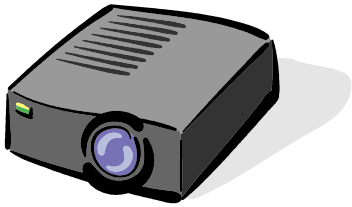
Modernisation et développement d'applications IBM i

Technologies, outils et nouveautés 2012/2013

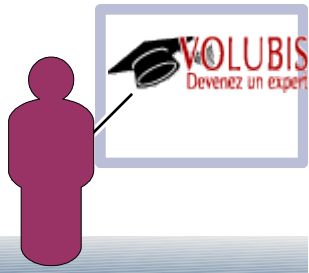
8 et 9 Avril 2013 – IBM Forum de Bois-Colombes



Volubis.fr

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

Christian Massé - cmasse@volubis.fr



Nouveautés concernant le CL

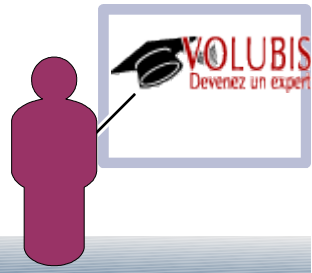
Avec la version 5.3, IBM met un « coup de jeune » au langage de contrôle

- Nouveaux types de variables (integer)
- Boucles (dowhile, dountil, dofor)
- Support de fichiers multiples (jusqu'à 5 par programme)

Les versions 5,4 et 6.1 continuent ces évolutions

- Sous programmes, par exemple.

Enfin, une PTF en version 7 apporte de nouvelles fonctions intégrées



Nouveautés CL : V5R30

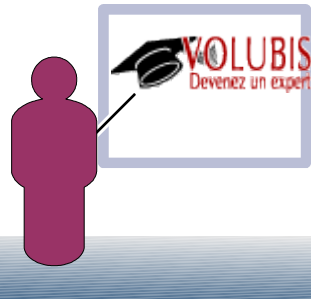
variables binaires

```
DCL        VAR(&CPT) TYPE(*INT)
```

il existe aussi le format *UINT (binaire non signé)

la longueur est de 2 octets ou 4 octets

la transformation de binaire en décimal ou caractère est possible avec CHGVAR



Nouveautés CL : V5R30

Commandes de structuration

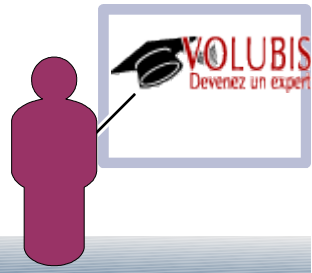
les seules commandes disponibles avant la V5R30, (hors GOTO) étaient

IF et DO/ENDDO (ces dernières utilisée avec IF et MONMSG en général)

Nous avons maintenant

DOWHILE COND(même test que sur un IF)
...
ENDDO

c'est un vrai dowhile, le test est réalisé avant (on peut ne jamais entrer)
et il s'agit de la condition pour faire



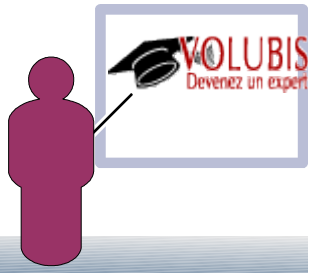
Nouveautés CL : V5R30

Commandes de structuration

DOUNTIL COND(même test que sur un IF)
...
ENDDO

pour dountil, le test est réalisé sur le ENDDO (on fait au moins une fois)

et la condition donnée est celle pour SORTIR.



Nouveautés CL : V5R30

Commandes de structuration

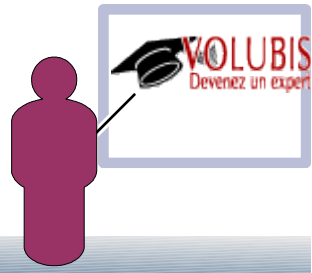
DOFOR VAR(&cpt) FROM(1) TO(22) BY(3)
...
ENDDO

pour ces trois boucles on peut forcer :

- une sortie anticipée par **LEAVE**

    sans paramètre, on sort de la boucle en cours (la dernière)
    on peut mettre un LABEL devant le DOxxx et indiquer le label lors du LEAVE

- un saut d'un tour de boucle par **ITERATE**



Nouveautés CL : V5R30

Commandes de structuration

et enfin, l'équivalent du CASE SQL ou du SELECT RPG :

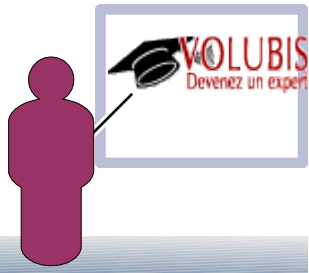
SELECT

WHEN COND( ) THEN( ) /* seul le premier test vrai est traité */

WHEN COND( ) THEN()

OTHERWISE CMD( )

ENDSELECT



Nouveautés CL : V5R30

support de fichiers multiples

Avant la V5R30, nous ne pouvions déclarer qu'un seul fichier par pgm
ce qui fait qu'il n'était pas utile de préciser le nom lors des lectures

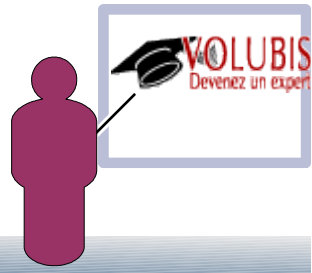
La v5r30 apporte un nouveau paramètre OPNID() sur les commandes DCLF
RCVF, SNDRCVF, SNDF permettant de manipuler plusieurs fichiers.

```
DCLF       FILE(ECRAN1) OPNID(ECRAN)
```

l'OPNID est facultatif, mais un SEUL fichier peut ne pas en avoir,
et nous sommes limités à 5 fichiers en tout dans un même programme.

ATTENTION : les variables seront préfixées par l'OPNID suivi de _

JOB -> &ECRAN_JOB , Indicateur 3 -> nommé &ECRAN_IN03



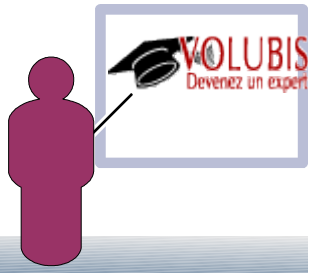
Nouveautés CL : V5R30

support de fichiers multiples

les commandes de manipulation de fichier (SNDF, RCVF et SNDRCVF) doivent utiliser la paramètre OPNID s'il a été utilisé lors de la déclaration.

exemple :

```
          PGM
          DCLF      FILE(QADSPOBJ) OPNID(OBJD)
          DCLF      FILE(ECRAN1) OPNID(ECRAN)


BCL1:     DOUNTIL   COND(&ECRAN_IN03)
            RCVF      OPNID(OBJD)
              MONMSG     MSGID(CPF0864) EXEC(LEAVE CMDLBL(BCL1))
            CHGVAR &ECRAN_ODOBNM &OBJD_ODOBNM
            CHGVAR &ECRAN_ODOBTP &OBJD_ODOBTP
            CHGVAR &ECRAN_ODOBTX &OBJD_ODOBTX


            SNDRCVF OPNID(ECRAN)

          ENDDO

          ENDPGM
```



Nouveautés CL : V5R30

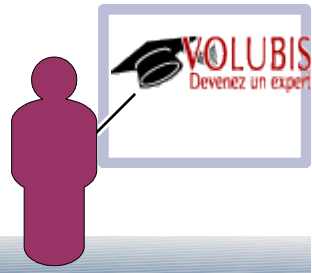
Enfin, les commandes CALL et CALLPRC (ILE) acceptent maintenant **255** paramètres (avant nous étions limités à 40) et les paramètres peuvent être transmis par référence (comme avant) ou **par valeur** (nouveauté), sur la commande CALLPRC.

si vous lancez un programme RPG4, les paramètres de la procédure doivent être déclarés avec VALUE (comme param1, ci-dessous)

```
DmaFonction      PR
D param1                  3  0 VALUE
D param2                 10
```

*en C, sans le préfixe * (indiquant un pointeur)*

-> CALLPRC PRC('maFonction') parm((P1 *BYREF) (P2))



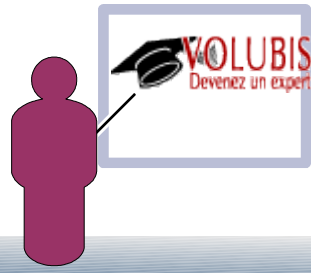
Nouveautés CL : V5R40

Après les type *INT et *UINT en V5R30, le type *PTR (pointeur) est maintenant proposé par la V5R40

DCL VAR(&pointeur) TYPE(*PTR)

ce qui permet ensuite la déclaration d'une variable basée sur ce pointeur

DCL VAR(&Data) TYPE( ) STG(*BASED) BASPTR(&pointeur)



Nouveautés CL : V5R40

un pointeur peut être renseigné :

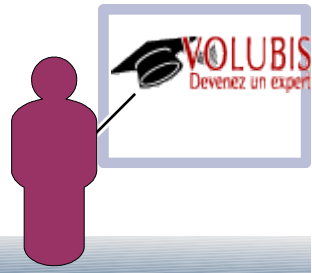
1/ par copie d'un pointeur

2/ par récupération de l'adresse d'une variable avec %ADDRESS() ou %ADDR

```
CHGVAR &pointeur %ADDRESS(&autreData)
```

3/ en incrémentation, décrémentation avec la fonction %OFFSET() ou %OFS

```
CHGVAR %OFFSET(&pointeur) VALUE(%OFFSET(&pointeur) + 1)
```

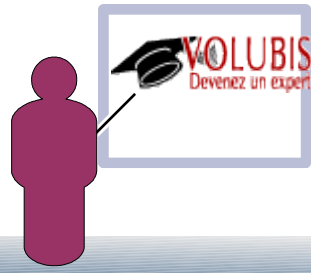


Nouveautés CL : V5R40

Enfin une variable peut pointer directement sur une autre (comme une DS), évitant ainsi les %SST() dans le code.

```
DCL VAR(&Data) TYPE(*CHAR) LEN(256)
DCL VAR(&portion) TYPE(*CHAR) LEN(10) +
    STG(*DEFINED) DEFVAR(&Data 11)
```

&portion recouvrant les Octets 11 à 20 de &Data



Nouveautés CL : V5R40

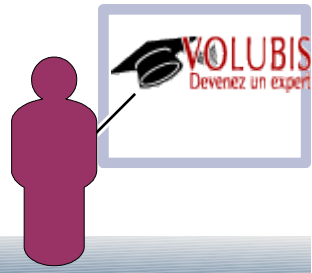
Pour finir, le CL implémente maintenant la notion de sous programme

définition d'un sous programme SUBR / ENDSUBR

- les sous programmes doivent être placés en FIN de pgm

- ils ne doivent pas être imbriqués

- ils peuvent bien sûr s'appeler les uns les autres dans la limite indiquée par la nouvelle commande DLCPRCOPT (dft = 99)



Nouveautés CL : V5R40

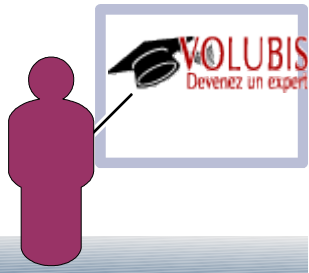
exemple de source CL avec des sous programmes

```
 PGM
   DCL
   DCL

     .../...

     CALLSUBR SUBR(TEST)

  /* fin logique du pgm */
SUBR SUBR(TEST)
  .../...
ENDSUBR
```

le sous programme peut retourner une valeur numérique
ENDSUBR RTNVAL(&rt)

cette valeur est alors récupérée lors de l'appel par
CALLSUBR ..RTNVAL(&R)

(&R doit être une variable de type *INT de 4 octets).



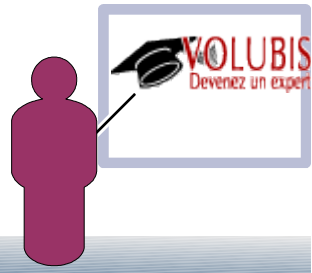
Nouveautés CL : V6R1

Nouvelles commandes utilisables en CL

CLOSE, ferme un fichier ouvert dans un pgm CL
(RCVF l'ouvre à nouveau)

INCLUDE, inclus (copie) un autre membre source CL.

DCLPRCOPT permet de stocker dans le source des options de compilation comme LOG() , ALWRTVSRC() ou USRPRF()



Nouveautés CL : 7.1

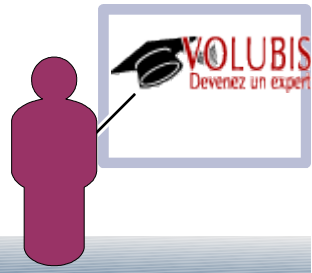
La commande **RTVCLSRC** est admise sur les sources de type CLLE
si vous avez utilisé ALWRTVSRC(*YES) sur CRTCLMOD / CRTBNDCL

Sur les commandes de compilation (CLP ou CLLE) vous pouvez préciser
OPTION(*DOSLTLVL) demandant l'apparition du niveau d'imbrication
sur les listes de compilation (DOWHILE, DOUNTIL, DOFOR, SELECT)

Les zones de type *INT (integer) *UINT (integer non signé) admettent
une longueur sur **8** octets (comme le RPG et le BIGINT de SQL)

La commande INCLUDE (V6) ne pouvait pas inclure un source contenant
lui-même la commande INCLUDE.

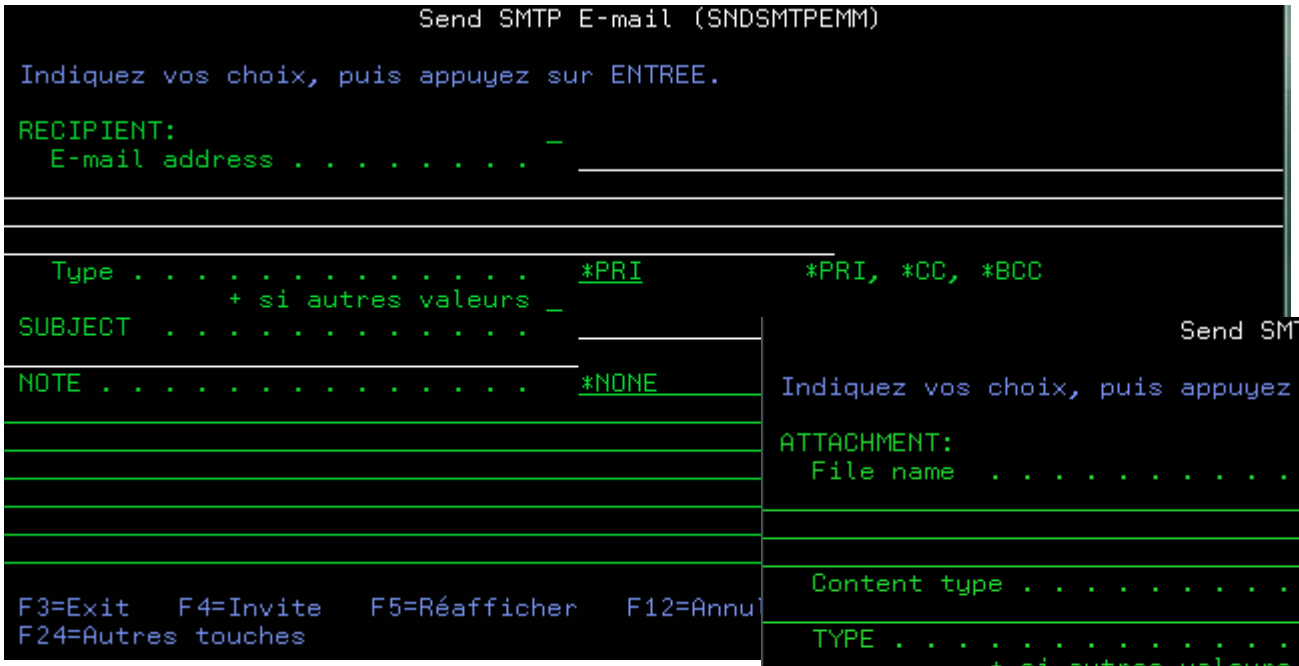
C'est chose faite en V7, sans limitation du nombre de niveau.

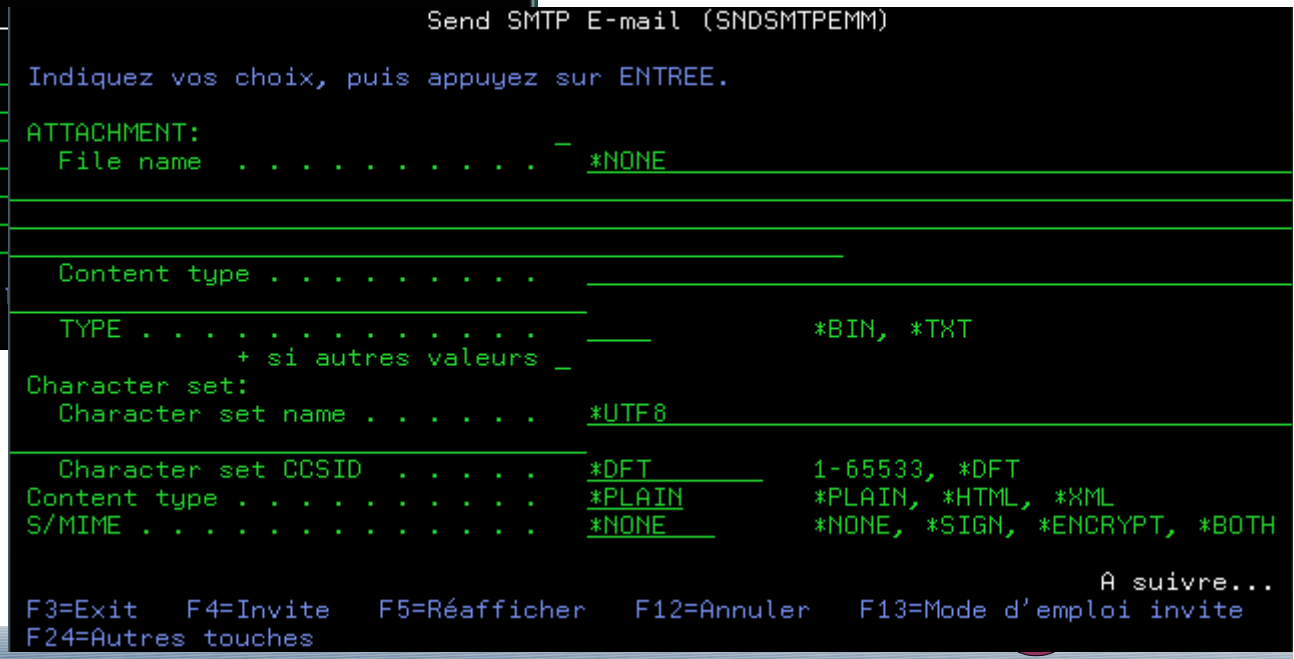


Nouveautés CL : 7.1

Nouvelles commandes

La PTF V7 SI44334 apporte la commande SNDSMTPEMM, (envoi de mail)



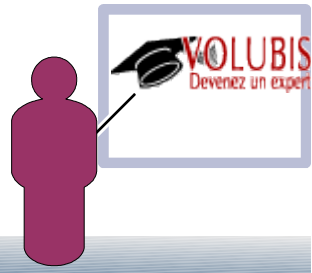


Nouveautés CL : 7.1

Nouvelles commandes

la SI46556 (V7) propose la commande RUNSQL

```
                    Exécuter des instructions SQL (RUNSQL)

Indiquez vos choix, puis appuyez sur ENTREE.

SQL  . . . . . .  _____________________________________________________
                  _____________________________________________________
                  _____________________________________________________
                  _____________________________________________________
                  _____________________________________________________
                  _____________________________________________________
                  _____________________________________________________ ...
Contrôle de validation . . . . .  *CHG          *CHG, *UR, *CS, *ALL, *RS...
Appellation  . . . . . .  *SYS          *SYS, *SQL

                          Autres paramètres

Format de date . . . . . .  *JOB          *JOB, *USA, *ISO, *EUR...
Séparateur de date . . . . . .  *JOB          *JOB, /, ., ,, -, ' ', *BLANK
Format d'heure . . . . . .  *HMS          *HMS, *USA, *ISO, *EUR, *JIS
Séparateur d'heure . . . . . .  *JOB          *JOB, :, ., ,, ' ', *BLANK
                                                                A suivre...
F3=Exit   F4=Invite   F5=Réafficher   F12=Annuler   F13=Mode d'emploi invite
F24=Autres touches
```



Nouveautés CL : 7.1

Comme avec RUNSQLSTM, la plupart des ordres SQL peuvent être utilisés, sauf SELECT

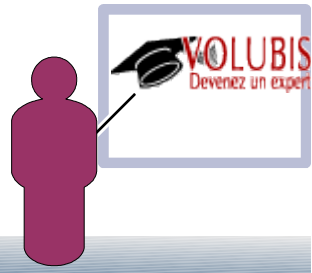
Contrairement à RUNSQLSTM, il n'y a pas :

- de commit automatique , mais vous pouvez utiliser la commande CL COMMIT
- de génération de spool d'erreur, vous recevez le code erreur dans le programme

Exemples d'utilisation donnés par la documentation

```
RUNSQL
SQL('INSERT INTO prodLib/worktable VALUES(1, CURRENT TIMESTAMP)')

RUNSQL
SQL('CREATE TABLE qtemp.worktable AS
      (SELECT * FROM qsys2.systables WHERE table_schema =
          ''MYSCHEMA'') WITH DATA')
    COMMIT(*NONE) NAMING(*SQL)
```

Vous pouvez ensuite lire WORKTABLE par RCVF

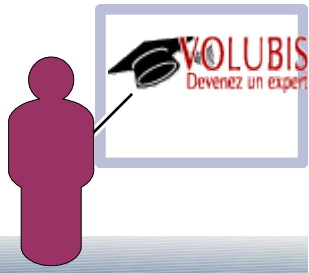


Nouveautés CL : 7.1

Enfin, commande SQL peut être une variable :

```
DCL &LIB TYPE(*CHAR) LEN(10)
DCL &SQLSTMT TYPE(*CHAR) LEN(1000)

 CHGVAR VAR(&SQLSTMT) VALUE('DELETE FROM qtemp.worktable +
                       WHERE table_schema = ''' *cat &LIB *cat ''')

 RUNSQL SQL(&SQLSTMT) COMMIT(*NONE) NAMING(*SQL)
```

Comme avec SNDSMTPEMM, il n'y a pas d'aide associé (à ce jour)



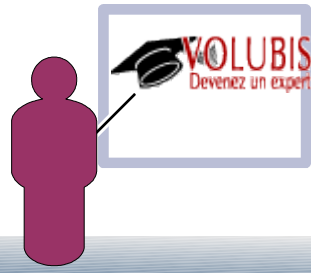
Nouveautés CL : 7.1

Après les fonctions intégrées suivantes :

%SST	extraction d'une chaîne de caractères
%BIN	extraction et conversion d'une sous-chaîne en binaire
%SWITCH	manipulation des switchs du job (concept d'origine 36)
%OFFSET	manipulation d'un pointeur (incrémentation/déplacement)
%ADDRESS	assignation d'un pointeur avec l'adresse d'une variable

La PTF SI48166 propose

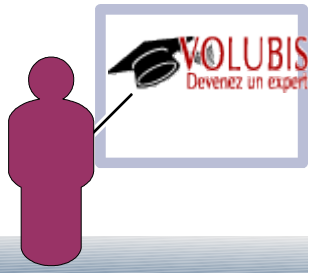
%TRIM	élimination d'un blanc d'extrémité d'une chaîne
%TRIMR	élimination des blancs de droite
%TRIML	élimination des blancs de gauche



Nouveautés CL : 7.1

Exemples :

```
DCL        VAR(&V1) TYPE(*CHAR)  VALUE('   test de chaine*****')
DCL        VAR(&V2) TYPE(*CHAR)
DCL        VAR(&V3) TYPE(*CHAR)


 CHGVAR     VAR(&V2) VALUE(%TRIM(&V1))
            /* contient 'test de chaîne*****' */


/* on peut indiquer le(s) caractère(s) à enlever */

 CHGVAR     VAR(&V3) VALUE(%TRIMR(&V2 '* '))
            /* contient 'test de chaîne' */
```



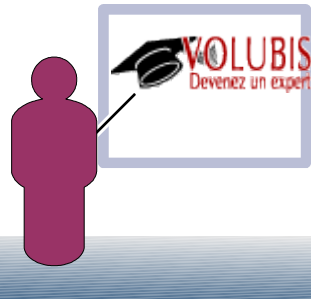
Nouveautés CL : 7.1

Après les fonctions intégrées suivantes :

%SST	extraction d'une chaîne de caractères
%BIN	extraction et conversion d'une sous-chaîne en binaire
%SWITCH	manipulation des switchs du job (concept d'origine 36)
%OFFSET	manipulation d'un pointeur (incrémentation/déplacement)
%ADDRESS	assignation d'un pointeur avec l'adresse d'une variable

La PTF SI49061 propose

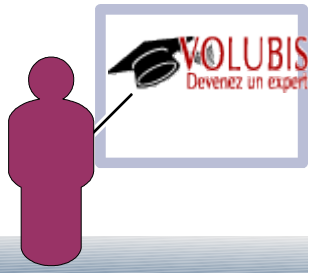
%CHECK	vérification des caractères d'une variable(gauche->droite)
%CHECKR	vérification des caractères d'une variable(droite->gauche)
%SCAN	recherche d'une chaîne dans une variable



Nouveautés CL : 7.1

Exemples :

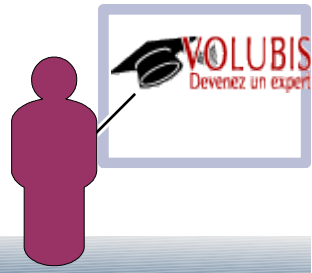
```
DCL        VAR(&V4) TYPE(*CHAR) VALUE('666,66')
DCL        VAR(&V5) TYPE(*INT)


 CHGVAR     VAR(&V5) VALUE(%CHECK('123456789' &V4))
            /* contient 4, la position de ',' */

 CHGVAR     VAR(&V5) VALUE(%CHECK('123456789, ' &V4))
            /* contient 0, tout est OK  */


/* on peut utiliser *LDA à la place d'un nom de variable */

 CHGDTAARA  DTAARA(*LDA (1 10)) VALUE('03216549*7')
 CHGVAR     VAR(&V5) VALUE(%CHECK('123456789' *LDA))
            /* contient 1, la position du '0' */
```



Nouveautés CL : 7.1

Exemples :

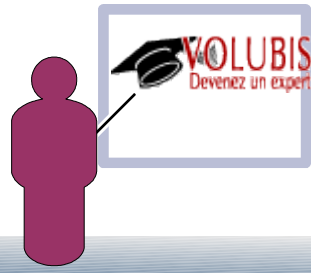
```
/* on peut préciser la position de début de recherche */
CHGDTAARA  DTAARA(*LDA (1 10)) VALUE('03216549*7')
CHGVAR     VAR(&V5) VALUE(&V5 + 1)  /* à partir de  2 */
CHGVAR     VAR(&V5) VALUE(%CHECK('123456789' *LDA &V5))
           /* contient 9, la position de '*' */


/* on peut faire le contrôle de droite à gauche */
 CHGVAR     VAR(&V5) VALUE(%CHECKR('123456789' *LDA))
           /* contient 9, la position de '*' */


/* %scan, mêmes fonctionnalités */
/* rappel, DCL        VAR(&V4) TYPE(*CHAR) VALUE('666,66') */
CHGVAR     VAR(&V5) VALUE(%SCAN(',' &V4))
           /* contient 4 */
CHGVAR     VAR(&V5) VALUE(%SCAN('*' *LDA))
           /* contient 0, non trouvé */
```



Quelques exemples

UN outils assez ancien, CHGPROPRIO :

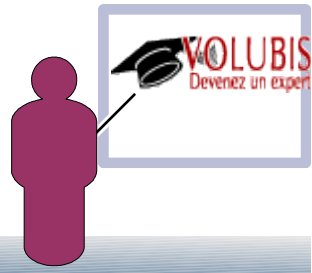
```
            PGM        PARM(&LIB &OWN)
             DCL        VAR(&LIB) TYPE(*CHAR) LEN(10)
             DCL        VAR(&OWN) TYPE(*CHAR) LEN(10)
             DCLF       FILE(QADSPOBJ)
 /* VARIABLES UTILISEES PAR LA GESTION DE MESSGAES */
             DCL        &ERRORSW *LGL                        /* SWITCH  */
             DCL        &MSGID *CHAR LEN(7)                  /* ID MSG  */
             DCL        &MSGDTA *CHAR LEN(100)               /* DATA    */
             DCL        &MSGF *CHAR LEN(10)                  /* FICHIER */
             DCL        &MSGFLIB *CHAR LEN(10)               /* BIBLI   */
             MONMSG CPF0000 *N (GOTO ERRMSG)

                   CHKOBJ     OBJ(&LIB) OBJTYPE(*LIB) AUT(*USE)
                   IF (&OWN *EQ '*CURRENT') GOTO SUIT1
                   CHKOBJ     OBJ(&OWN) OBJTYPE(*USRPRF)
                     GOTO SUIT2
SUIT1:      RTVJOBA    USER(&OWN)
SUIT2:      DSPOBJD    OBJ(&LIB/*ALL) OBJTYPE(*ALL) +
                         OUTPUT(*OUTFILE) OUTFILE(QTEMP/CHGPROP)
             OVRDBF     FILE(QADSPOBJ) TOFILE(QTEMP/CHGPROP)
LECTURE:    RCVF
              MONMSG CPF0864 *N (GOTO FINTRT)
             CHGOBJOWN  OBJ(&ODLBNM/&ODOBNM) OBJTYPE(&ODOBTP) +
                          NEWOWN(&OWN)
             GOTO LECTURE

/* RENVOI DES MESSAGES DE TYPE *COMP SI FIN NORMALE */

COMPMSG:    RCVMSG     MSGTYPE(*COMP) MSGDTA(&MSGDTA) MSGID(&MSGID) +
                         MSGF(&MSGF) MSGFLIB(&MSGFLIB)
```



Quelques exemples

UN outils assez ancien, CHGPROPRIO :

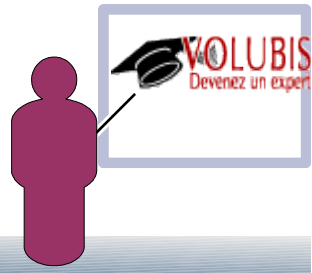
```
             IF         COND(&MSGID *EQ '       ') THEN(RETURN)
             SNDPGMMSG  MSGID(&MSGID) MSGF(&MSGFLIB/&MSGF) +
                          MSGDTA(&MSGDTA) MSGTYPE(*COMP)
             GOTO       COMPMSG /* BOUCLE SUR MESSAGES *COMP      */

ERRMSG:
              /*-----------------------------------------*/
              /*        GESTION DES ERREURS              */
              /*-----------------------------------------*/
             IF         &ERRORSW SNDPGMMSG MSGID(CPF9999) +
                          MSGF(QCPFMSG) MSGTYPE(*ESCAPE) /* 2EME FOIS*/
                                                         /* ARRET PGM*/
             CHGVAR     &ERRORSW '1' /* MISE EN PLACE DU SWTICH     */

 /* RENVOI DES MESSAGES DE TYPE *DIAG SI FIN ANORMALE */
 DIAGMSG:    RCVMSG     MSGTYPE(*DIAG) MSGDTA(&MSGDTA) MSGID(&MSGID) +
                          MSGF(&MSGF) MSGFLIB(&MSGFLIB)
             IF         (&MSGID *EQ '       ') GOTO EXCPMSG
             SNDPGMMSG  MSGID(&MSGID) MSGF(&MSGFLIB/&MSGF) +
                          MSGDTA(&MSGDTA) MSGTYPE(*DIAG)
             GOTO       DIAGMSG /* BOUCLE SUR MESSAGES *DIAG      */

 /* RENVOI DU MESSAGE D'ERREUR                              */
 EXCPMSG:    RCVMSG     MSGTYPE(*EXCP) MSGDTA(&MSGDTA) MSGID(&MSGID) +
                          MSGF(&MSGF) MSGFLIB(&MSGFLIB)
             SNDPGMMSG  MSGID(&MSGID) MSGF(&MSGFLIB/&MSGF) +
                          MSGDTA(&MSGDTA) MSGTYPE(*ESCAPE)

             ENDPGM
```



Quelques exemples

Nouvelle version :

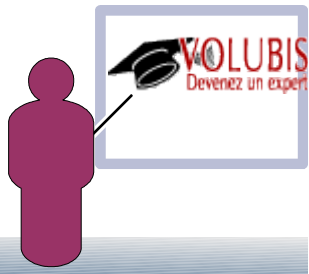
```
PGM         PARM(&LIB &OWN)
               DCL         VAR(&LIB) TYPE(*CHAR) LEN(10)
               DCL         VAR(&OWN) TYPE(*CHAR) LEN(10)
               DCL         VAR(&EOF) TYPE(*LGL)
               DCLF        FILE(QADSPOBJ)
 /* DCL MSG*/ INCLUDE     SRCMBR(MSGINC1)

               CHKOBJ      OBJ(&LIB) OBJTYPE(*LIB) AUT(*USE)
               SELECT
                WHEN (&OWN *NE '*CURRENT') THEN( +
                  CHKOBJ     OBJ(&OWN) OBJTYPE(*USRPRF) )
                OTHERWISE  CMD(RTVJOBA USER(&OWN))
               ENDSELECT

               DSPOBJD     OBJ(&LIB/*ALL) OBJTYPE(*ALL) +
                             OUTPUT(*OUTFILE) OUTFILE(QTEMP/CHGPROP)
               OVRDBF      FILE(QADSPOBJ) TOFILE(QTEMP/CHGPROP)

               CALLSUBR    LECTURE
               DOWHILE     COND(*NOT &EOF)
                CHGOBJOWN  OBJ(&ODLBNM/&ODOBNM) OBJTYPE(&ODOBTP) +
                             NEWOWN(&OWN)
                CALLSUBR LECTURE
               ENDDO

               SNDPGMMSG  MSG(&OWN *BCAT 'est propriétaire de tous les +
                             objets de la bibliothèque ' *CAT &LIB) MSGTYPE(*COMP)

 /*MESSAGES*/ INCLUDE     SRCMBR(MSGINC2)
 SUBR        SUBR(LECTURE)
               RCVF
                MONMSG      MSGID(CPF0864) EXEC(CHGVAR VAR(&EOF) VALUE('1'))
 ENDSUBR
ENDPGM
```



Quelques exemples

Source à inclure MSGINC2 : :

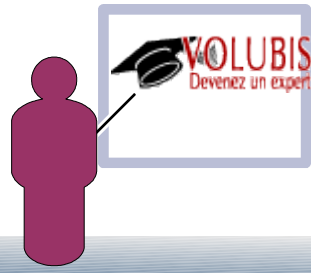
```
   /* RENVOI DES MESSAGES DE TYPE *COMP SI FIN NORMALE */

             CALLSUBR MSGCOMP
             DOWHILE   COND(&MSGID *NE '       ')
              SNDPGMMSG  MSGID(&MSGID) MSGF(&MSGFLIB/&MSGF) +
                           MSGDTA(&MSGDTA) MSGTYPE(*COMP)
              CALLSUBR MSGCOMP
             ENDDO
             RETURN

ERRMSG:
              /*-------------------------------------*/
              /*        GESTION DES ERREURS          */
              /*-------------------------------------*/
             IF         &ERRORSW SNDPGMMSG MSGID(CPF9999) +
                          MSGF(QCPFMSG) MSGTYPE(*ESCAPE) /* 2EME FOIS*/
                                                          /* ARRET PGM*/
              CHGVAR     &ERRORSW '1' /* MISE EN PLACE DU SWTICH     */

 /* RENVOI DES MESSAGES DE TYPE *DIAG SI FIN ANORMALE */
              CALLSUBR MSGDIAG
              DOWHILE   COND(&MSGID *NE '       ')
               SNDPGMMSG  MSGID(&MSGID) MSGF(&MSGFLIB/&MSGF) +
                            MSGDTA(&MSGDTA) MSGTYPE(*DIAG)
               CALLSUBR MSGDIAG
              ENDDO
```



Quelques exemples

Source à inclure MSGINC2 : :

```
    /* RENVOI DU MESSAGE D'ERREUR FINAL      */

 EXCPMSG:    RCVMSG     MSGTYPE(*EXCP) MSGDTA(&MSGDTA) MSGID(&MSGID) +
                          MSGF(&MSGF) MSGFLIB(&MSGFLIB)
             SNDPGMMSG  MSGID(&MSGID) MSGF(&MSGFLIB/&MSGF) +
                         MSGDTA(&MSGDTA) MSGTYPE(*ESCAPE)


    /* sous programmes */

SUBR        SUBR(MSGCOMP)
              RCVMSG     MSGTYPE(*COMP) MSGDTA(&MSGDTA) MSGID(&MSGID) +
                           MSGF(&MSGF) MSGFLIB(&MSGFLIB)
ENDSUBR

SUBR        SUBR(MSGDIAG)
              RCVMSG     MSGTYPE(*DIAG) MSGDTA(&MSGDTA) MSGID(&MSGID)
                           MSGF(&MSGF) MSGFLIB(&MSGFLIB)
ENDSUBR
```



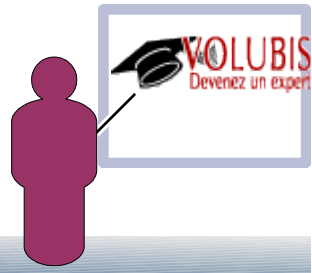
Quelques exemples

Autres exemples

L'api ***'stat'*** indique si un fichier stream (IFS) existe :

Cette api étant dans le programme de service QLECWI, il faut utiliser le type CLLE

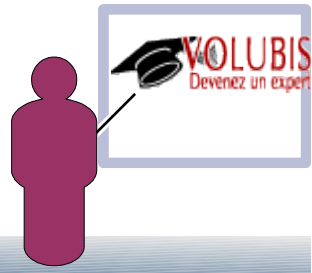
```
PGM         PARM(&PARM)
DCLPRCOPT   BNDSRVPGM((QLECWI))

DCL         VAR(&PARM)      TYPE(*CHAR) LEN(256)
DCL         VAR(&RTNVAL)    TYPE(*int) LEN(4)
DCL         VAR(&PATH)      TYPE(*CHAR) LEN(100)
DCL         VAR(&NULL)      TYPE(*CHAR) LEN(1) VALUE(X'00')
DCL         VAR(&BUF)       TYPE(*CHAR) LEN(4096)

CHGVAR      VAR(&PATH) VALUE(&PARM *TCAT &NULL)

CALLPRC     PRC('stat') PARM(&PATH &BUF) +
              RTNVAL(&RTNVAL)

IF          COND(&RTNVAL *NE 0) THEN(SNDPGMMSG +
              MSGID(CPF9898) MSGF(QCPFMSG) +
               MSGDTA('Objet ' !! &PARM !< ' non +
               trouvé.') MSGTYPE(*ESCAPE))

ENDPGM
```



Quelques exemples

L'api QDCRDEVD contient l'adresse IP du terminal en 878 (sur 15 c.)

```
PGM    PARM(&IPADR)
 DCL          VAR(&RETOUR) TYPE(*CHAR) LEN(900)
 DCL          VAR(&RETOUR_IP) TYPE(*CHAR) STG(*DEFINED) LEN(15) +
               DEFVAR(&RETOUR 878)
 DCL          VAR(&BIN) TYPE(*INT) LEN(4) VALUE(900)
 DCL          VAR(&FORMAT) TYPE(*CHAR) LEN(8) VALUE('DEVD0600')
 DCL          VAR(&DEV) TYPE(*CHAR) LEN(10)
 DCL          VAR(&ERR) TYPE(*CHAR) LEN(8) +
                        VALUE(X'0000000000000000')
 DCL          VAR(&IPADR) TYPE(*CHAR) LEN(15)

      RTVJOBA     JOB(&DEV)

      CALL        QDCRDEVD  PARM(&RETOUR &BIN &FORMAT &DEV &ERR  )

      CHGVAR      &IPADR &retour_IP

ENDPGM
```

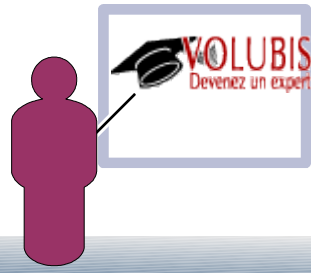


Quelques exemples

Arrêt de tous les jobs ayant le même nom.

On utilise la commande ENDJOB DUPJOBOPT(*MSG) et s'il y a des noms dupliqués , lecture (par RCVMSG) de tous les messages d'erreur pour en récupérer les coordonnées

```
PGM         PARM(&JOB)
 DCL &JOB *CHAR 10
 DCL &USR *CHAR 10
 DCL &NBR *CHAR  6
 DCL &MSGDTA *CHAR 26
 DCL &MSGID *CHAR 7

 ENDJOB     JOB(&JOB) OPTION(*IMMED) DUPJOBOPT(*MSG)
    MONMSG CPF0000


  /*  A suivre … */
```



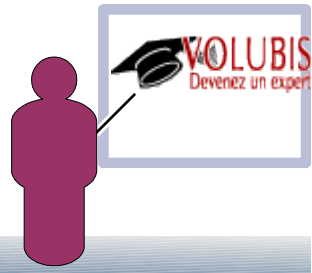
Quelques exemples

Arrêt de tous les jobs ayant un nom donné.

On utilise la commande ENDJOB DUPJOBOPT(*MSG) et s'il y a des noms dupliqués , lecture (par RCVMSG) de tous les messages d'erreur pour en récupérer les coordonnées

/* si le nom est dupliqué */

```
RCVMSG      PGMQ(*SAME (*)) MSGDTA(&MSGDTA) MSGID(&MSGID)
DOWHILE     COND(&MSGID *NE ' ')
  IF (&MSGID = 'CPF0906') THEN(DO)
     CHGVAR &USR %SST(&MSGDTA 11 10)
     CHGVAR &NBR %SST(&MSGDTA 21 6)
     ENDJOB      JOB(&NBR/&USR/&JOB) OPTION(*IMMED)
       MONMSG CPF0000
  ENDDO
  RCVMSG      PGMQ(*SAME (*)) MSGDTA(&MSGDTA) MSGID(&MSGID)
ENDDO

ENDPGM
```



Quelques exemples

Destruction de tous les récepteurs d'une bibliothèque.

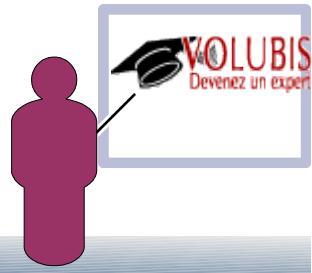
L'api QUSLOBJ donne une liste d'objets dans un *USRSPC

L'api QUSPTRUS retrouve le pointeur de début d'un User Space

Dans l'entête du uSer Space, en position 124 on trouve, en binaire :

- La position de début de la liste d'objets
- Le nombre de postes (c.a.d d'objets dans notre exemple)
- la taille d'un poste

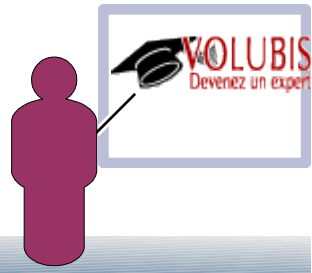
```
             PGM        PARM(&BIB)
 /* ============================================================== */
 /* BUT : lister les RÉCEPTEURS DE JOURNAUX AFIN DE LES DÉTRUIRE   */
 /*        sauf ceux attachés                                      */
 /* ============================================================== */
             DCL        VAR(&COMPTEUR) TYPE(*INT)
             DCL        VAR(&QUAL) TYPE(*CHAR) LEN(20) VALUE(*ALL)
             DCL        VAR(&BIB) TYPE(*CHAR) LEN(10)
             DCL        VAR(&pointeur) TYPE(*PTR)
```



Quelques exemples

Destruction de tous les récepteurs d'une bibliothèque.

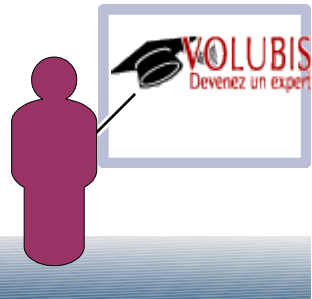
```
DCL         VAR(&ptrinfos) TYPE(*PTR)
DCL         VAR(&DATA) TYPE(*CHAR) STG(*BASED) LEN(16) +
              BASPTR(&PTRINFOS)

DCL         VAR(&DEBUT)  TYPE(*INT) STG(*DEFINED) DEFVAR(&DATA)
DCL         VAR(&NOMBRE) TYPE(*INT) STG(*DEFINED) DEFVAR(&DATA  9)
DCL         VAR(&TAILLE) TYPE(*INT) STG(*DEFINED) DEFVAR(&DATA 13)


DCL         VAR(&ptrretour) TYPE(*PTR)
DCL         VAR(&RETOUR) TYPE(*CHAR) STG(*BASED) LEN(30) +
                         BASPTR(&PTRRETOUR)

DCL         VAR(&OBJ) TYPE(*CHAR) STG(*DEFINED) LEN(10) +
                         DEFVAR(&RETOUR)
DCL         VAR(&OBJLIB) TYPE(*CHAR) STG(*DEFINED) LEN(10) +
                         DEFVAR(&RETOUR 11)
DCL         VAR(&OBJTYPE) TYPE(*CHAR) STG(*DEFINED) LEN(10) +
                         DEFVAR(&RETOUR 21)
```



Quelques exemples

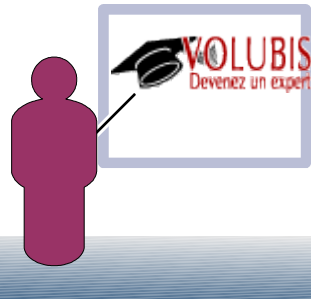
Destruction de tous les récepteurs d'une bibliothèque.

```
   /* VARIABLES UTILISEES PAR LA GESTION DE MESSAGES */
       DCL        &MSGID   *CHAR LEN(7)              /* ID MSG  */
       DCL        &MSGDTA  *CHAR LEN(100)            /* DATA    */
       DCL        &MSGF    *CHAR LEN(10)             /* FICHIER */
       DCL        &MSGFLIB *CHAR LEN(10)             /* BIBLI   */
         MONMSG     MSGID(CPF0000) EXEC(GOTO ERREUR)

       DLTUSRSPC QTEMP/DLTLIBRCV
         MONMSG     MSGID(CPF2105) EXEC(RCVMSG PGMQ(*SAME) +
                    MSGTYPE(*EXCP))

 /* CRÉATION DU USER SPACE */
       CALL       PGM(QUSCRTUS) PARM('DLTLIBRCV QTEMP' /* USRSPC      */ +
                                     '          '      /* ATTRIBUT    */ +
                                     X'0000FFFF'       /* TAILLE      */ +
                                     X'00'             /* VAL INITIALE*/ +
                                     '*USE'            /* DROITS      */ +
                                     'POUR DLTLIBRCV') /* TEXTE       */
```



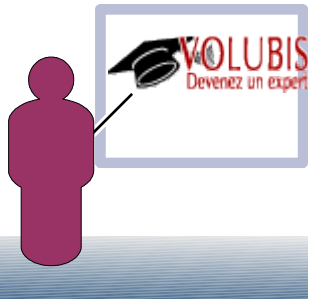
Quelques exemples

```
/* REMPLISSAGE, LISTE DES OBJETS */
        CHGVAR     VAR(%SST(&QUAL 11 10)) VALUE(&BIB)
        CALL QUSLOBJ  PARM('DLTLIBRCV QTEMP'  /* USRSPC   */       +
                           'OBJL0100'          /* FORMAT   */       +
                           &QUAL               /* bib/obj  */       +
                           '*JRNRCV'           /* type     */       +
                          )

 /* positionnement sur début du USer Space */
        CALL       PGM(QUSPTRUS) PARM('DLTLIBRCV QTEMP' &Pointeur)

 /* récupération de &DATA, donc de &DEBUT &TAILLE et &NOMBRE    */
        chgvar     &ptrinfos &pointeur
        CHGVAR     %OFFSET(&ptrinfos) VALUE(%OFFSET(&ptrinfos) + 124)

/* positionnement début de liste (on place &retour DANS le User Space) */
        chgvar     &ptrretour &pointeur
        CHGVAR     %OFFSET(&ptrretour) VALUE(%OFFSET(&ptrretour) &DEBUT )
```



Quelques exemples

```
/* Lecture suivant le nombre d'objets indiqué */

     DOFOR       VAR(&COMPTEUR) FROM(1) TO(&NOMBRE) BY(1)

          DLTJRNRCV  JRNRCV(&OBJLIB/&OBJ)
            MONMSG     MSGID(CPF7022)


         /* positionnement sur le suivant */
          if          (&compteur < &nombre) then(do)
             CHGVAR     %OFFSET(&ptrretour) VALUE(%OFFSET(&ptrretour) +
                                + &TAILLE)
          ENDDO

     ENDDO
```



Quelques exemples

```
/* RENVOI MESSAGE DE TYPE *COMP SI FIN NORMALE */

 COMPMSG:
         DLTUSRSPC QTEMP/DLTLIBRCV
         SNDPGMMSG  MSG('Ménage sur les récepteurs de journaux +
                        effectué') TOPGMQ(*PRV (*PGMBDY)) +
                        MSGTYPE(*COMP)
         return


                      /* RENVOI DU MESSAGE D'ERREUR RECU  */
 ERREUR:
         RCVMSG     MSGTYPE(*EXCP) MSGDTA(&MSGDTA) MSGID(&MSGID) +
                    MSGF(&MSGF) MSGFLIB(&MSGFLIB)
         SNDPGMMSG  MSGID(&MSGID) MSGF(&MSGFLIB/&MSGF) +
                        MSGDTA(&MSGDTA) TOPGMQ(*PRV (*PGMBDY)) +
                        MSGTYPE(*ESCAPE)
ENDPGM
```