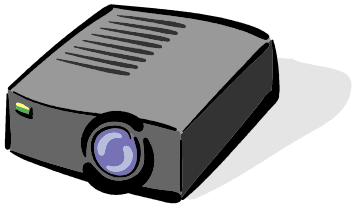
Modernisation et développement d'applications IBM i
Stratégies, technologies et outils

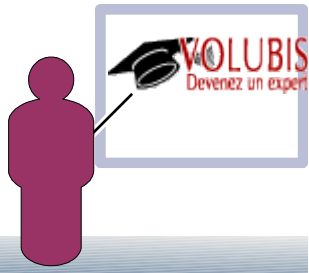
16 et 17 mai 2011 – IBM Forum de Bois-Colombes



Volubis.fr

Conseil et formation sur OS/400, I5/OS puis IBM ***i***
depuis 1994 !

Christian Massé - cmasse@volubis.fr



Nouveautés RPG4

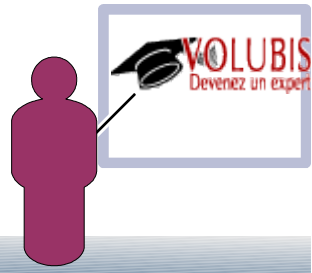
- Nouveautés liées au langage RPG en V6 et V7

- Support complet du multithreading avec THREAD(*CONCURRENT) en specif H , Chaque thread possède une copie des variables déclarées STATIC

- support plus souple d'UNICODE

  une conversion implicite est réalisée lors des affectations (EVAL/EVALR)

  et des tests (IF, DOW, …). Elle était déjà réalisée par MOVE, MOVEL.

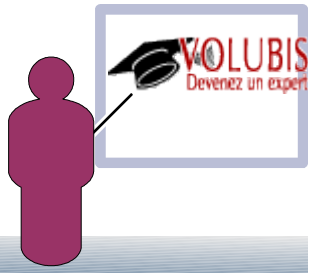


Nouveautés RPG4

- la taille maxi des constantes passe à 16380 (la moitié pour UNICODE)
- la taille maxi des variables caractères passe de 65535 à 16 Mo !!
  (pratique pour la manipulation du XML)
- le nombre d'occurrences d'une DS ou d'un tableau n'est plus limité

  qu'à la taille globale maxi de 16.773.104 octets (16 Mo)
- le mot-clé LEN peut être utilisé à la place des colonnes 33/39 (lg)

  sur la définition d'une zone ou d'une DS (spécif D)



Nouveautés RPG4

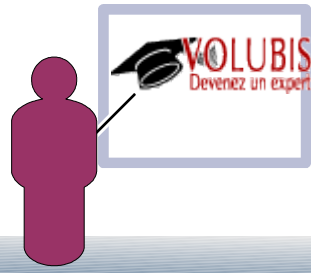
- la nouvelle taille des variables permettra de mieux gérer XML-INTO
- *Rappel*, il s'agit de « parser » un flux XML (variable ou fichier IFS) et de placer les données dans une DS ayant des zones de même nom.

- Nous bénéficions des avancées suivantes, des versions précédentes :
  - Une DS peut être qualifiée et donc deux DS peuvent contenir des zones de même nom.
  - Une DS qualifiée peut être à dimension

```
eval clients(i).nocli = 45 ;
```

  - Une DS admet comme sous zone une DS avec LIKEDS

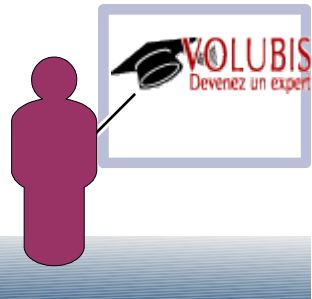
```
eval commande.article.qte = 127 ;
eval commande(x).article(y).qte = 12 ;
```



Nouveautés RPG4

- XML-INTO, exemple

```
 *
 *  test avec attributs et éléments, doc=string-> le XML est dans une variable
 *
 *

Ddata             S             1024    inz('+
D                                        <cours nom="XML"  module="PGM">
D                                          <sujet>RPG</sujet>
D                                          <texte>manipuler du XML en RPG
D                                          </texte>
D                                        </cours>
D                                       ')
DCOURS            DS
D NOM                               10
D MODULE                            10
D SUJET                             10
D TEXTE                             50

 /free

  xml-into cours   %xml(data : 'doc=string');

  *inlr = *on;

 /end-free
```



Nouveautés RPG4

- Quelques améliorations sur les options de XML-INTO *(PTF SI34938 et SI42426)*

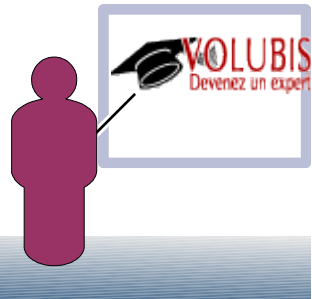
1/ datasubf permet de régler le problème suivant:

```
     Ddata             S            1024    inz('+
     D                                       <cours nom="XML"  module="PGM">
     D                                         manipuler du XML en RPG
     D                                       </cours>
     D                                      ')
```

La balise cours contient deux attributs nom et module, mais aussi une donnée directement encadrée par la balise (ici le texte descriptif)

l'option datasubf=text ("text" est un exemple), permet de recevoir les données dans une DS de ce type :

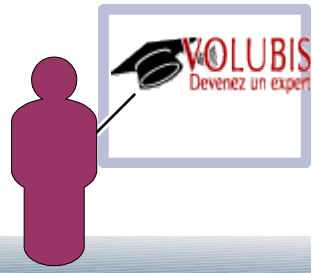
```
     DCOURS            DS
     D NOM                              10
     D MODULE                           10
     D TEXT                             50
```



Nouveautés RPG4

- Quelques améliorations sur le XML-INTO (SI34938 et SI42426)

```
2/ countprefix permet de recevoir le nombre d'éléments extrait dans un

     compteur.

     Ddata             S           1024    inz('+
     C                                      <les cours>
     D                                      <cours nom="XML"  module="PGM">
     D                                      </cours>
     D                                      <cours nom="XML"  module="RPG">
     D                                      </cours>
     D                                      <cours nom="XMl" module ="GAP">
     D                                      </cours></lescours>')

  countprefix=nbr, permet d'avoir une DS de ce type  associé à xml-into

     Dlescours
     Dnbrcours                       10I  0
     Dcours                                 likeds(cours_modele) DIM(5)


la variable nbrcours contenant ensuite la valeur 3.
```



Nouveautés RPG4

- Quelques améliorations sur le XML-INTO (SI34938 et SI42426)

3/ ns , gestion des namespaces (espaces de nommage)

parfois la structure XMl utilise les namespaces, pour "qualifer" les noms :

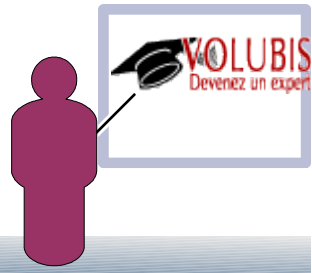
```
Ddata             S           1024    inz('+
D                                      <les cours  xmlns:vNS=
D                                        "http://www.volubis.fr/NS">
D                                      <vNS:cours nom="XML" module="PGM">
D                                      </cours>
D                                      <vNS:cours nom="XML"  module="RPG">
D                                      </cours>
D                                      <vNS:cours nom="XMl" module ="GAP">
D                                      </cours></lescours>')
```

ns=remove enlève les espaces de nommage pour rechercher les sous-zones

(la data structure s'appelle cours)

ns=merge ajoute l'espace de nommage en remplaçant ":" par "_"

(la data structure s'appelle vNS_cours)



Nouveautés RPG4

- Quelques améliorations sur le XML-INTO (SI34938 et SI42426)

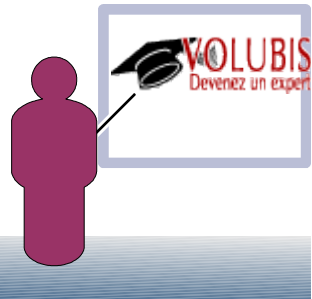
4/ ns_prefix, récupération du namespace

nsprefix=namespc_

si on trouve une zone namespc_xxx, où xxx est le nom de la zone
cette dernière recevra le namespace, soit "vNS" dans nos exemples.

exemple

```
DCOURS             DS

D NOM                              10
D namespc_NOM                      32
D MODULE                           10
D namespc_MODULE                   32
```



Nouveautés RPG4

- Quelques améliorations sur le XML-INTO (SI34938 et SI42426)

  `5/ case=convert`

  gère les balises XML portant des noms, non valides en RPG (code-postal)

  les noms sont convertit en majuscules (comme case=any),les caractères

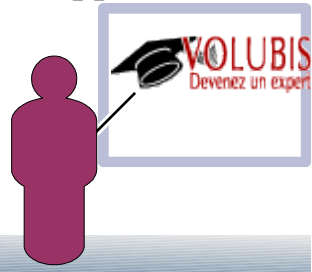
  invalides remplacés par '_', sauf le première caractère qui est ignoré

  s'il est invalide. (code-postal devient CODE_POSTAL)

- Pour toute utilisation plus complexe (nombre d'occurrences inconnu), voyez plutôt la notion de %handler avec XML-INTO :

```
xml-into        %HANDLER(trt10 : flag)
                 %XML('/af4dir/courshtm/xml/cours.xml'  :
                 'doc=file case=upper path=AF400/COURS')
```

`XML-INTO` appel la procédure trt10 qui reçoit une DS à dimension (ici, DIM(10)), le parser appelle alors la procédure autant de fois que nécessaire .



Nouveautés RPG4

- Autres nouveautés V6

EXTDESC() permettant d'indiquer le nom du fichier à la compilation

-> cela permet de compiler avec un fichier qui n'est pas dans *LIBL

EXTDESC('BDVIN1/VINS')

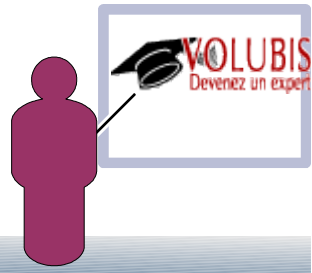
le paramètre EXTFILE(), qui lui, indiquait un nom de fichier à utiliser à l'exécution, admet maintenant EXTFILE(*EXTDESC)

-les définitions de fichier (F) et de DS (D) admettent un nouveau mot-clé **TEMPLATE** indiquant que cette déclaration n'est qu'un modèle.

-Les ordres d'entrée/sortie READ,CHAIN,UPDATE,etc, admettaient un nom de DS résultat dans le cadre d'un fichier décrit en interne.

la DS résultat est maintenant admise aussi pour un fichier externe

et EXFMT possède aussi cette possibilité.



Nouveautés RPG4

- Autres nouveautés V6

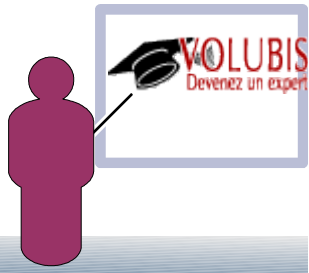
-un nom de format peut être qualifié par QUALIFIED en spécif F

le nom de format doit être manipulé sous la forme "fichier.format"

il n'a plus à être unique MAIS il n'y a plus de spécif I et O pour ce fichier

==> les I/O doivent utiliser une DS :

```
Fvins      IF   E             DISK    QUALIFIED
Dinds             DS                   LIKEREC(vins.vinsf1)


 /free

  read vins.vinsf1 INDS;
  dsply  INDS.VIN_NOM;
  return;

 /end-free
```



Nouveautés RPG4

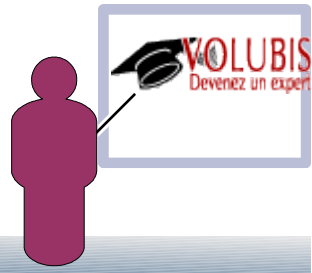
- Autres nouveautés V6

le mot-clé MAIN en spécif H permet d'indiquer une sous procédure en tant que procédure principale. Cette dernière ne peut être lancée que par CALL.

le source ne contient donc plus que des procédures, la procédure déclarée MAIN doit contenir EXTPGM('le-nom-du-pgm') sur le prototype

il n'y a plus de cycle GAP , il faut donc fermer explicitement par CLOSE les fichiers (le compilateur signale une erreur RNF7534 de gravité 10)

```
h MAIN(ENTREE)
D ENTREE          PR                  EXTPGM('TESTMAIN')
D                                 5A

P ENTREE          B
D                 PI
D parm1                           5A

 /free
  dsply parm1;
 /end-free

P ENTREE          E
```



Nouveautés RPG4

- Autres nouveautés V6

les spécif F sont admises maintenant locales (à l'intérieur des procédures)

entre la spécif P et les D, à condition d'être qualifiées.

->les entrées/sorties doivent être réalisées à l'aide de DS.

les fichiers sont fermés automatiquement à la fin de la procédure, sauf à

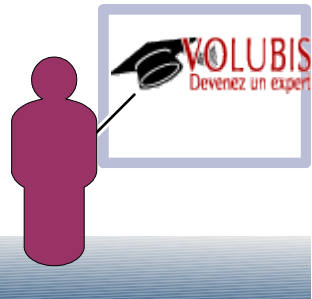
utiliser STATIC, qui garde les fichier ouverts et les variables chargées.

- **LIKEFILE** , définition d'un fichier relativement à un autre.

on récupère tous les attributs de la déclaration du fichier d'origine (type d'ouverture, utilisation de la clé, etc...)

Attention cela implique encore QUALIFIED (implicite), donc DS résultat.

```
Fvins      IF   E           DISK
Fproducteur                          LIKEFILE(vins)
```



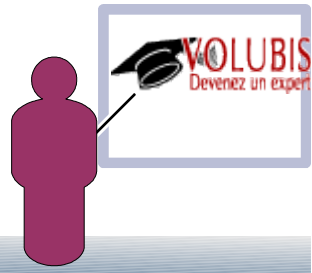
Nouveautés RPG4

- Autres nouveautés V6

Enfin, et c'était probablement le but de tout ce que nous venons de voir, une ouverture de fichier peut être transmise en tant que paramètre, permettant la lecture dans une procédure et la mise à jour dans une autre.

En indiquant tout simplement LIKEFILE sur la définition du paramètre dans le prototype.

```
H ALWNULL(*USRCTL)
FVINS      UF   E           K DISK    TEMPLATE QUALIFIED
FVINLU                                LIKEFILE(VINS) EXTFILE('BDVIN1/VINS')

  … / …

/free

   read vinlu inds;
   dow not %eof;

    if inds.vin_nom = ' ';
     miseajour(vinlu:inds) ;
    endif;

    read vinlu inds;
   enddo;

  *inlr = *on;

/end-free
```



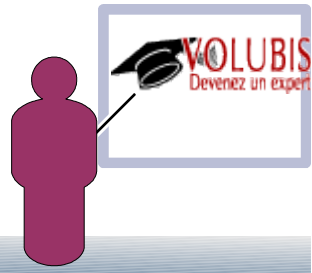
Nouveautés RPG4

- Autres nouveautés V6

```
H NOMAIN
FVINS      UF   E           K DISK    TEMPLATE QUALIFIED
Dds_modele       DS                    likerec(vins.vinsf1) TEMPLATE

  … / …

pmiseajour       B                     EXPORT
D                PI
D vin_in                               LIKEFILE(vins)
D OUTDS                                LIKEDS(DS_MODELE)

 /free

   eval outds.vin_nom = '(non précisé)' ;
   update vin_in.vinsf1 outds;

 /end-free

Pmiseajour       E
```

cela va permettre de programmer de manière **très** modulaire.

est-ce accessible au plus grand nombre ?



Nouveautés RPG4

- Nouveautés syntaxiques de la V7

elle sont assez peu nombreuses

- plus de possibilités sur les DS à dimensions

-> possibilité de les trier sur une colonne particulière

```
SORTA clients(*).depart;
```

-> possibilité de recherche par lookup

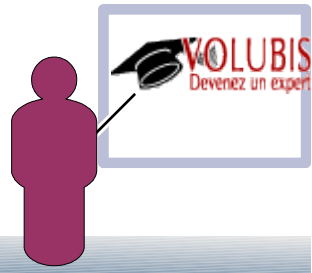
```
pos = %lookup(44 : clients(*).depart );
```

- amélioration du tri par SORTA

en plus du fait de savoir trier des DS à dimension,l'ordre SORTA admet maintenant un critère de tri (A ou D) en option

```
SORTA(A) tbtva;
```

le tableau doit avoir été déclaré SANS critère de tri (ASCEND ou DESCEND)



Nouveautés RPG4

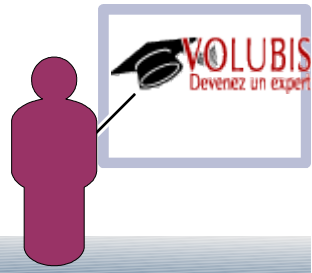
- Nouveautés syntaxique de la V7

  - nouvelle fonction **%SCANRPL** qui cherche ET remplace une chaîne

```
chaine1 = 'recherche de la chaîne "RPG" en RPG';
chaine1 = %scanRPL('RPG' : 'ILE' : chaine1);
// chaine1 contient 'recherche de la chaîne "ILE" en ILE'
```

  - la fonction %LEN admet en 2eme argument *MAX permet de retrouver pour une variable à taille variable, non pas la taille actuelle, mais la taille maxi.

```
Dchaine2          s            255    varying
Di                s              5I 0
 /free

  chaine2 = 'Bonjour';
  i = %len(chaine2);      // i = 7
  i = %size(chaine2);     // i = 257 (à cause des deux octets binaires)
  i = %len(chaine2:*MAX); // i = 255

 /end-free
```



Nouveautés RPG4

- Nouveautés syntaxique de la V7

  - la fonction **%PARMNUM** retourne la position d'un paramètre

  - les DS externes utilisent les noms d'ALIAS (au sens SDD)
    plutôt que les noms courts (sur 10) avec le mot-clé RPG **ALIAS.**

  - les valeurs retour des fonctions peuvent être maintenant
    transmises en tant que paramètre (caché)

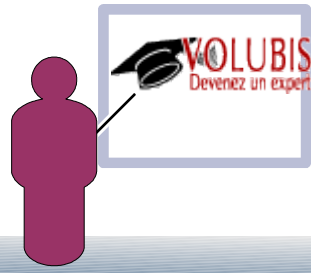
    En effet les valeurs retour des fonctions étaient transmises en tant
    que valeur sur la pile d'appel.

    lorsque ces dernières étaient de grande taille, cela avait un impact
    sur les performances, particulièrement lors d'appels successifs.

    en utilisant RTNPARM la valeur retour est en fait un paramètre
    supplémentaire transmis par adresse, donc plus rapide d'accès.

    ce paramètre doit être indiqué dans le prototype ET l'interface de
    procédure.

  - les prototype ne doivent plus être indiqués impérativement dans le source
    qui implémente une fonction, mais uniquement dans ceux qui l'utilisent

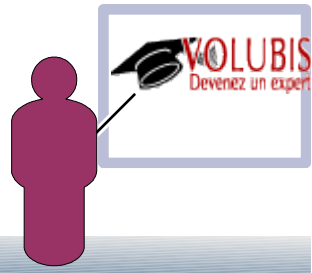


Nouveautés RPG4

Rational Open Access , RPG Edition

Rational Open Access, RPG edition (5733OAR) est un produit apparu en 7.1 et disponible aussi pour la V6R1(avec PTF).

- Il permet de passer la main à un "driver" externe lors des ordres d'entrée sortie RPG, plutôt que d'appeler les routines historiques d'IBM. Par exemple
  - Vous avez actuellement un programme RPG écrivant dans un fichier physique.
  - Vous écrivez ou achetez un Handler qui reçoit les données et qui les écrit au format CSV ou XML dans l'IFS à la place de l'écriture dans le fichier physique.
  - Vous ajoutez le mot-clé HANDLER sur la déclaration du fichier en sortie de votre programme initial et le tour est joué, ce dernier écrit dans l'IFS.
- Attention
  - Le fichier auquel vous avez ajouté HANLDER doit être présent à la compilation ET à l'exécution (malgré que vous ne vous en serviez pas vraiment, il fournit le format)
  - le produit 5733OAR (facturable) doit être présent à la compilation ET à l'exécution.



Nouveautés RPG4

- Rational Open Access , RPG Edition
- Utilisation du Handler

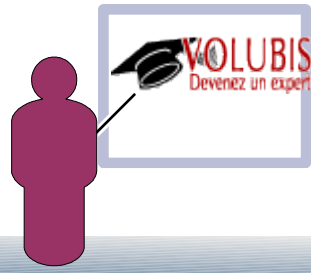
```
Fsortie  o    e            Disk    Handler('CVS_HDLR' : ifs_info1)
```

la transmission d'information est facultative, ici elle contiendra le nom du fichier à générer dans l'IFS.

- - Ecriture du Handler

Le Handler est appelé à chaque opération (open, close, read, etc...) et reçoit une DS d'information structurée (QrnOpenAccess_T, voir QRNOPENACC) décrivant l'Entrée/sortie et permettant aussi un retour d'informations auprès des couches système.

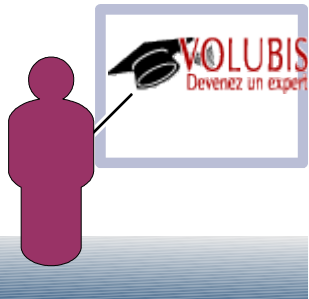
```
// Exemple
// copie des définitions standard fournies par IBM
/copy QOAR/QRPGLESRC,QRNOPENACC
// paramètre reçu par le pgm
// de type QrnOpenAccess_T

D CVS_HDLR        PI
D   info                                  likeds(QrnOpenAccess_T)
```



Nouveautés RPG4

- Rational Open Access , RPG Edition

```
zones de QrnOpenAccess_T remarquables :

    structLen       : lg totale de cette structure
    parameterFormat : toujours 'ROAC0100'
    userArea        : pointeur vers l'espace mémoire fourni en 2ème paramètre
                      du mot-clé Handler par le pgm (libre)
    stateInfo       : pointeur vers un espace mémoire permettant au Handler de
                      mémoriser quelque chose entre deux appels (libre)
    RecordName      : nom du format du fichier
    rpgOperation    : Opération en cours, voir les constantes QrnOperation_xxx
                      dans QRNOPENACC
    rpgStatus       : s'il est appelé pour une opération qu'il ne sait pas
                      gérer, doit signaler une erreur (rpgStatus <> 0).
    Eof             : à mettre à *ON pour signaler une fin de fichier
    Found           : à mettre à *ON pour signaler un accès direct réussi
    Equal           : à mettre à *ON pour signaler une opération de
                      positionnement par égalité réussie
    inputBuffer     (pointeur vers un buffer mémoire si useNamesValues=*OFF)
    NamesValue      (pointeur vers un espace mémoire structuré de type
                      QrnNamesValues_T, si useNamesValues=*ON)


lors de l'OPEN le Handler doit signaler comment il veut recevoir les données
(QrnOpenAccess_T.useNamesValues)

    *OFF ('0') les données sont fournies dans un buffer identique à
               celui reçu par les routines systèmes (suite d'octets)
    *ON ('1') les données sont fournies dans une structure contenant
              la description de la zone et les données en clair
              (en caractère, même pour le numérique)
```



Nouveautés RPG4

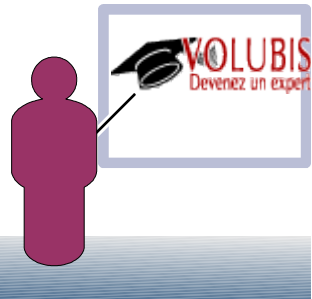
- Rational Open Access , RPG Edition

QrnNamesvalues_T (si useNamesvalues est à *ON) :

nombre de zones (binaire sur 4 octets)

tableau de n occurrences (suivant zone précédente) de type QrnNameValue_T

```
//Exemple
// déclaration de la structure basée sur QrnNamesvalues_T

D PnvInput        S                   *
D nvInput         ds                    likeds(QrnNamesValues_T)
D                                       based(pNvInput)
 // contient
 // --------
 //D   num                          10I 0
 //D   field                              LIKEDS(QrnNameValue_T)
 //D                                      DIM(32767)

/free
     // utilisation du pointeur lu dans info (QrnOpenAccess_T)
     if info.rpgOperation = QrnOperation_WRITE;
        pNvInput = info.namesValues;
```



Nouveautés RPG4

- Rational Open Access , RPG Edition

QrnNamevalue_T (description d'une zone) , les lg sont données en octets :

Nom, CHAR(10)

Type de données, Integer(1), voir les constante dans QRNOPENACC

Longueur définie (numérique uniquement) , Integer(1)

Décimales, Integer(1)

Format (date/heure), Integer(1)

Séparateur (date/heure), CHAR(1)

Input ? (*ON/*OFF), CHAR(1)

Output ? (*ON/*OFF), CHAR(1)

NULL possible ? (*ON/*OFF), CHAR(1)

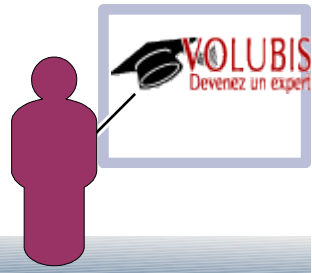
Contient NULL ? (*ON/*OFF), CHAR(1)

Lg de la valeur (peut être renseignée par RPG), Integer(4)

Lg maxi de la valeur (fixée par le système), Integer(4)

CCSID de la valeur, Integer(4)

Valeur, pointeur vers cette dernière, qui elle est en clair
(résultat de %CHAR si numérique, de %DATE si c'est une date...)



Nouveautés RPG4

- Rational Open Access , RPG Edition

```
//Exemple, suite

       // boucle autours du nbr de variables
       // pour récupérer les valeurs et les traiter
   D pvaleur         s                 *
   D valeur          s           32470a   Based(pvaleur)

    /free

     For i = 1 to nvInput.num;
        pvaleur = nvInput.field(i).value; // accès à la donnée

        If ( nvInput.field(i).dataType = QrnDatatype_Alpha );
          …
          manipulation de "valeur"
          …
        EndIf;
     EndFor;

.../...
```

- Pour des exemples complets, voyez notre réunion technique de septembre 2010

  http://www.volubis.fr/Pausecaf/PAUSECAF56.htm