Techniques de chargement de sous-fichier en Adélia

- Les articles -

Date de mise en ligne : samedi 29 avril 2006

Description :

Cet article présente différentes techniques de chargement de sous-fichier en Adélia.

Environnement iSeries

Cet article présente différentes techniques de chargement de sous-fichier en Adélia, présentées au travers d'un programme interactif de type fenêtre.

L'objectif est ici de vous présenter quelques unes des techniques de chargement de sous-fichier, parmi les plus utilisées dans les entreprises utilisant l'AGL Adélia.

Pour vous présenter ces techniques, j'ai choisi d'écrire 3 programmes Adélia, qui font la même chose mais de façon différente. Il s'agit en fait d'une fenêtre de recherche sur un fichier client. Cette fenêtre reçoit en entrée un code société, et affiche les clients appartenant à cette société. Petite difficulté supplémentaire, je souhaite n'afficher que les clients "valides", c'est à dire dont le code statut est à blanc (cf. structure du fichier présentée ci-après). La fenêtre permet à l'utilisateur de faire des recherches sur le code client et sur le nom de client, en saisissant l'un ou l'autre dans les zones prévues à cet effet, juste au dessus des colonnes du sous-fichier. Une fois que l'utilisateur a identifié le client qui l'intéresse, il peut saisir l'option "1=Choisir" devant ce client pour rapatrier son code sur le programme appelant. Il peut également utiliser l'option "5=Afficher" qui appelle un programme de consultation de la fiche client. Ce programme de consultation ne sera pas décrit dans le présent article.

Concrètement, la fenêtre de recherche se présentera de la façon suivante :

```
/////////////// Sélection d'un client /////////////
 //                                                //
 //  Indiquez vos choix, puis appuyez sur ENTREE.  //
 //    1=Choisir   5=Afficher                      //
 //  Opt Client  Raison sociale                    //
 //      _______ ______________________________    //
 //   _  6666666 000000000000000000000000000000    //
 //   _  6666666 000000000000000000000000000000    //
 //   _  6666666 000000000000000000000000000000    //
 //   _  6666666 000000000000000000000000000000    //
 //   _  6666666 000000000000000000000000000000    //
 //   _  6666666 000000000000000000000000000000    //
 //   _  6666666 000000000000000000000000000000    //
 //   _  6666666 000000000000000000000000000000    //
 //   _  6666666 000000000000000000000000000000    //
 //                                                //
 //// F12=Annuler //////////////////////////////////
```

Avant de poursuivre, je vous donne ci-dessous la structure du fichier CLIENT (bien évidemment c'est un fichier client très "allégé" dans lequel je n'ai conservé que les zones utiles pour les programmes de cet article) :

- Fichier logique CLIENTL1 : (tri par numéro de client)

```
Clé Zone   Mot directeur   Désignation            Lg.D
 1 C1CSOC    C1_COD_SOC      Code société            3
 2 C1NCLI    C1_NUM_CLI      Numéro de client        7 0
 C1RSCL    C1_RS_CLI       Raison sociale client   30
 C1CSTA    C1_COD_STATUT   Code Statut            1
```

-

Fichier logique CLIENTL2 : (tri par nom de client)

```
Clé Zone    Mot directeur    Désignation            Lg.D
 1 C1CSOC    C1_COD_SOC       Code société             3
 3 C1NCLI    C1_NUM_CLI       Numéro de client         7 0
 2 C1RSCL    C1_RS_CLI        Raison sociale client   30
 C1CSTA    C1_COD_STATUT    Code Statut              1
```

Vous trouverez dans le fichier zippé joint à cet article le source de 3 programmes Adélia : ZFC01F, ZFC02F et ZFC03F. Chacun de ces sources est fourni dans 2 formats différents :

- le format ODT d'Open Office 2.0,
- le format PDF pour ceux d'entre vous qui ont la malchance de ne pas avoir installé cette excellente suite bureautique qu'est Open Office.

Le programme **ZFC01F** a pour particularité d'utiliser le mode de chargement le plus standard d'Adélia, appelé généralement "mode de chargement automatique". C'est certainement le plus facile à utiliser, spécialement pour les développeurs en phase d'apprentissage de l'AGL. C'est probablement aussi la méthode de codage la plus rapide, si l'on suit scrupuleusement la méthodologie préconisée par les formateurs de la société Hardis. Il faut noter que, dans ce mode de chargement, chaque type de chargement (par code, par nom, etc...) nécessite de développer une transaction dédiée. Comme notre fenêtre offre 2 types de chargement (par code client, et par nom de client), notre programme contiendra 2 transactions identiques. Le code de chargement du sous-fichier est extrêmement compact (cf. mots clés GESTION_SFL et FIN_GESTION_SFL dans le pavé INITIALISATION). Les "move" des zones fichiers vers les zones écran sont gérés par la cinématique et l'ordre PRESENTER. La pagination avant est gérée par l'ordre Adélia PAGINER_AVANT. Difficile de faire plus simple.

Le programme **ZFC02F** a pour particularité d'utiliser le mode de chargement dit "manuel", qui nécessite davantage de codage. Pour corser un peu l'exercice, j'ai introduit une petite différence par rapport au programme ZFC01F, en ajoutant dans le ZFC02F la possibilité d'effectuer une recherche sur une partie d'un nom. Pour cela, l'utilisateur saisit un "%" au début et à la fin du nom (exemple : %MARTIN%), et le programme va effectuer une recherche dans le fichier client au moyen d'une requête SQL de type "LIKE" (CONTIENT). Vous noterez que dans ce programme, le chargement du sous-fichier nécessite davantage de codage (cf. procédures SFLCOD et SFLNOM) et une technique de codage particulière pour la pagination avant (cf. pavé TRANSACTION). En contrepartie, tous les types de chargement sont gérés au travers d'une seule transaction.

Le programme **ZFC03F** fonctionne de manière identique au programme ZFC02F, mais il a pour particularité d'utiliser un mode de chargement que j'appelle "Full SQL". Ce mode de chargement ne nécessite lui aussi qu'une seule transaction, et vous constaterez qu'il nécessite beaucoup moins de codage que les 2 autres programmes car tous les chargements (par code, par nom entier ou partiel) sont gérés par une requête SQL, constituée "en live", par concaténation de différents critères de sélection.

N.B. : Il aurait tout à fait été possible d'ajouter la recherche par SQL de type "CONTIENT" au programme ZFC01F, pour cela il aurait suffi de créer dans ce programme une 3ème transaction identique aux précédentes, mais avec un mode de chargement de type SQL, et en ajoutant au source du programme le code nécessaire à la gestion de cette 3ème transaction. Je n'ai pas voulu le faire pour ne pas alourdir le code du programme ZFC01F, vous pouvez vous y essayer si vous le voulez, l'exercice est intéressant.

Conclusion

Entre les 3 solutions proposées, ma préférence va au programme ZFC03F pour la compacité du code, et l'évolutivité (l'ajout de nouveaux positionnements nécessite très peu de modifications).

En seconde position, je vais peut être vous surprendre, mais je préfère la technique employée sur le programme ZFC01F, car elle nécessite elle aussi assez peu de codage. Elle a en revanche le défaut d'obliger le développeur à créer autant de transactions que de types de chargement souhaité. Ceci peut être préjudiciable à la maintenance ultérieure, car si une modification doit être faite sur la présentation d'une transaction, elle doit être répercutée à l'identique sur les autres transactions. De plus, pour chaque transaction, on est obligé de dupliquer dans le source du programme les pavés TRANSACTION, INITIALISATION, VERIFICATION, et éventuellement VALIDATION (si besoin). Ceci a pour effet d'alourdir la maintenance du programme. Sur un petit programme tel que celui présenté ici cela n'est pas trop préjudiciable, mais sur des programmes plus complexes, cela peut vite devenir problématique. Je réserverais donc cette technique à des programmes de consultation simples, dont je sais par expérience qu'ils sont peu susceptibles d'évoluer.

Il faut noter que la mise au point de requêtes SQL - telle qu'elles sont pratiquées dans les programmes ZFC02F et ZFC03F - doit faire l'objet d'une attention particulière de la part du développeur, qui doit s'assurer que les requêtes créées s'appuient sur des chemins d'accès existants, sous peine de rencontrer de gros problèmes de performances par la suite.

En complément de cet article, je vous recommandé la lecture d'un autre article que j'avais écrit il y a pas mal de temps, et qui s'intitule "Forcer le positionnement de sous-fichier sous Adélia.". Cet article présente une petite variante au mode de positionnement utilisé dans le chargement de sous-fichier, présenté dans le programme ZFC01F.

Il faut noter que je n'ai pas traité ici le problème de la pagination arrière, d'autant que celle-ci ne fonctionne pas dans les programmes Adélia interactifs de type fenêtre. Si vous souhaitez néanmoins des informations à ce sujet, je vous renvoie à un autre article que j'avais écrit sur ce sujet, intitulé "Gérer la pagination arrière dans un sous-fichier sous Adélia.".

Je vous recommande également la lecture de 2 articles de Serge Gomes, qui vous propose un "Prototype Chargement de sous-fichier avec SQL", ainsi qu'un "Générateur freeware de code source pour AS400", cette fois-ci dans un contexte de développement en RPG.