gratuitement pour toutes licences comprenant le module SAS/IML®, depuis notre site internet, à partir de l'adresse suivante :
http://support.sas.com/rnd/app/da/workshop.html
- Download (IMLWS21.exe)

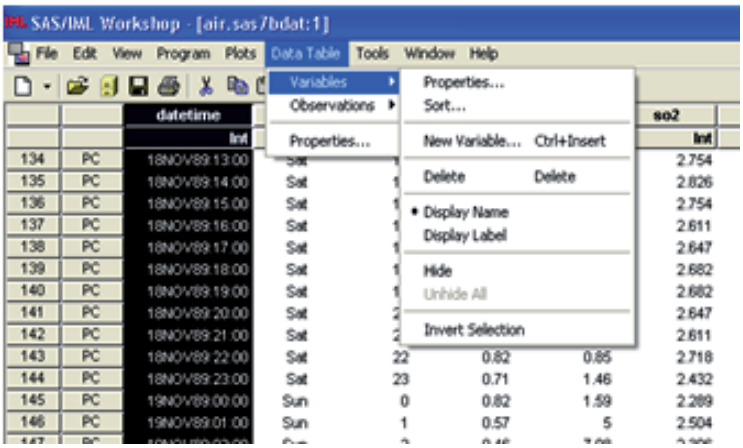
Remarque :
Il est important d'accepter au préalable le contrat de licence et de lire le document Readme.pdf afin de vérifier les pré-requis système que nécessite cette interface avant de lancer le téléchargement.

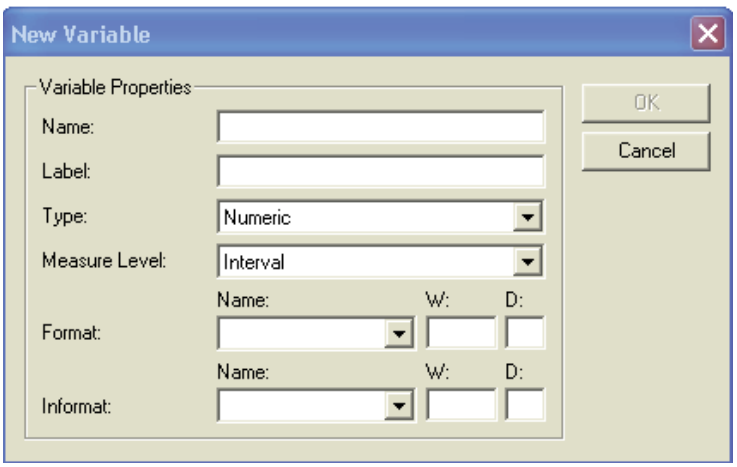
Pré-requis
Les pré-requis système sont les suivants :
• Système d'exploitation:
  • Windows 98
  • Windows Me
  • Windows NT 4.0 (Service Pack 5 ou plus)
  • Windows 2000 (Service Pack 2 ou plus)
  • Windows XP
Les pré-requis au niveau du Système SAS sont :
• Version 8.2 ou 9.1, installée sur la même machine que IML Workshop et incluant au minimum les modules SAS/Base et SAS/IML®

Objectifs
IML Workshop permet :
- d'analyser des données en utilisant des graphiques interactifs : histogramme, boite à moustache, courbe, nuage de points…



- d'exécuter des procédures SAS à partir d'un programme IMLPlus
- de lire / créer des tables SAS
- d'ajouter / supprimer / renommer des variables de façon interactive





- de modifier les propriétés d'une variable
- de supprimer des observations
- d'exécuter vos propres programmes C, Fortran, ou Java
- de développer des programmes d'analyse

Exemple d'utilisation
Ce programme illustre l'appel d'une procédure SAS, la procédure REG, à travers un programme IMLPlus, dans IML Workshop.
Cet exemple utilise :
- des instructions IML pour créer deux matrices IML
- des instructions IML pour sauvegarder des matrices dans des tables SAS
- des instructions SUBMIT pour exécuter une procédure REG afin de générer une régression linéaire sur les données
- des instructions IML standards pour lire les sorties de la procédure REG
- les classes IMLPlus DataObject et LinePlot pour tracer la courbe des données et celle de la régression (la liste et le descriptif de ces classes sont disponibles dans l'aide de IML Workshop : > Help > Help topics > IMLPlus Class reference)

Notez que la procédure REG nécessite le module SAS/STAT®.

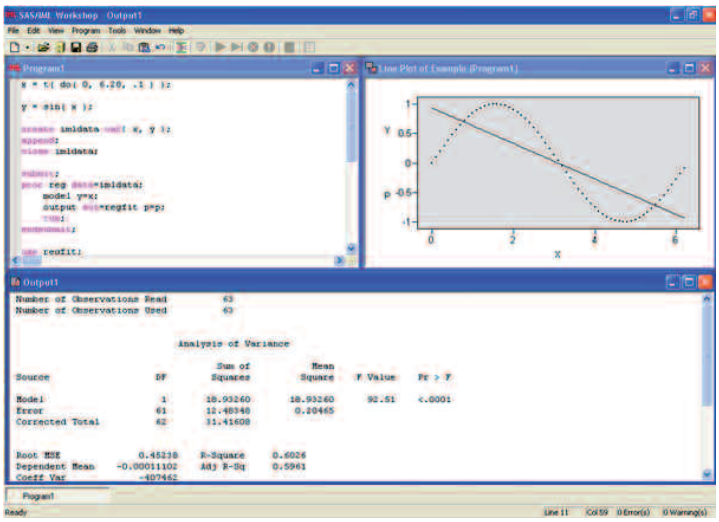
RETOUR SOMMAIRE

Programme	Description

x = t(do(0, 6.28, .1));
y = sin(x);
 	
create imldata var{ x, y };
append;
close imldata;
 	
submit;
proc reg data=imldata;
model y=x;
output out=regfit p=p;
run;
endsubmit;
 	
use regfit;
read all var {p} into p;
close regfit;
 	
declare DataObject dobj;
dobj = DataObject.Create("Example");
dobj.AddVar("X", x);
dobj.AddVar("Y", y);
dobj.AddVar("P", p);
 	
declare LinePlot plot;
plot = LinePlot.Create(dobj, "X", {"Y" "P"}, false);
plot.ConnectPoints("Y", false);
 	
plot.ShowPoints("P", false);
plot.ShowWindow();
	Création de deux matrices avec une instruction IML
	
	Sauvegarde des matrices dans la table IMLDATA avec des instructions IML
	
	Appel de la procédure REG pour générer une régression linéaire de la table précédemment créée, avec l'instruction SUBMIT
	
	Lecture des sorties de la procédure REG avec des instructions IML
	
	Utilisation de la classe DataObject de IMLPlus pour créer le graphique
	
	Utilisation de la classe LinePlot de IMLPlus pour tracer la droite de régression

Pour exécuter ce programme, il faut
- soit cliquer sur la flèche Bleue
- soit appuyer sur la touche F5
- soit sélectionner dans le menu « Program » et cliquer sur « Execute ».

Résultats :



Les résultats (OUTPUT) de la procédure REG ainsi que les graphiques produits s'affichent dans IML Workshop, après exécution.

IML Workshop est une interface conviviale et très complète. Pour plus d'informations sur l'utilisation de cet outil, vous pouvez consulter l'aide directement disponible depuis IML Workshop : Help > Help Topics

Aurélie GILBERT

Quelques nouveautés SAS®9 de la procédure LOGISTIC

Avec SAS®9, la procédure LOGISTIC continue à s'enrichir en mettant à disposition deux nouvelles instructions SCORE et STRATA.

Comme son nom l'indique, l'instruction SCORE permet le calcul des prédictions (scoring) d'après un modèle donné, alors que l'instruction STRATA permet de réaliser des régressions logistiques conditionnelles.
En voici un aperçu.

L'instruction SCORE :
Il existe 2 méthodes pour utiliser l'instruction SCORE. Pour les illustrer, le jeu de données suivant sera utilisé dans un contexte de régression logistique généralisée:

```
data matable;
   input hopital traitement $ gravite $ @@;
   cards;
1 a none     1 a slight     1 a moderate
1 b none     1 b slight     1 b moderate
1 c none     1 c slight     1 c moderate
1 d none     1 d slight     1 d moderate
2 a none     2 a slight     2 a moderate
2 b none     2 b slight     2 b moderate
2 c none     2 c slight     2 c moderate
2 d none     2 d slight     2 d moderate
3 a none     3 a slight     3 a moderate
3 b none     3 b slight     3 b moderate
3 c none     3 c slight     3 c moderate
3 d none     3 d slight     3 d moderate
```

• **La première méthode** - calcul du modèle et des prédictions dans une seule procédure LOGISTIC:

La table utilisée pour élaborer le modèle est spécifiée dans l'option DATA= de la procédure LOGISTIC, alors que la nouvelle table pour laquelle on souhaite obtenir les prédictions est, elle, spécifiée dans l'option DATA= de l'instruction SCORE.

```
proc logistic data=matable;
   class traitement hopital;
   model gravite(order=data) = traitement hopital / link=glogit;
   score data=a out=out;
   run;

proc print;
   run;
```

Avec la version 8, il était également possible de faire du scoring tout en élaborant le modèle, au sein d'une seule procédure LOGISTIC, mais il était pour cela nécessaire de concaténer la nouvelle table à la première.

RETOUR SOMMAIRE

• **Deuxième méthode** - calcul du modèle et des prédictions dans des procédures LOGISTIC séparées:

Dans un premier temps, le modèle est calculé et stocké dans la table définie par l'option OUTMODEL=

```
proc logistic data=matable outmodel=model;
   class traitement hopital;
   model gravite(order=data) = traitement hopital / link=glogit;
   run;
```

Le modèle stocké en table peut ensuite être appliqué à un nouveau jeu de données, grâce à l'option INMODEL= et à l'instruction SCORE :

```
proc logistic inmodel=model;
   score data=a out=out;
   run;

proc print;
   run;
```

Remarque:
La table générée par l'option OUTMODEL, contenant les informations du modèle, est une table destinée uniquement au scoring au sein de la procédure LOGISTIC.

Pour utiliser les informations du modèle en dehors de la procédure LOGISTIC, les méthodes utilisées dans les versions antérieures sont toujours disponibles, notamment les options INEST et OUTEST.

L'instruction STRATA :
Les régressions logistiques conditionnelles sont beaucoup utilisées dans le cas d'études cas-témoins (case-control). Au lieu de modéliser, par exemple, sur une population le fait d'être ou non malade par différents facteurs qualitatifs ou quantitatifs, on associe (apparie) à un cas malade un ou plusieurs cas témoins non-malades mais de profil similaire (même âge, même sexe,...). Cela donne lieu à une régression logistique par profil.
La procédure LOGISTIC permettait jusqu'à présent de traiter les cas d'appariement 1 :1 (un cas pour un témoin). Elle est à présent, grâce à l'instruction STRATA, en mesure de traiter les cas de données appariées m :n (m cas pour n témoins).

Ainsi, en reprenant l'exemple évoqué plus haut, une régression logistique par profil âge pourra être réalisée en spécifiant la variable Age dans l'instruction STRATA :

```
data castemoin;
input age cas $ x1 x2;
datalines;
41 cas 3 5
41 temoin 4 9
41 temoin 1 4
42 temoin 3 9
42 cas 5 8
43 cas 7 9
43 temoin 2 3
43 temoin 4 7
43 temoin 2 4
…
;
run;

proc logistic data=castemoin;
class cas;
strata age;
model cas=x1 x2;
run;
```

L'instruction STRATA peut contenir autant de variables qu'il est nécessaire pour définir le profil désiré.

Un exemple complet de procédure LOGISTIC avec l'instruction STRATA est présenté dans la documentation en ligne à l'adresse suivante :
http://support.sas.com/onlinedoc/913/getDoc/en/statug.hlp/logistic_sect63.htm
Cet exemple est une étude de cas-témoins 1:1, mais il s'applique aussi à une étude cas-témoins m :n

A noter que la procédure PHREG permet également de réaliser les régressions logistiques conditionnelles m :n, mais la procédure LOGISTIC possède l'avantage de pouvoir utiliser les instructions CLASS et CONTRAST.

Une question subsiste :
Si les instructions SCORE et DATA sont opérationnelles en SAS®9, pourquoi apparaissent-elles en rouge dans l'éditeur ?

Aucune crainte, même en rouge, ces instructions sont valides. Il s'agit simplement d'une petite anomalie de l'éditeur, dû au fait que ces nouvelles procédures n'ont pas encore été référencées. En attendant qu'elles le soient dans les prochaines versions, il est possible de déclarer ces nouvelles instructions de la façon suivante:
Outils -> Options -> Editeur amélioré -> General -> Mot-clés définis par l'utilisateur -> Ajouter -saisir "score" et « strata ».
Elles seront alors reconnues par l'éditeur et n'apparaîtront plus en rouge.

Ce bref aperçu des nouvelles instructions SCORE et STRATA montre que la procédure LOGISTIC gagne encore en convivialité et en fonctionnalités.

D'autres améliorations ont été apportées à la procédure LOGISTIC en SAS®9, notamment l'ajout de nouveaux paramétrages pour les variables de classe.
Elles sont visibles dans la section « What's New in SAS/STAT® 9 and 9.1 » de la documentation en ligne que je vous invite à consulter :
http://support.sas.com/onlinedoc/913/docMainpage.jsp

Blandine Colas

RETOUR SOMMAIRE

A la découverte de la procédure INFOMAPS…

A partir de SAS® 9.1.3 Service Pack 4, il est maintenant possible de créer des données métiers (information maps) en utilisant la procédure INFOMAPS.
Cette procédure permet également de modifier une information map existante en ajoutant par exemple de nouvelles sources de données physiques ou de nouveaux éléments de données.

Après un rappel sur la définition d'une information map, nous détaillerons dans cet article quelques instructions de cette procédure.
Enfin, un exemple de son utilisation viendra illustrer cet article.

1. Qu'est-ce qu'une information map ?
Une *information map* est une vue métier sur les métadonnées qui permet aux utilisateurs finaux d'exécuter des requêtes sur un entrepôt de données au travers des outils tels que SAS® Web Report Studio, le portail SAS, SAS® Enterprise Guide® 4.1 etc.

2. Syntaxe de la procédure INFOMAPS
• Les arguments nécessaires pour l'instruction proc INFOMAPS
Afin de pouvoir exécuter la procédure INFOMAPS, il est nécessaire de se connecter au serveur de métadonnées au préalable.
Les options METAUSER, METAPASS, METASERVER, METAPORT et METAREPOSITORY seront utilisées à cet effet.

```
proc infomaps metauser="nom_utilisateur"
              metapass="mot_de_passe_utilisateur"
              metaserver="nom_serveur"
              metaport=8561
              metarepository="Foundation";
```

Contrairement aux 5 arguments indispensables ci-dessus, l'argument MAPPATH est optionnel.
Cet argument, permettant de spécifier le dossier de stockage de l'information map dans le serveur WebDav, pourra être déclaré dans d'autres instructions.

```
mappath="/BIP Tree/ReportStudio/Maps"
```

• L'intruction DELETE MAP
Cette instruction permet de supprimer une information map présente dans un répertoire spécifique.

```
delete infomap "donnees clients" mappath="/BIP
Tree/ReportStudio/Maps";
```

Dans le cas où l'option MAPPATH est positionnée au niveau de l'instruction PROC INFOMAPS, il n'est pas nécessaire de la spécifier pour l'instruction DELETE MAP.
La syntaxe devient alors:

```
delete infomap "donnees clients";
```

Il faut noter que cette instruction est très utile pour éviter de dupliquer les éléments de données au sein d'une information map. En effet, si une information map existe déjà et que cette dernière est ouverte avec la procédure INFOMAPS, un élément de données portant le même nom sera alors dupliqué.
Si l'information map indiquée dans l'instruction DELETE INFOMAP n'existe pas, un message d'avertissement est écrit dans le journal.

• L'instruction OPEN INFOMAP
L'instruction OPEN INFOMAP permet d'ouvrir une information map existante ou de créer une nouvelle information map.

```
open infomap "donnees clients" mappath="/BIP
Tree/ReportStudio/Maps";
```

Comme pour l'instruction DELETE MAP, si l'option MAPPATH a été spécifiée auparavant, il n'est pas nécessaire de la rajouter dans l'instruction OPEN INFOMAP :

• L'instruction INSERT DATASOURCE
L'instruction INSERT DATASOURCE permet d'ajouter une table ou un cube OLAP, définis dans les métadonnées, dans l'information map en cours d'ouverture.
Il faut noter que la syntaxe à utiliser est différente selon que nous souhaitons insérer une table ou un cube OLAP.

Dans les deux cas le paramètre SASSERVER doit être spécifié afin d'identifier le serveur d'applications SAS.
Nous spécifierons ensuite le nom de la bibliothèque pour une table ou le nom du schéma OLAP pour un cube.

Insertion d'une table

```
insert datasource sasserver="SASMain"
                  table="Basic Data".CUSTOMER;
```

Où SASMain correspond au serveur d'applications SAS, “Basic Data” à la bibliothèque SAS et CUSTOMER à la table.

Insertion d'un cube

```
insert datasource sasserver="SASMain"
                  cube="SASMain - OLAP Schema".CUBE_
DONNEES_CLIENTS;
```

Où SASMain correspond au serveur d'applications SAS, « SASMain - OLAP Schema » au schéma OLAP et CUBE_DONNEES_CLIENTS au cube.

• L'instruction INSERT DATAITEM
Cette instruction permet de rajouter un nouvel élément de données dans l'information map en cours d'ouverture.

```
insert dataitem column="CUSTOMER".CUST_ID;
```

Pour ce cas précis, nous ajoutons un nouvel élément de données “CUST_ID” correspondant à la colonne CUST_ID de la table CUSTOMER.

RETOUR SOMMAIRE

Il est bien sûr possible de créer un nouvel élément de données dont l'expression se base sur un calcul entre éléments de données, par l'intermédiaire de l'option EXPRESSION.

```
insert dataitem
expression="<<CUSTOMER.VENTES1>>+<<CUSTOMER.VENTES2>>"
  name="Ventes_total "
  type=numeric
  desc="Total des ventes" classification=measure
format="Comma8.2";
```

Dans l'exemple ci-dessus, nous créons un élément de données « Ventes_total » de type mesure qui correspond à la somme des éléments de données VENTES1 et VENTES2.

• L'instruction SAVE

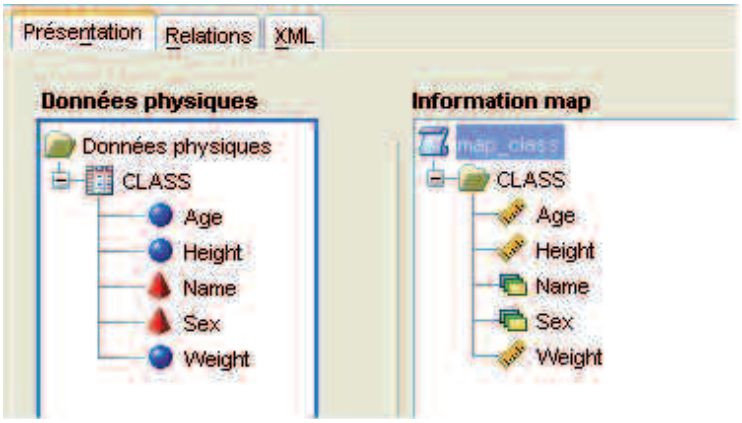
L'instruction SAVE permet de sauvegarder *l'information map* ouverte.
En effet *l'information map* ouverte est mise en mémoire et il est nécessaire d'utiliser cette instruction afin de la sauvegarder dans les métadonnées.
Si aucun nom n'est spécifié pour *l'information map* dans cette instruction, le nom de *l'information map* en cours est alors utilisé..
Si aucun chemin n'est spécifié pour l'enregistrement de *l'information map,* cette dernière est sauvegardée dans le répertoire utilisé par l'option MAPPATH de l'instruction OPEN INFOMAP, ou si cette option n'est pas présente dans l'instruction OPEN INFOMAP, dans le répertoire utilisé par l'option MAPPATH de l'instruction PROC INFOMAPS.

3. Exemple d'utilisation de la procédure INFOMAPS

L'exemple de code suivant permet de créer une information map « map_class » sur le serveur nbdel051 dans le répertoire /BIP Tree/ReportStudio/Maps, en insérant les métadonnées de la table SASHELP.CLASS et toutes ses colonnes.

```
/* Connexion au serveur de métadonnées*/
proc infomaps metauser="sasdemo"
              metapass="xxxxxx"
              metaserver="nbdel051"
              metaport=8561
              metarepository="Foundation";

/* Supprime l'information map pour éviter de dupliquer les éléments de données*/
delete infomap "map_class" mappath="/BIP
Tree/ReportStudio/Maps";

/* Ouvre une nouvelle information map*/
open infomap "map_class" mappath="/BIP
Tree/ReportStudio/Maps";

/* Insère la table SASHELP.CLASS et toutes ses colonnes*/
insert datasource sasserver="SASMain"
table="SASHELP".CLASS _all_;

/* sauvegarde l'information map dans les métadonnées*/
save;

run;
```

Voici un aperçu de l'information map, créée par le code ci-dessus, dans SAS® Information Map Studio :



D'autres instructions ainsi que de nombreuses options sont disponibles dans les documentations suivantes :
- l'aide en ligne
- ou la SAS OnlineDoc® disponible à l'adresse : http://support.sas.com/onlinedoc/913/docMainpage.jsp.

Sylvain BERTIN

RETOUR SOMMAIRE

L'ARM pour contrôler vos applications, de SAS/BASE A OLAP SERVER

Le suivi des performances est une problématique récurrente pour les utilisateurs d'applications SAS.
L'ARM en fournissant des logs complètes sur la disponibilité et la performance des applications est une solution possible à mettre en place.

L'objectif de cet article est de vous décrire les options et macro-programmes mis à disposition avec l'ARM en SAS®9. Vous trouverez également un exemple d'implémentation de l'ARM avec Olap Server.

1. Qu'est-ce que l'ARM ?
L'ARM, Application Response Measurement, est disponible depuis la version 8.2. L'ARM permet de contrôler la disponibilité et la performance des transactions.
Ces informations sont stockées dans un journal.

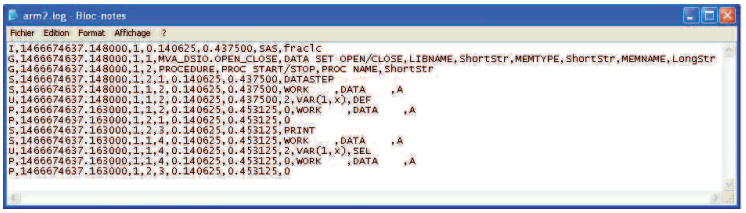
En utilisant l'ARM, vous avez entre autres, la possibilité de déterminer :
- Quels sont les temps de réponse d'une application
- Qui utilise une application
- Quelles requêtes sont utilisées
- Quels sont les serveurs utilisés
- Le temps de chargement des entrepôts de données
- Pourquoi une application n'est pas disponible

2. Fonctionnement :
Il existe un ensemble de fonctions ARM que l'on peut appeler depuis une application. SAS implante ces fonctions API au travers d'un agent ARM.
SAS propose en plus un ensemble de macros et d'options système SAS qui permet de gérer l'environnement ARM et d'écrire dans un journal les transactions SAS.
L'ARM a été construit de façon à avoir un impact minimum sur les applications..

Les options système SAS :
- ARMLOC : chemin complet de stockage du journal ARM.
- ARMAGENT : permet de spécifier un agent ARM autre que celui de SAS (par défaut).
- ARMSUBSYS : choix du niveau de détail des transactions SAS à analyser.
  Syntaxe : ARMSUBSYS=(ARM_NONE | ARM_ALL | *subsystem1 <item1 <item2 <...> > > <, subsystem2 <item1 <item2 <...> > >*><OFF>)
- Le détail des options est le suivant :
  - ARM_NONE : aucune transaction n'est enregistrée dans le journal ARM.
  - ARM_ALL : toutes les transactions sont enregistrées dans le journal ARM.
  - ***subsystem*** : spécifie un groupe de transactions SAS internes.
  - ARM_DSIO : informations sur les I/O de tables SAS.
  - ARM_IOM : informations sur le serveur IOM.
  - ARM_PROC : informations sur les procédures et étapes DATA utilisées.
  - ARM_OLAP_SESSION : collecte les informations de la session SAS® OLAP Server.
  - ***item*** : spécifie le type et la quantité de transaction à écrire dans le journal ARM.
  - **OFF** : désactive le groupe de transaction spécifié.

Exemples :
```
options armloc = "c:\arm2.log";
/*active les traces*/
options armsubsys=(arm_all);
data a;
x=1;
run;
proc print;
run ;
```

Aperçu du fichier généré :



Chaque ligne commence par une lettre qui indique quel type d'informations va être stocké sur l'ensemble de la ligne.
Dans cet exemple, la 1ère ligne commence par « I ».
Pour chaque Initialisation d'une session ARM, une telle ligne sera écrite dans le fichier log. Les informations qui suivent ce paramètre sont :
- la date de démarrage de la session (au format datetime)
- un ID d'application incrémenté par session
- « user start time »
- « system start time »
- le nom de l'application
- l'ID de l'utilisateur

Pour plus de détails sur ces options et sur l'interprétation du fichier généré, se reporter au lien suivant :
http://support.sas.com/onlinedoc/913/getDoc/en/lrdict.hlp/a002165157.htm

Les macros SAS :
6 macros SAS ont pour but de générer les appels aux API de l'agent ARM.
En détail :
- %**ARMINIT** : génère un appel à l'API ARM_INIT, qui nomme l'application, les utilisateurs et initialise l'environnement ARM.
- %**ARMGTID** : génère un appel à l'API ARMGTID, qui nomme la transaction.
- %**ARMSTRT** : génère un appel à l'API ARMSTRT, qui signale le début de la transaction.
- %**ARMUPDT** : macro optionnelle qui génère un appel à l'API ARMUPDT, qui fourni des informations complémentaires sur l'état d'avancement de la transaction.
- %**ARMSTOP** : génère un appel à l'API ARMSTOP, qui signale la fin de la transaction.

RETOUR SOMMAIRE

- %**ARMEND** : génère un appel à l'API ARMEND, qui finalise l'environnement ARM.

Exemple d'utilisation de ces macros et options système :

```
   /* mise en place de l'environnement ARM*/
options
   armagent=sasarmmg
   armsubsys=(arm_all)
   armloc=mylog;

filename mylog 'C:\ARMlog.txt';


%let _armexec=1;

data work1 work2 work3;
do _I_ = 1 to 100;
  output;
end;
run;

%arminit(APPNAME='Sample ARM',APPUSER='Arm
UserID',MACONLY=YES);
%armgtid(TXNNAME='Sample 1',MACONLY=YES);
%armstrt(LEVEL=1,MACONLY=YES);

data one;
   set work1;
run;

%armstop(MACONLY=YES);
%armgtid(TXNNAME='Sample 2',MACONLY=YES);
%armstrt(LEVEL=1,MACONLY=YES);

data two;
   set work2;
run;

%armstop(MACONLY=YES);
%armgtid(TXNNAME='Sample 3',MACONLY=YES);
%armstrt(LEVEL=1,MACONLY=YES);

data three;
   set work3;
run;
%armstop(MACONLY=YES);
%armend(MACONLY=YES);
run;
```

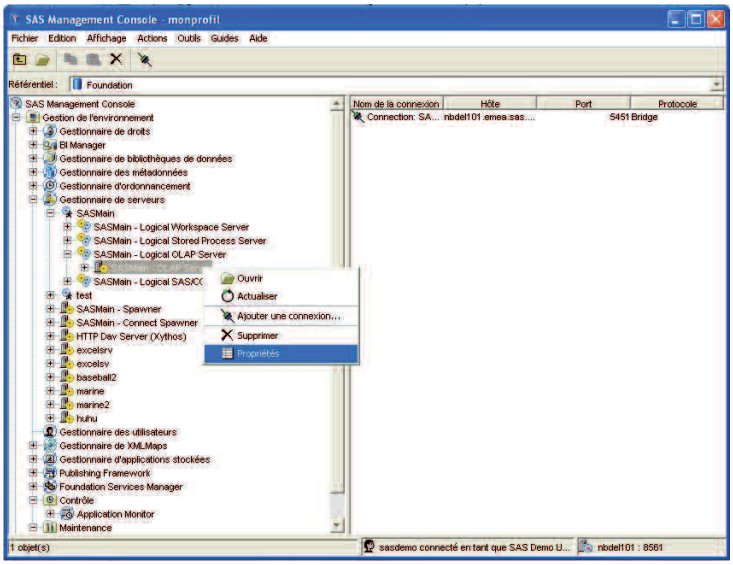
3. Cas particulier de la mise en œuvre de l'ARM avec OLAP Server

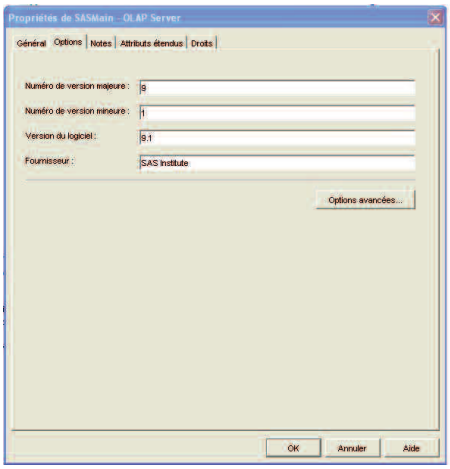
Au niveau Olap Server, l'ARM peut être activé de façon ponctuelle pour contrôler les performances des structures Olap ou pour optimiser la construction des cubes.

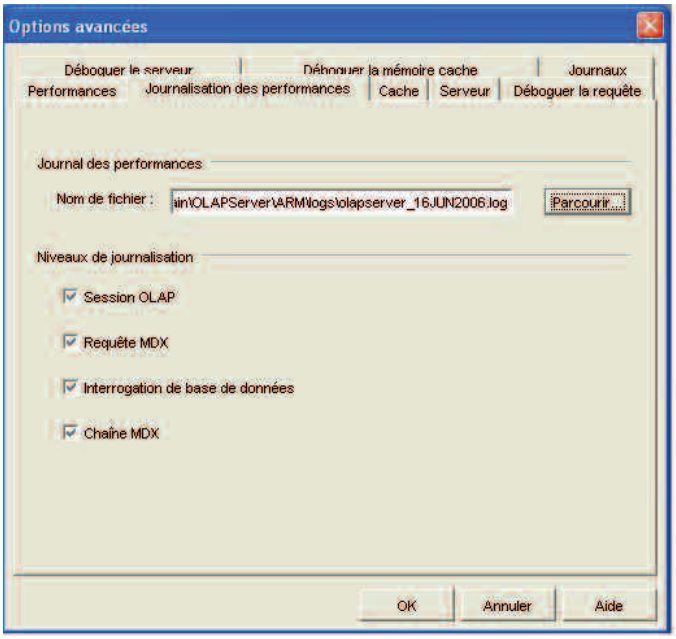
La fonctionnalité ARM peut être mise en place via l'option Avancées dans la définition du serveur OLAP dans la console d'administration (SMC).

Mais, vous avez aussi la possibilité d'utiliser les options SAS : ARMSUBSYS et ARMLOC.

Ci-dessous le détail des paramétrages à effectuer pour activer l'ARM via la SMC







RETOUR SOMMAIRE

Dans ce dernier écran, vous devez spécifier le fichier destinataire du journal des performances ainsi que le répertoire de stockage.
Nous vous recommandons de sauvegarder les logs générées dans un sous-répertoire du serveur OLAP en utilisant la date du jour comme paramètre. Ceci dans un souci d'organisation.

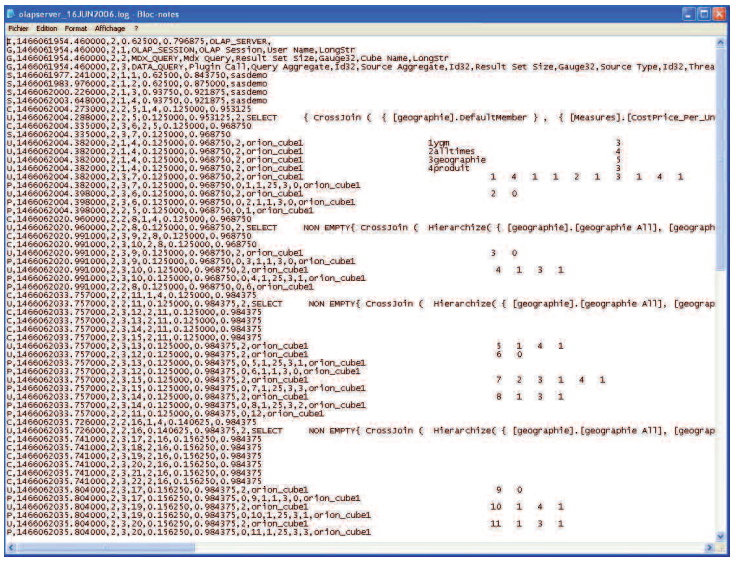
Exemple :
C:\SAS\BIArchitecture\Lev1\SASMain\OLAPServer\arm\logs**olapserver_16JUN2006.log**

Autre point important concernant le répertoire de création de la log, vous devez avoir les droits d'écriture sur ce répertoire.

Vous pouvez ensuite spécifier les différents niveaux de détails des traces.
Pour obtenir des détails sur les 4 niveaux (Session Olap, Requêtes MDX, Interrogation de bases de données, chaînes MDX), veuillez vous référer à l'aide en ligne, au niveau Syntaxe, items, ARM_OLAP_SESSION :
http://support.sas.com/onlinedoc/913/getDoc/en/lrdict.hlp/a002165157.htm

4. Analyse du journal ARM
Le fichier compte-rendu généré par l'ensemble des actions réalisées dans une application se basant sur des structures OLAP contient des informations qu'il faut structurer afin de rendre l'analyse possible.

Comme vous pouvez le constater ci-dessous, la log est très dense en informations :



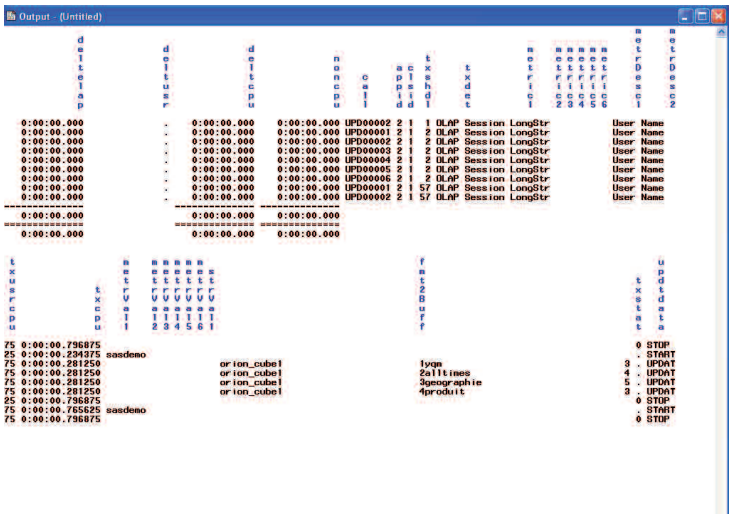
Vous avez donc à votre disposition 3 macro-programmes SAS complémentaires pour aider à l'analyse des données.

ARMCONV permet de convertir une LOG ARM créée à partir de la version SAS®9 en une LOG formatée pour la version 8.2. Ce macro-programme n'est donc utilisé que de manière restreinte.

ARMPROC convertit l'information présente dans le journal ARM en données SAS enregistrées dans 6 tables temporaires (librairie WORK).
ARMJOIN analyse les tables SAS créées avec le macro-programme ARMPROC afin d'alimenter de nouvelles tables avec des informations synthétiques.

Un exemple de programme permettant d'exécuter ces actions et de produire un aperçu rapide des performances OLAP (temps CPU)

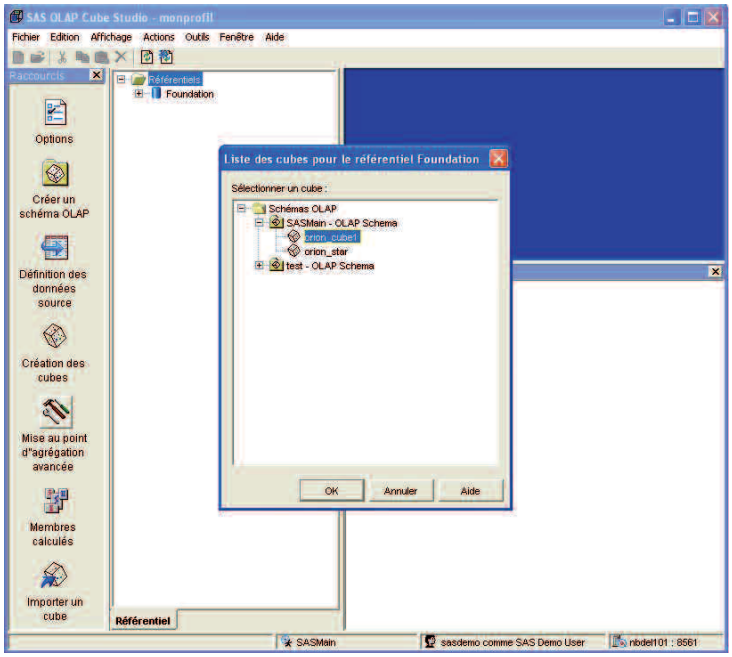
```
/* Production des tables ARM */

filename armlog
"C:\SAS\BIArchitecture\Lev1\SASMain\OLAPServer\arm\logs\olapserver_16JUN2006.log";
%armproc;
%armjoin;
run;

/* Restitution des performances */

proc sort data=updtview out=test;
  by txname ;
run ;

proc print data=test ;
   sum deltelap deltcpu noncpu ;
   by txname ;
   sumby txname ;
run ;
```



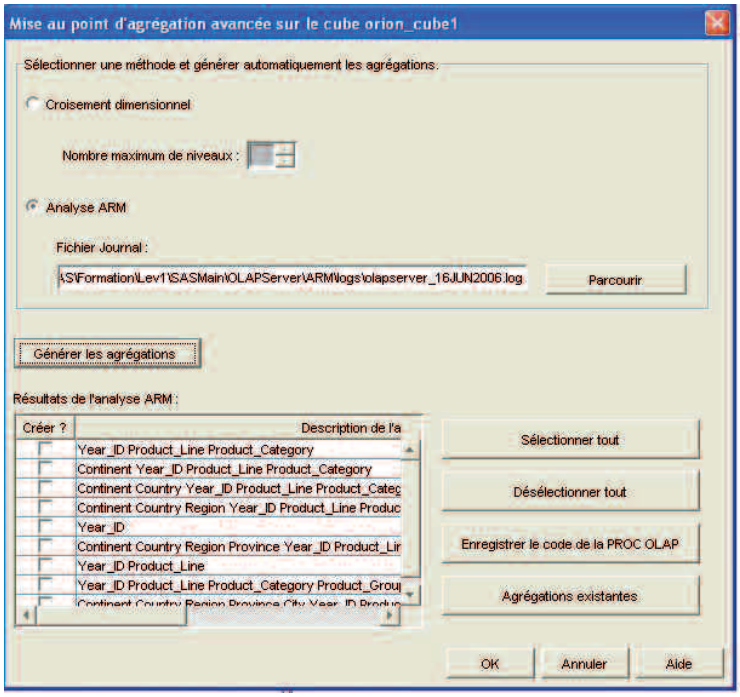
Attention, vous ne pourrez définir un filename pointant sur la log qu'une fois que le serveur Olap aura été arrêté.

5. Modifier un cube grâce à la log ARM
Dans Olap Cube Studio, vous avez la possibilité de mettre à jour un cube suite aux informations récupérées dans la log ARM.

RETOUR SOMMAIRE



Après la sélection du cube, sélectionnez « Analyse ARM » et spécifiez le répertoire de stockage du fichier ARM précédemment généré ainsi que son nom.
En utilisant le bouton « Générer les Agrégations », vous obtiendrez la liste des croisements utilisés lors de la manipulation du cube Olap (voir écran ci-dessous).



Vous avez la possibilité de choisir ou non d'intégrer ses croisements dans votre cube.
Une fois la sélection effectuée, les métadonnées et le cube vont être mis à jour.
Pour gagner en performance, il est important de stocker les agrégations les plus souvent utilisées.

Cet article vous a présenté un rapide aperçu de l'utilisation de l'ARM et plus particulièrement dans le contexte OLAP.
Pour de plus amples informations, je vous invite à consulter l'aide de SAS :
SAS Products > Base SAS > SAS Language Concepts > SAS System Concepts > Monitoring Performance Using Application Response Measurement (ARM)

Et pour des informations plus précises sur les macros ARM :
SAS Products > Base SAS > SAS Language Dictionary > Dictionary of Language Elements > SAS ARM Macros

Cécile Guillouzo

Directeurs de la publication :
Philippe Hoffmann
Pierre Lorenzetti

Comité de rédaction :
Karine Chrillesen
Aurélie Gilbert
Blandine Colas
Sylvain Bertin
Cécile Guillouzo

Comité de relecture :
Fabrice Giraud
Mouloud Dey
Philippe Hoffmann

Réalisation :
aressyassociés
+33 (0)1 41 38 85 53 - 07/06 - 2771

RETOUR SOMMAIRE