Tips & Tricks Abap

Jordi Calopa Bosch

Revisión: 2006.08

Indice

Muy Importante

Los programas que se muestran en este documento fueron desarrollados por el autor y se ofrecen únicamente a título informativo y absolutamente sin ninguna garantía. El lector puede utilizarlos a modo de consulta, presuponiendo siempre que su utilización estará sujeta a los postulados de la buena fe.

1. Tips & Tricks

Arrancar un programa concreto desde nuestro programa abap

```
report yxxxxxxx.
submit nombre_del_programa.
```

Arrancar un programa cuyo nombre está en una variable

```
report yxxxxxxx.
data: nombre(8) type c value 'nombre_de_programa'.
submit (nombre).
```

Bloquear la entrada de un criterio de selección

```
*
select-options: soc for t001-bukrs,
                cen for t001w-werks.
*
initialization.
  soc-sign   = 'i'.
  soc-option = 'bt'.
  soc-low    = '0200'.
  soc-high   = '0299'.
  append soc.
  cen-sign   = 'i'.
  cen-option = 'bt'.
  cen-low    = '01b1'.
  cen-high   = '01c1'.
  append cen.
*
loop at screen.
  if screen-name+00(3) eq 'SOC'.
    screen-input = 0.                ' Bloqueamos la entrada SOC
    screen-intensified = 1.
    modify screen.
  endif.
endloop.
```

Borrar todos los valores personales de un matchcode

```
program zjcbxxx.
include radshmac.
type-pools: SHLP.
data: campo TYPE shlp_descr_t.
campo-shlpname = 'ZSAKO1'.           ‘ Nombre del MatchCode
campo-shlptype = 'SH'.
perform personal_values_del_all(saplsdsd) using campo.
*
Nota: también podemos acceder directamente a la tabla DDSHPVAL50 o
DDSHPVALUE
```

Buscar un string

```
report yxxxxxxx.
data: a(10) type c value 'aeioujklmn'.   ' campo en el cual se
busca
data: b(01) type c value 'j'.            ' valor a buscar"
start-of-selection.
search a for b.
write: sy-subrc, sy-fdpos.              '   sy-fdpos   contiene   el
valor del offset
                              donde se ha encontrado el
                                        '   string   buscado   (en
nuestro ejemplo, el 5)
```

Calcular fechas

```
Primeramente:   D_FECHA = SY-DATUM.   “Por ejemplo.
```

- Primer día del mes

```
D_FECHA = D_FECHA – D_FECHA+6(2) + 1
```

- Ultimo día del mes

```
D_FECHA = D_FECHA – D_FECHA+6(2) + 33
D_FECHA = D_FECHA – D_FECHA+6(2)
```

- Primer día del mes siguiente

```
D_FECHA = D_FECHA – D_FECHA+6(2) + 33
D_FECHA = D_FECHA – D_FECHA+6(2) + 1
```

- Ultimo día del mes anterior

```
D_FECHA = D_FECHA – D_FECHA+6(2)
```

Calcular la raíz cuadrada de un numero

```
report yxxxxxxx.
data: ind2(3)    type n value 100,
      raiz       type f.
      raiz = sqrt( ind2 ).
write: / 'la raíz cuadrada de',ind2, 'es', raiz.
```

Calcular el factorial de un número

```
report yxxxxxxx.
data: factorial   type f value 1,
      numero(02)  type n value 1.
while numero lt '51'.
   write: / 'el factorial de',
            numero,
            'es',
            factorial.
    numero = numero + 1.
   factorial = factorial * numero.
endwhile.
```

Calcular el logaritmo neperiano

```
report yxxxxxxx. "
 data: campo1   type f value 1,
       campo2   type f value 0,
       numero_e type f value 0.
       numero_e = exp( campo1 ).
write: /01 ' 1 elevado a e =', numero_e color 5.
campo2   = log( numero_e ).      "
write: /01 ' logaritmo neperiano de e =', campo2 color 5.
```

Calcular el valor absoluto de un número

```
report yxxxxxxx.
data: valor1 type p value '-123',
      valor2 type p.
compute valor2 = abs( valor1 ).
write: /01 valor1,                    ' Imprime 123-
           valor2.                    ' Imprime 123
```

Calcular potencias de base “base”

```
report yxxxxxxx.
data: ind1(2)         type n value 0,
      potencia(10)    type n value 1,
      base(01)        type n value 2."
while ind1 lt '32'.
   ind1 = ind1 + 1.
   potencia = potencia * base.
   write: / '2 elevado a',ind1,'=',potencia color 7"
endwhile.
```

Calcular integral definida y = 2x en el intervalo de x de 0 a 10

```
report yxxxxxxx.
data:  y(5)                type p   decimals 2  value  '0,00',
       x(5)                type p   decimals 2  value  '0,00',
       incremen(5)         type p   decimals 2  value  '0,01',
       maximo_de_x(5)      type p   decimals 2  value '10,00',
       mini_area(7)        type p   decimals 4  value  '0,0000',
       integral_definida(7) type p   decimals 4  value  '0,0000'.
start-of-selection.
while x lt maximo_de_x.
   x = x + incremen.
   y = x + x.
   mini_area = incremen * y.
   integral_definida = integral_definida + mini_area.
endwhile.
```

Calcular los n términos de una progresión aritmética

```
report ymltest.
data: n        type i value 0,              '     cantidad;     de;
elementos
      suma     type f value 0,              '    suma;    de;    la;
progresi¢n
      w1  type i value 0.              ' campo de trabajo
parameters:  a1     type i default 1,   ' primer término
             an     type i default 9,   ' ultimo término
             increm   type i default 1. ' incremento
start-of-selection.
w1 = a1.  "
while w1 le an.
  w1 = w1 + increm.
  n = n + 1.
 endwhile. "
 suma = ( n * ( ( a1 + an ) / 2 ) ). 'suma n terminos de una p.a.
 write: /01 suma.
```

Cambiar de password tantas veces como se precise

```
Program yxxxxxxx.
tables: usr02.
start-of-selection.
* ejecutar este programa antes de cambiar el password
   select single * from usr02
          where bname eq sy-uname.
   if sy-subrc eq 0.
      usr02-ocod1 = '00000000'.
      usr02-bcda1 = '99991231'.
      usr02-ocod2 = '00000000'.
      usr02-bcda1 = '99991231'.
      usr02-ocod3 = '00000000'.
      usr02-bcda3 = '99991231'.
      usr02-ocod4 = '00000000'.
      usr02-bcda4 = '99991231'.
      usr02-ocod5 = '00000000'.
      usr02-bcda5 = '99991231'.
      usr02-trdat = sy-datum - 1.
      update usr02.
   endif.
```

Condensar una estructura

```
report yxxxxxxx.
data: begin of nombre_struc,
   a(10), b(7), c(12), d(10), e(14),
end of nombre_struc.
start-of-selection.
   nombre_struc-a = 'uno       '.
   nombre_struc-b = '   dos '.
   nombre_struc-c = 'tres        '.
   nombre_struc-d = '   cuatro '.
   nombre_struc-e = 'cinco         '.
   condense nombre_struc.                    ' concatena
   write:/01 nombre_struc.                     ' imprime literal -
uno dos tres cuatro cinco-
```

Convertir mayúsculas / minúsculas

```
program zxxxxxx.
data: campo  type string   value 'abcd'.
start-of-selection.
```

```
  write: /01 Campo.
  perform MayMin using CAMPO 'U'.  " Convertir a Mayúsculas
  write: /01 Campo.
  perform MayMin using CAMPO 'L'.  " Convertir a Minúsculas
  write: /01 Campo.
*
form MayMin using string tipo.
  if  tipo = 'U'.
    translate string to upper case.
  else.
    translate string to lower case.
  endif.
endform.                    "MayMin
```

Convertir un tipo packed con decimales a character

```
report yxxxxxxx.
data: campo1 type p decimals 2 value '123,45',
      campo2(10) type c.
write campo1 to campo2 no-sign.         ' convertimos tipo
translate campo2 using ' 0'.            '   cambiamos   blancos   por
ceros
write: / campo1,  campo2."

* Al objeto de facilitar la utilización de decimales, es posible
realizar todos los cálculos en coma flotante y posteriormente
convertir el campo al formato requerido."
report yxxxxxxx.                        "
data:    c1(8)   type c.                "
data:    a(8)    type c value '3'.      "
data:    b(10)   type p decimals 2.     "
data:    f       type f.                "
f = 100 / a.                            "
write  f  exponent  0  decimals  2  to  c1.       '<--  conversión  al
formato"
move c1 to b.                            '<-- requerido."
write: / a, f, b, c1."
```

Declarar una tabla en tiempo de ejecución

```
report yxxxxxxx.
data: nombre(10) type c value 'T9JCB',
      work_area(100).
start-of-selection.
select descr1 from (nombre) into work_area.
   write: /01 work_area.
endselect.
```

Detectar un año bisiesto

```
Function Y_BISIESTO.
  DATA: ANY     TYPE I VALUE 0,
        ENTERO  TYPE P,
        RESTO   TYPE P.
*
  IF PARAM1 CO '0123456789'.     " Validamos en valor de entrada
     MOVE PARAM1 TO ANY.
  ELSE.
     PARAM2 = 'E'.               " E = Error
     EXIT.                       " Salida de la función, con error
  ENDIF.
*
  RESTO = ANY MOD 4.
  IF RESTO NE 0.
    PARAM2 = 'N'.                " N = No es bisiesto
    EXIT.
  ELSE.
    RESTO = ANY MOD 100.
    IF RESTO NE 0.
      PARAM2 = 'S'.              " S = Si es bisiesto
      EXIT.
    ELSE.
      RESTO = ANY MOD 400.
      IF RESTO = 0.
        PARAM2 = 'S'.            " S = Si es bisiesto
      ELSE.
        PARAM2 = 'N'.            " N = No es bisiesto
      ENDIF.
    ENDIF.
  ENDIF.
ENDFUNCTION.
```

Ejecutar comandos de unix/linux

```
report yxxxxxxx
       line-size 250
       no standard page heading.
data: begin of t1 occurs 100,
         line(200),
      end of t1.
data: w_comando_unix(250)   type c.
parameters: wcom(70) type c lower case obligatory.
start-of-selection.
  perform ejecuta_comando.
```

```
  perform listar_t1.
end-of-selection.
*
form ejecuta_comando.
  concatenate wcom ' ' into w_comando_unix separated by space.
  refresh t1.
  CALL 'SYSTEM' ID 'COMMAND' FIELD W_COMANDO_UNIX
                ID 'TAB'  FIELD T1-*SYS*.
endform.
*
form listar_t1.
  format intensified on.
  format color col_heading.
  uline.
  loop at t1.
    write: /01 sy-vline,
               t1-line,
           at sy-linsz sy-vline.
  endloop.
  uline.
endform.
```

Enviar mensajes a un usuario (conectado al sistema)

```
program zxxxxxxx.
data: opcode_send_pop_up(1) type x value 31,
      longitud              like sy-index,
      loc_cut_blanks        type x value 1,
      wuser                 like sy-uname.
selection-screen: begin of block b1 with frame title text-001.
selection-screen: skip.
parameters: mensaje(128)  type c lower case.
parameters: usuario       like sy-uname default sy-uname,
            mandante      like sy-mandt default sy-mandt.
selection-screen: skip.
selection-screen: end of block b1.
*
initialization.
  mensaje = 'escriba aqui su mensaje '.
start-of-selection."
  describe field mensaje length longitud.
  perform funcion_enviar_mensaje.
*
form funcion_enviar_mensaje.
  CALL 'ThUsrInfo' ID 'OPCODE' FIELD OPCODE_SEND_POP_UP
     ID 'CLIENT'     FIELD MANDANTE
     ID 'USR'        FIELD USUARIO
     ID 'MSG'        FIELD MENSAJE
     ID 'MSG_LEN'    FIELD LONGITUD
```

```
     ID 'CUT_BLANKS' FIELD LOC_CUT_BLANKS.
endform.
```

Evitar que caduque la constraseña periodicamente

```
report zxxxxxxx.
tables: usr02.
start-of-selection.
   select single for update * from usr02
          where bname eq sy-uname.
    if sy-subrc eq 0.
       usr02-bcda1 = '99991231'.
       update usr02.
    endif.
```

Generar un programa dinámicamente

```
program zxxxxxxx.
data: NomProg like sy-repid value 'ZJCB018'.
data: begin of t1 occurs 999,
         reg(255) type c,
      end of t1.
start-of-selection.
  t1-reg = '*
********************************************************** *'.
append t1.
  t1-reg = '* *** Autor: ……
*** *'.        append t1.
  t1-reg = '*
********************************************************** *'.
append t1.
  t1-reg = 'program ZJCB018.'.
append t1.
  t1-reg = 'start-of-selection.'.
append t1.
  t1-reg = 'write: /01 ''Este programa se ha generado
dinámicamente''.'.             append t1.
  INSERT REPORT NomProg FROM T1.
```

Generar y borrar una tabla de bd (sql nativo)

```
exec sql.
  create table tabla_work (
         campo1    char(20) not null,
         primary key (campo1)
```

```
         )
endexec.
*
exec sql.
   drop table tabla_work
endexec.
```

Generar menú dinámicamente (a partir de las forms del propio programa)

```
program zxxxxxx.
data: begin of t1 occurs 3000,
         reg(255) type c,
      end of t1.
data: begin of t2 occurs 5,
         mnu(30) type c,
      end of t2.
data: selecc     like sy-tabix,
      longi      type i.
start-of-selection.
  read report sy-repid into t1.
  loop at t1.
    if t1-reg+00(04) = 'form'.
      longi = strlen( t1-reg ) - 5.
      move t1-reg+04(longi) to t2-mnu.
      append t2.
    endif.
  endloop.
  CALL FUNCTION 'POPUP_WITH_TABLE_DISPLAY'
    EXPORTING
      ENDPOS_COL   = 70
      ENDPOS_ROW   = 21
      STARTPOS_COL = 41
      STARTPOS_ROW = 3
      TITLETEXT    = 'JCB - TOOLS'
    IMPORTING
      CHOISE       = selecc
    TABLES
      VALUETAB     = t2
    EXCEPTIONS
      BREAK_OFF    = 1
      OTHERS       = 2.
  if sy-subrc eq 0.
    perform selecc OF subrutina1 subrutina2 subrutina3.
  endif.
form subrutina1.
  write: /01 'UNO'.
endform.
"subrutina1
form subrutina2.
```

```
  write: /01 'DOS'.
endform.
"subrutina2
form subrutina3.
  write: /01 'TRES'.
endform.
"subrutina3
```

Insertar registro en tabla de bd

```
report yxxxxxxx."
tables: tvarv."
start-of-selection."
  tvarv-name = 'FI_NUEVA_VARIABLE'.  'informamos los campos clave
  tvarv-type = 'P'."
  tvarv-numb = '0000'."
  insert into tvarv values tvarv.    'insertamos registro
```

Insertar registros en tabla interna

```
- Inicializándola
report yxxxxxxx.
tables: usr03.
data: begin of t1 occurs 1000,
         bname like usr03-bname,             ' código de usuario
         name1 like usr03-name1,             ' nombre PRIMERO
      end of t1.
start-of-selection.
select bname name1 into table t1 from usr03.  ' Refresh t1
sort t1.
loop at t1.
   write: /01 t1-bname,
              t1-name1.
endloop.

- SIN Inicializarla
report yxxxxxxx.
tables: usr03.
data: begin of t1 occurs 1000,
         bname like usr03-bname,             'código de usuario
         name1 like usr03-name1,             'nombre PRIMERO
      end of t1.
start-of-selection.
select * appending corresponding fields       'append t1
         of table t1 from usr03.
sort t1.
```

```
loop at t1.
   write: /01 t1-bname,
              t1-name1.
endloop.
```

Justificar a la derecha un campo de tipo character

```
program ysux0800.
parameters: inp(10) type c.
data: out(10) type c.
start-of-selection.
   write inp to out right-justified.
   write: /01 inp,
          /01 out.
```

Nota: Esta sentencia actúa correctamente si los campos han sido definidos como TYPE C, pero no funciona correctamente cuando se definen como LIKE de un campo CHAR. Para este último caso véase la sentencia SHIFT.

Listar el contenido de los system fields

```
program yxxxxxxx.
tables: dd03l.

data: nomcamp(8)      type c value 'SY-',
      valcamp(50)     type c.

start-of-selection.
   select * from dd03l where tabname eq 'SYST'.
      nomcamp+3 = dd03l-fieldname.
      write (nomcamp) to valcamp.
      write: /01 nomcamp color col_key,
                 valcamp color col_normal.
   endselect.
```

Listar los colores standard

```
program zcolors.
data: variable_color type i value 0.
start-of-selection.
  do 8 times.
    format intensified on color = variable_color.
    write: /01 'intensified ON ; color',
               variable_color.
    format intensified off color = variable_color.
```

```
    write: /01 'intensified OFF; color',
               variable_color.
    variable_color = sy-index.
  enddo.
```

Multiplicar shiftando

```
program zxxxxxx.
data: numero(8)   type n    value 1,
      entero      type i.

start-of-selection.
  do 7 times.
    move numero to entero.
    write: /01 entero.
    shift numero circular.
  enddo.
```

Obtener información del sistema

```
program yxxxxxxx.
data: nombre_del_sistema(100)  type c,
      sistema_gestion_bd(100)  type c.
start-of-selection.
* La lista completa de todos los parámetros disponibles se puede
obtener
* mediante el report RSPARAM, facilitado por SAP.
*
* *** Identificación del Sistema ***
  call 'C_SAPGPARAM' id 'NAME'    field 'SAPSYSTEMNAME'
                     id 'VALUE'   field nombre_del_sistema.
  write: /01 nombre_del_sistema.
* *** Sistema de Gestión de BD ***
  call 'C_SAPGPARAM' id 'NAME'    field 'dbms/type'
                     id 'VALUE'   field sistema_gestion_bd.
  write: /01 sistema_gestion_bd.
```

Obtener la longitud de un campo

```
report yxxxxxxx.
   data : campo(15) type c value '0123456789',
          longi type i.
          longi = strlen( campo ).
   write: /01 longi.                        ' Imprime el valor 10
```

Provocar un tiempo de espera (entre 1 y 5 segundos)

```
report yxxxxxxx.
data: int_seconds type i value 5.
start-of-selection."
  get time.
  write: /01 sy-uzeit.
  call 'ALERTS' id 'ADMODE'         field 50
                id 'STORAGE_OPCODE' field 'SLEEP'
                id 'TIME'           field int_seconds.
get time.
 write: /01 sy-uzeit.
```

Quitar los blancos de un campo tipo string

```
report ymltest.
data: begin of nombre_struc,
         a(10),
      end of nombre_struc.
start-of-selection.
   nombre_struc-a = 'a b c     '.
   condense nombre_struc no-gaps.      '   el   parámetro   no-gaps
quita los blancos
   write: /01 nombre_struc.            ' imprime el literal -abc-
```

Romper (split) el contenido de una variable

```
report yxxxxxxx."
data: a(40) type c value 'PRIMERO SEGUNDO TERCERO CUARTO QUINTO',
      b(10) type c,
      c(20) type c,
      d(30) type c.
start-of-selection.
split a at space into b c d.
write:   /01 b,                        ' b = PRIMERO
         /01 c,                        ' c = SEGUNDO
         /01 d.                  ' d = TERCERO; CUARTO; QUINTO
```

Seno, coseno y tangente

```
form seno_coseno_tangente.
  data: ind2(3)     type n value 0,
        ind3        type f value 0,
```

```
        pi          type f value '3.14159',
        seno        type f value 0,
        coseno      type f value 0,
        tangente    type f value 0.
  do 360 times.
    ind2 = ind2 + 1.
    ind3 = ( ind2 * 2 * pi ) / 360.    ‘ Convertir a radianes
    seno = sin( ind3 ).
    coseno = cos( ind3 ).
    tangente = seno / coseno.
    write: / ind2,'--> ',
             seno color 1,
             coseno color 2,
             tangente color 3.
  enddo.
endform.                    "seno_coseno_tangente
```

Shiftar el contenido de un campo

```
report yxxxxxxx.
data: campo(20) type c.
start-of-selection.
   campo = ' .+ .1 .+ .2'.
  do 20 times.
     shift campo.                        ' Desplaza string hacia la
Izquierda
    write: /01 campo.
  enddo.
  campo = ' .+ .1 .+ .2'.
  do 20 times.
     shift campo right.                  ' Desplaza string hacia la
Derecha
    write: /01 campo.
  enddo.
```

Tratar hexadecimales

```
program ZJCBxxx.
data:  oper1(2)  type X,     "(2) ==> de 0000 a FFFF
       oper2(2)  type X,
       total(2)  type X.
parameters: num1(4)   type N,
            num2(4)   type N.
start-of-selection.
  oper1 = num1.
  oper2 = num2.
  total = oper1 + oper2.
```

```
  write: /01 total.
```

Truncar un valor

```
report yxxxxxxx.
data: valor1 type f value '123.45',
      valor2 type f.
compute valor2 = trunc( valor1 ).
write: /01 valor1,                    ‘ Imprime; 1, 2.345E+16
           valor2.                    ‘ Imprime; 1, 2.3E+16
```

Utilizar offsets

```
report yxxxxxxx."
data: a(10) type c value 'abcdefghij'."
data: b(10) type c."
data:  offset type i value 1,"
       longi  type i value 2."
start-of-selection."
  move a+offset(longi) to b."
  write: /01 a,                       'Imprime abcdefghij
         /01 b.                       'Imprime bc
```

Utilizar sentencia rollback

```
program yxxxxxxx.
tables: t9jcb.
start-of-selection.
select single for update * from t9jcb where codigo eq 'ZZZZZ'.
if sy-subrc eq 0.
  move 'primera modificaci¢n' to t9jcb-descr1.
  update t9jcb.
  commit work.
  move 'segunda modificaci¢n' to t9jcb-descr1.
  update t9jcb.
    rollback  work.               '  Deshace  solamente  la  segunda
modificación
endif.
```

Utilizar sentencia hide

```
report yxxxxxxx"
       no standard page heading.
```

```
data: campo(3) type c.
start-of-selection.
  write: /01 'primera linea'.
  campo = '111'.
   hide campo.              ' asociamos el valor de campo (en este
caso 111)
                         ' a la linea escrita anteriormente.
  write: /01 'segunda linea'.
  campo = '222'."
   hide campo.              ' asociamos el valor de campo (en este
caso 222)
                         ' a la linea escrita anteriormente.
at line-selection.
   write: /01 campo.      ' campo valdrá 111 si se ha seleccionado
la
                            ' primera linea y valdrá 222 si se ha
seleccionado
                         ' la segunda línea
```

Visualizar lista de usuarios conectados

```
report zxxxxxxx
       line-size 120
       no standard page heading.
data: begin of t1 occurs 1.
        include structure uinfo.
data: end of t1.
data: w_conects   type i.
start-of-selection.
  refresh t1.
  CALL FUNCTION 'THUSRINFO'
    TABLES
      USR_TABL = T1
    EXCEPTIONS
      OTHERS   = 1.
  describe table t1 lines w_conects.
  sort t1 by mandt bname.
  format color col_heading.
  format intensified off.
  uline.
  loop at t1.
    write: /01 sy-vline,
               t1-mandt       intensified on,
               t1-bname,
               t1-tcode       intensified on,
               t1-zeit,
               t1-extmodi     intensified on,
               t1-hostadr,
               t1-term        intensified on,
```

```
              t1-tid,
           at sy-linsz sy-vline.
  endloop.
  uline.
  format color col_positive.
  write: /01 sy-vline,
             'usuarios conectados al sistema',
            w_conects color col_normal,
          at sy-linsz sy-vline.
  uline.
```

Visualizar tabla de bd

```
program zxxxxxx.
parameters: NomTbl  LIKE DD02V-TABNAME.
start-of-selection.
  CALL FUNCTION 'RS_DD_TABL_EDIT'
    EXPORTING
      OBJNAME              = NomTbl
      FNAME                = ' '
      EDIT_MODE            = 'S'           'S=Visualizar, E=Editar
      POPUP                = ' '
    EXCEPTIONS
      OBJECT_NOT_FOUND     = 1
      OBJECT_NOT_SPECIFIED = 2
      PERMISSION_FAILURE   = 3
      NOT_EXECUTED         = 4
      OTHERS               = 5.
```

2. Programas de ejemplo

AutoRefresh

```
* ********************************************************** *
*  Autor: Jordi Calopa Bosch (JCB)                           *
*                                                            *
* Este ejemplo muestra como realizar un programa en el cual la *
* pantalla de salida se refresque automáticamente cada segundo *
*                                                            *
* ********************************************************** *
PROGRAM Zxxxxxxx
        LINE-SIZE 250
        NO STANDARD PAGE HEADING.

CONSTANTS: COMANDO(4) TYPE C VALUE 'XXXX'.

*---------------------------------------------------------------------*"
*       START-OF-SELECTION                                            *"
*---------------------------------------------------------------------*"
START-OF-SELECTION.

  PERFORM PROCESO_PRINCIPAL.
  SET USER-COMMAND COMANDO.

*---------------------------------------------------------------------*"
*       FORM PROCESO_PRINCIPAL                                        *"
*---------------------------------------------------------------------*"
FORM PROCESO_PRINCIPAL.

  Skip 10.
  WRITE: /30 SY-ULINE(23).
  WRITE:  50 SY-ULINE(21).
  WRITE: /30 SY-VLINE, 'Host   :', SY-HOST,  SPACE, SY-VLINE.
  WRITE:  50 SY-VLINE, 'Fecha:', SY-DATUM INTENSIFIED OFF,
                                          SY-VLINE.
  WRITE: /30 SY-VLINE, 'Entorno:', SY-SYSID, SPACE, SY-VLINE.
  WRITE:  50 SY-ULINE(21).
  WRITE: /30 SY-VLINE, 'S.O.   :', SY-OPSYS, SY-VLINE.
  WRITE:  50 SY-VLINE, 'Hora: ', SY-UZEIT INTENSIFIED OFF,
                                       ' ', SY-VLINE.
  WRITE: /30 SY-ULINE(23).
  WRITE:  50 SY-ULINE(21).
  SKIP."

ENDFORM.
*---------------------------------------------------------------------*"
*       AT USER-COMMAND                                               *"
*---------------------------------------------------------------------*"
AT USER-COMMAND.

  SY -LSIND = 0.

   IF SY-UCOMM EQ COMANDO.
     PERFORM PROCESO_PRINCIPAL.
```

```
     CALL FUNCTION 'Y_DUMMY'
      STARTING New TASK 'IF'
      PERFORMING REFRESCAR ON END OF TASK.
   ENDIF.
*---------------------------------------------------------------------*"
*       FORM REFRESCAR                                                *"
*---------------------------------------------------------------------*"
FORM REFRESCAR USING TASKNAME.
  DATA: INT_SECONDS TYPE I VALUE 1.

  CALL 'ALERTS' ID 'ADMODE'         FIELD 50
                ID 'STORAGE_OPCODE' FIELD 'SLEEP'
                ID 'TIME'           FIELD INT_SECONDS.

  SET USER-COMMAND COMANDO.
ENDFORM.
```

Backup

```
* ************************************************************ *
* Autor ........................ Jordi Calopa Bosch (JCB)     *
* Ultima actualización ............ 09.10.2002                 *
* Entorno de desarrollo ........... SAP/R3 (31I) bajo Unix     *
* Lenguaje de programación ........ ABAP/4                     *
*                                                                *
*  Backup Automático de Programas del User activo. Este programa *
*  revisa la tabla TRDIR y cuando encuentra un programa generado *
*  y/o modificado por el usuario activo, lo vuelca a PC,         *
*  añadiéndole la extensión TXT                                  *
*                                                                *
* ************************************************************ *
program zxxxxxxx
        line-size 120
        no standard page heading.

include zjcb_include.

* *** Declaración de tablas y Vistas de Base de Datos            *** *
TABLES: TRDIR.

* *** Declaración de tablas dinámicas de memoria interna         *** *
DATA: BEGIN OF t1 OCCURS 5000,          " Tabla de volcado
         reg(255) type c,
      end of t1.

DATA: BEGIN OF t2 OCCURS 10000,         " Tabla directorio
         name like trdir-name,
         clas like trdir-clas,
         cnam like trdir-cnam,
         cdat like trdir-cdat,
         unam like trdir-unam,
         udat like trdir-udat,
         secu like trdir-secu,
         appl like trdir-appl,
      end of t2.

* *** Declaración de variables globales                          *** *
data: wdir       type string,
      tipo(10)   type c value 'ASC',
      founds     type i value 0,
      longi      type i value 0,
      campo      type c,
      numero(4)  type n.

* *** Definición de los Criterios de Selección                   *** *
SELECTION-SCREEN: BEGIN OF BLOCK BL1 WITH FRAME TITLE TEXT-001.
SELECTION-SCREEN: SKIP.
PARAMETERS: DIRPC(25) TYPE C OBLIGATORY.
SELECTION-SCREEN: SKIP.
SELECT-OPTIONS: WMODIF FOR sy-datlo
                           NO-EXTENSION.
SELECTION-SCREEN: SKIP.
SELECTION-SCREEN: END OF BLOCK BL1.
```

```
INITIALIZATION.
  DIRPC = 'c:\temp\'.
  WMODIF-SIGN = 'I'.
  WMODIF-OPTION = 'EQ'.
  WMODIF-LOW = sy-datlo - 1.
  WMODIF-HIGH = sy-datlo.
  APPEND WMODIF.

START-OF-SELECTION.
  clear t2.
  refresh t2.

* Selección de Programas
  SELECT NAME CLAS CNAM CDAT UNAM UDAT SECU APPL
         INTO TABLE t2
         FROM TRDIR
         WHERE ( CNAM EQ sy-uname OR UNAM EQ sy-uname ) AND
               UDAT IN WMODIF                           and
               ( name like 'Y%'      or
                 name like 'Z%'      or
                 name like 'SAPMY%'  or
                 name like 'MY%' )
         ORDER BY NAME.

* *** Proceso de la tabla interna con los programas seleccionados
  FORMAT INTENSIFIED OFF.
  FORMAT COLOR COL_NORMAL.
  ULINE.

  loop at t2.

    numero = sy-tabix.
    CALL FUNCTION 'SAPGUI_PROGRESS_INDICATOR'
      EXPORTING
        PERCENTAGE = sy-tabix
        TEXT       = Numero.

    CHECK t2+00(01) GE 'A' AND t2+00(01) LE 'Z'.
    LONGI = STRLEN( DIRPC ).
    LONGI = LONGI - 1.
    MOVE DIRPC+LONGI(01) TO CAMPO.
    IF CAMPO EQ '\'.
      CONCATENATE DIRPC t2-NAME '.txt' INTO WDIR.
    ELSE.
      CONCATENATE DIRPC '\' t2-NAME '.txt' INTO WDIR.
    ENDIF.

    read report t2-name into t1.

    CALL FUNCTION 'GUI_DOWNLOAD'
      EXPORTING
        filename                = wdir
        FILETYPE                = tipo
      TABLES
        data_tab                = t1
      EXCEPTIONS
        FILE_WRITE_ERROR        = 1
        NO_BATCH                = 2
        GUI_REFUSE_FILETRANSFER = 3
        INVALID_TYPE            = 4
        NO_AUTHORITY            = 5
```

```
        UNKNOWN_ERROR           = 6
        HEADER_NOT_ALLOWED      = 7
        SEPARATOR_NOT_ALLOWED   = 8
        FILESIZE_NOT_ALLOWED    = 9
        HEADER_TOO_LONG         = 10
        DP_ERROR_CREATE         = 11
        DP_ERROR_SEND           = 12
        DP_ERROR_WRITE          = 13
        UNKNOWN_DP_ERROR        = 14
        ACCESS_DENIED           = 15
        DP_OUT_OF_MEMORY        = 16
        DISK_FULL               = 17
        DP_TIMEOUT              = 18
        FILE_NOT_FOUND          = 19
        DATAPROVIDER_EXCEPTION  = 20
        CONTROL_FLUSH_ERROR     = 21
        OTHERS                  = 22.

    CASE SY-SUBRC.

      WHEN '0'.
        WRITE: /01 SY-VLINE,
                   'Volcado del fichero'(002),
                   WDIR COLOR COL_POSITIVE,
                AT SY-LINSZ SY-VLINE.

      WHEN OTHERS.
        WRITE: /01 SY-VLINE,
                   'ERROR AL VOLCAR ...'(003)   COLOR COL_NEGATIVE,
                   WDIR                      COLOR COL_NEGATIVE,
                AT SY-LINSZ SY-VLINE.

    ENDCASE.

  ENDLOOP.

  ULINE.

END-OF-SELECTION.

* *** Cabeceras del listado                                          *** *
top-of-page.

  WRITE: /01 '  Backup Automático de Programas del User:',
             SY-UNAME.
```

Copia de programas de forma dinámica (desde un report ABAP)

Existen algunas funciones externas que permiten copiar programas, desde dentro de un programa desarrollado en Abap/4. En el ejemplo que mostramos seguidamente efectuámos una copia de un report, con toda la documentación, dynpros, includes... sin que aparezca ninguna pantalla de dialogo ni selección, asignándolo a la clase de desarrollo '$TMP'. Para ello utilizamos la función RS_COPY_PROGRAM, que también puede ser lanzada en la modalidad de dialogo.

```
* Ejemplo - Batch"

* ********************************************************************* *"
* Descripción: Este programna sirve para realizar copias de otros      *"
*              programas, de forma din mica. En nuestro ejemplo        *"
*              copiamos el repost YSUXTES2 sobre el YSUXTES3, que      *"
*              previamente no existe.                                  *"
* ********************************************************************* *"
REPORT Yxxxxxxx.

DATA: NOM_DESDE LIKE SY-REPID,
      NOM_HASTA LIKE SY-REPID.

Move 'YSUXTES2' TO NOM_DESDE.
Move 'YSUXTES3' TO NOM_HASTA.

CALL FUNCTION 'RS_COPY_PROGRAM'
     exporting
*         CORRNUMBER          = '          '
          DEVCLASS            = '$TMP'
          program = nom_hasta
          source_program = nom_desde
          SUPPRESS_CHECKS     = 'X'
          SUPPRESS_COMMIT     = 'X'
          SUPPRESS_NORM_CHECK = 'X'
          SUPPRESS_POPUP      = 'X'
          SUPPRESS_SCREEN     = 'X'
          WITH_CUA            = 'X'
          WITH_DOCUMENTATION  = 'X'
          WITH_DYNPRO         = 'X'
          WITH_INCLUDES       = 'X'
          WITH_TEXTPOOL       = 'X'
          WITH_VARIANTS       = 'X'
*    IMPORTING
*         CORRNUMBER          =
*         DEVCLASS            =
*         PROGRAM             =
     exceptions
          enqueue_lock = 1
          object_not_found = 2
          permission_failure = 3
          reject_copy = 4
          others = 5#

WRITE: /01 SY-SUBRC.
```

Desbloquear Código de Usuario

```
* ********************************************************************* *"
*  Este programa permite desbloquear un código de usuario que haya     *"
*  quedado bloqueado (p.e. cuando nos equivocamos 3 veces de password) *"
* ********************************************************************* *"
REPORT ZXXXXXXX.

TABLES: usr02.

PARAMETERS: usuario  LIKE sy-uname  DEFAULT sy-uname.

START-OF-SELECTION.

  FORMAT INTENSIFIED OFF.
  FORMAT COLOR COL_NORMAL.
  ULINE.

  SELECT SINGLE FOR UPDATE * FROM usr02
                           WHERE bname EQ usuario.

  IF sy-subrc EQ 0.

    usr02-locnt = 0#
    usr02-uflag = 0#
   UPDATE usr02.

    IF sy-subrc EQ 0.
* *** Desbloqueado ***
      FORMAT COLOR COL_POSITIVE.
      WRITE: /01 sy-vline,
                 'Usuario Desbloqueado',
                 usuario   COLOR COL_TOTAL,
              AT sy-linsz sy-vline.
    ELSE.
* *** No Desbloqueado ***
      FORMAT COLOR COL_NEGATIVE.
      WRITE: /01 sy-vline,
                'NO ha sido posible desbloquear este usuario',
                  usuario   COLOR COL_TOTAL,
            AT sy-linsz sy-vline.
    ENDIF.
  ELSE.
* *** No Encontrado ***
    FORMAT COLOR COL_NEGATIVE.
    WRITE: /01 sy-vline,
               'Usuario NO encontrado en USR02',
                 usuario   COLOR COL_TOTAL,
            AT sy-linsz sy-vline.
  ENDIF.

END-OF-SELECTION.
,  ULINE.
```

Manejador de programas

```
* ************************************************************
*  Autor ........................... Jordi Calopa Bosch (JCB)  *
*  Ultima actualización ............ 16.01.2001                *
*  Entorno de desarrollo ........... SAP/R3 (31H) bajo Unix    *
*  Lenguaje de programación ........ ABAP/4                    *
*                                                              *
*    Este programa permite Visualizar, Renombrar, Copiar,      *
*    Borrar, Bloquear, Desbloquear, Regenerar, Imprimir        *
*    Volcar a PC y/o Ejecutar (en local) un report             *
*    existente en la tabla de programas TRDIR.                 *
*                                                              *
* ************************************************************ *
REPORT Yxxxxxxx
       line-size 120
       NO STANDARD PAGE HEADING.

INCLUDE <LIST>.
INCLUDE YSU0000.

TABLES: TRDIR, DD02L, DD03L, USR03.

DATA: NOMPROG          LIKE SY-REPID,
      NOMPNEW          LIKE SY-REPID,
      NUMLINES         TYPE I,
      RESP(01)         TYPE C,
      NAMETAB(10)      TYPE C,
      FOUNFLD(01)      TYPE C VALUE 'F',
      FIELD_CLASIF(30) TYPE C,
      RETCODE          LIKE SY-SUBRC.

DATA: BEGIN OF T0 OCCURS 5000,
         C(72) TYPE C,
      END OF T0.

DATA: BEGIN OF T1 OCCURS 1,
      TABNAME   LIKE DD03L-TABNAME  ,
      FIELDNAME LIKE DD03L-FIELDNAME,
      END OF T1.

DATA: BEGIN OF T2 OCCURS 1,
         r(72) type c,
END OF T2.

DATA: BEGIN OF T3 OCCURS 50.
        INCLUDE STRUCTURE TEXTPOOL.
DATA: END OF T3.

*---------------------------------------------------------------------*"
*       START-OF-SELECTION                                            *"
*---------------------------------------------------------------------*"
START-OF-SELECTION."

 SY-TITLE = 'Program Manager'.

  PERFORM PEDIR_NOMBRE_PGM.
```

```
  IF RETCODE NE 0.
    EXIT.
  ENDIF.
  IF NOMPROG EQ SY-REPID.
    EXIT.
  ENDIF.

 READ REPORT NOMPROG INTO T0.
  IF SY-SUBRC NE 0.
    EXIT.
  ENDIF.
 DESCRIBE TABLE T0 LINES NUMLINES.

  PERFORM TRATAMIENTO_GRAL.
  IF RETCODE NE 0.
    EXIT.
  ENDIF.

END-OF-SELECTION.

*---------------------------------------------------------------------*"
*       FORM PEDIR_NOMBRE_PGM                                         *"
*---------------------------------------------------------------------*"
FORM PEDIR_NOMBRE_PGM.

  DATA: VALOR_INP LIKE NOMPROG,
        VALOR_OUT LIKE NOMPROG,
         VALOR_RES.

  CLEAR: NOMPROG.

CALL FUNCTION 'POPUP_TO_GET_VALUE'
       EXPORTING
            FIELDNAME           = 'NAME'
            TABNAME             = 'TRDIR'
            TITEL               = 'Indique el Nombre del'
            VALUEIN = VALOR_INP
       IMPORTING
            ANSWER = VALOR_RES
            VALUEOUT = VALOR_OUT
       EXCEPTIONS
            FIELDNAME_NOT_FOUND = 1
            OTHERS = 2.

  IF VALOR_OUT IS INITIAL.
    RETCODE = 4.
  ELSE.
    SELECT SINGLE * FROM TRDIR
           WHERE NAME EQ VALOR_OUT.
    RETCODE = SY-SUBRC.
    IF RETCODE EQ 0.
      NOMPROG = VALOR_OUT.
    ENDIF.
  ENDIF.

ENDFORM.

*---------------------------------------------------------------------*"
*       FORM TRATAMIENTO_GRAL                                         *"
*---------------------------------------------------------------------*"
FORM TRATAMIENTO_GRAL.
```

```
  REFRESH T3.
  CLEAR: T3.
  READ TEXTPOOL NOMPROG INTO T3 LANGUAGE 'S'.
  READ TABLE T3 WITH KEY 'R'.

  FORMAT INTENSIFIED OFF.
  FORMAT COLOR COL_HEADING.
ULINE.

  WRITE: /01 SY-VLINE,
         120 sy-vline.
  WRITE: /01 SY-VLINE,
             NOMPROG COLOR COL_NORMAL,
             T3-ENTRY(70),
         120 sy-vline.
  WRITE: /01 SY-VLINE,
         120 SY-VLINE,
         /01 SY-VLINE,
             'Grupo autorizaciones..', TRDIR-SECU    COLOR COL_NORMAL,
          40 'Autor     ..', TRDIR-CNAM    COLOR COL_NORMAL,
          80 'Fecha generación  ', TRDIR-CDAT    COLOR COL_NORMAL,
         120 SY-VLINE,
         /01 SY-VLINE,
             'Versión     ', TRDIR-VERN    COLOR COL_NORMAL,
          40 'Modificador   ..', TRDIR-UNAM    COLOR COL_NORMAL
                                               INTENSIFIED OFF,
          80 'Fecha modificación .', TRDIR-UDAT    COLOR COL_NORMAL,
         120 SY-VLINE,
         /01 SY-VLINE,
             'Protección fuente  .', TRDIR-SQLX   COLOR COL_NORMAL,
          40 'Protección edición  ', TRDIR-EDTX   COLOR COL_NORMAL,
          80 'Mayusculas/minusculas.', TRDIR-VARCL  COLOR COL_NORMAL,
         120 SY-VLINE,
         /01 SY-VLINE,
             'Generado automático ', TRDIR-OCCURS  COLOR COL_NORMAL,
          40 'Tipo de programa  ', TRDIR-SUBC    COLOR COL_NORMAL,
          80 'Base de Datos Lógica..', TRDIR-DBNA    COLOR COL_NORMAL,
                                       TRDIR-DBAPL   COLOR COL_NORMAL,
         120 SY-VLINE,
         /01 SY-VLINE,
             'Nivel     ..', TRDIR-LEVL    COLOR COL_NORMAL,
          40 'Status     .', TRDIR-RSTAT   COLOR COL_NORMAL,
          80 'Mandante    ..', TRDIR-RMAND   COLOR COL_NORMAL,
         120 SY-VLINE,
         /01 SY-VLINE,
             'Idioma maestro  ..', TRDIR-RLOAD   COLOR COL_NORMAL,
          40 'Coma fija    .', TRDIR-FIXPT   COLOR COL_NORMAL,
          80 'Solo con variante ..', TRDIR-SSET    COLOR COL_NORMAL,
         120 SY-VLINE.
  WRITE: /01 SY-VLINE,
            'Cantidad Líneas   ', NUMLINES COLOR COL_NORMAL,
         120 sy-vline.
  WRITE: /01 SY-VLINE,
         120 sy-vline.

  WRITE: /01 SY-VLINE,
          10 ICON_DISPLAY        AS ICON,
          20 ICON_RENAME         AS ICON,
          30 ICON_COPY_OBJECT    AS ICON,
          40 ICON_DELETE         AS ICON,
```

```
          50 ICON_LOCKED         AS ICON,
          60 ICON_UNLOCKED       AS ICON,
          70 ICON_GENERATE       AS ICON,
          80 ICON_PRINT          AS ICON,
          90 ICON_MOVE           AS ICON,
         100 ICON_EXECUTE_OBJECT AS ICON,
         120 SY-VLINE.

  WRITE: /01 SY-VLINE,
         120 SY-VLINE.
  ULINE.

ENDFORM.

*---------------------------------------------------------------------*"
*       AT LINE-SELECTION                                             *"
*---------------------------------------------------------------------*"
AT LINE-SELECTION.

  SY-LSIND = 0.

  IF SY-CUROW EQ 13.
    IF SY-CUCOL EQ 11 OR SY-CUCOL EQ 12.
      PERFORM DISPLAY_PGM.
    ELSEIF SY-CUCOL EQ 21 OR SY-CUCOL EQ 22.
      PERFORM RENAME_PGM.
      PERFORM TRATAMIENTO_GRAL.
    ELSEIF SY-CUCOL EQ 31 OR SY-CUCOL EQ 32.
      PERFORM COPY_PGM.
      PERFORM TRATAMIENTO_GRAL.
    ELSEIF SY-CUCOL EQ 41 OR SY-CUCOL EQ 42.
      PERFORM DELETE_PGM.
      IF RETCODE = 0.
        LEAVE TO SCREEN 0.
      ENDIF.
    ELSEIF SY-CUCOL EQ 51 OR SY-CUCOL EQ 52.
      PERFORM LOCKED_PGM.
      PERFORM TRATAMIENTO_GRAL.
    ELSEIF SY-CUCOL EQ 61 OR SY-CUCOL EQ 62.
      PERFORM UNLOCKED_PGM.
      PERFORM TRATAMIENTO_GRAL.
    ELSEIF SY-CUCOL EQ 71 OR SY-CUCOL EQ 72.
      PERFORM GENERATE_PGM.
    ELSEIF SY-CUCOL EQ 81 OR SY-CUCOL EQ 82.
      PERFORM PRINT_PGM.
    ELSEIF SY-CUCOL EQ 91 OR SY-CUCOL EQ 92.
      PERFORM VOLCADO_PGM.
    ELSEIF SY-CUCOL EQ 101 OR SY-CUCOL EQ 102.
      SUBMIT (NOMPROG) AND RETURN VIA SELECTION-SCREEN.
    ENDIF.

  ENDIF.
*---------------------------------------------------------------------*"
*       FORM DISPLAY_PGM                                              *"
*---------------------------------------------------------------------*"
FORM DISPLAY_PGM.

  CALL FUNCTION 'RS_SOURCE_STRUCTURE'
       EXPORTING
            Source = NOMPROG
       EXCEPTIONS
```

```
            OTHERS = 1.

ENDFORM.
*---------------------------------------------------------------------*"
*       FORM RENAME_PGM                                               *"
*---------------------------------------------------------------------*"
FORM RENAME_PGM.

  CALL FUNCTION 'RS_RENAME_PROGRAM'
       EXPORTING
            SOURCE_PROGRAM = NOMPROG
       IMPORTING
            PROGRAM = NOMPNEW
       EXCEPTIONS
            ENQUEUE_LOCK = 1
            OBJECT_NOT_FOUND = 2
            PERMISSION_FAILURE = 3
            REJECT_COPY = 4
            REJECT_DELETION = 5
            OTHERS = 6.

 IF SY-SUBRC EQ 0.
    NOMPROG = NOMPNEW.
  ENDIF.

ENDFORM.
*---------------------------------------------------------------------*"
*       FORM COPY_PGM                                                 *"
*---------------------------------------------------------------------*"
FORM COPY_PGM.

  CALL FUNCTION 'RS_COPY_PROGRAM'
       EXPORTING
           SOURCE_PROGRAM = NOMPROG
       IMPORTING
            PROGRAM = NOMPNEW"
       EXCEPTIONS
            ENQUEUE_LOCK = 1
            OBJECT_NOT_FOUND = 2
            PERMISSION_FAILURE = 3
            REJECT_COPY = 4
            OTHERS = 5.

  IF SY-SUBRC EQ 0.
    NOMPROG = NOMPNEW.
  ENDIF.

ENDFORM.
*---------------------------------------------------------------------*"
*       FORM DELETE_PGM                                               *"
*---------------------------------------------------------------------*"
FORM DELETE_PGM.

  CALL FUNCTION 'RS_DELETE_PROGRAM'
       EXPORTING
           PROGRAM = NOMPROG
       EXCEPTIONS
            ENQUEUE_LOCK = 1
            OBJECT_NOT_FOUND = 2
            PERMISSION_FAILURE = 3
            REJECT_DELETION = 4
```

```
            OTHERS = 5.

  RETCODE = SY-SUBRC.

ENDFORM.
*---------------------------------------------------------------------*"
*       FORM LOCKED_PGM                                               *"
*---------------------------------------------------------------------*"
FORM LOCKED_PGM.

  SELECT SINGLE * FROM TRDIR
         WHERE NAME EQ NOMPROG.

  IF SY-SUBRC EQ 0.
    TRDIR-EDTX = 'X'.
    UPDATE TRDIR.
    COMMIT WORK.
  ENDIF.

ENDFORM.
*---------------------------------------------------------------------*"
*       FORM UNLOCKED_PGM                                             *"
*---------------------------------------------------------------------*"
FORM UNLOCKED_PGM.

  SELECT SINGLE * FROM TRDIR
         WHERE NAME EQ NOMPROG.

  IF SY-SUBRC EQ 0.
    TRDIR-EDTX = SPACE.
    UPDATE TRDIR.
    COMMIT WORK.
  ENDIF.

ENDFORM.
*---------------------------------------------------------------------*"
*       FORM GENERATE_PGM                                             *"
*---------------------------------------------------------------------*"
FORM GENERATE_PGM.

  GENERATE REPORT NOMPROG.

ENDFORM.
*---------------------------------------------------------------------*"
*       FORM PRINT_PGM                                                *"
*---------------------------------------------------------------------*"
FORM PRINT_PGM.

  CALL FUNCTION 'RS_PROGRAM_PRINT'
       EXPORTING
            DIALOG            = 'X'
            PROGRAMM = NOMPROG
       EXCEPTIONS
            CANCELED = 1
            OPEN_FORM_ERROR = 2
            PROGRAM_NOT_FOUND = 3
            NO_AUTHORITY = 4
            OTHERS = 5.

ENDFORM.
*---------------------------------------------------------------------*"
```

```
*       FORM VOLCADO_PGM                                              *"
*-------------------------------------------------------------------*"
FORM VOLCADO_PGM.

  DATA: PATHNAME(128) TYPE C.

  CONCATENATE 'c:\datos\' NOMPROG INTO PATHNAME.

  CALL FUNCTION 'WS_DOWNLOAD'
       EXPORTING
            filename = PATHNAME
            FILETYPE            = 'ASC'
       TABLES
            DATA_TAB = T0
       EXCEPTIONS
            FILE_OPEN_ERROR = 1
            FILE_WRITE_ERROR = 2
            INVALID_FILESIZE = 3
            INVALID_TABLE_WIDTH = 4
            INVALID_TYPE = 5
            NO_BATCH = 6
            UNKNOWN_ERROR = 7
            OTHERS = 8.

  IF SY-SUBRC EQ 0.
    MESSAGE ID 38 TYPE 'I' NUMBER 016
        WITH 'Programa volcado correctamente a PC'.
  ENDIF.

ENDFORM.
*-------------------------------------------------------------------*"
*       FORM CABECERA_REPORT                                          *"
*-------------------------------------------------------------------*"
FORM CABECERA_REPORT.
  DATA: NOMBRE(40).
  SELECT SINGLE * FROM USR03 WHERE BNAME EQ SY-UNAME.
  IF SY-SUBRC EQ 0.
    CONCATENATE USR03-NAME1 USR03-NAME2 '(' SY-UNAME ')'
                INTO NOMBRE SEPARATED BY ' '.
  ELSE.
    NOMBRE = SY-UNAME.
  ENDIF.
  REFRESH T2.
  PERFORM R_BLANCO.
  T2-R+00 = '* Autor:    '.
  T2-R+14 = NOMBRE.                              APPEND T2.
  T2-R+00 = '* Fecha:    '.
  T2-R+14 = SY-DATUM.                            APPEND T2.
  T2-R+00 = '* Hora:     '.
  T2-R+14 = SY-UZEIT.                            APPEND T2.
  T2-R+00 = '* Host:     '.
  T2-R+14 = SY-HOST.                             APPEND T2.
  T2-R+00 = '* Sistema:    '.
  T2-R+14 = SY-OPSYS.                            APPEND T2.
  perform r_blanco.
  t2-r+00 = 'REPORT'.
  t2-r+08 = nomprog.                             append t2.
  T2-R = '        line-size 94'.                 APPEND T2.
  T2-R = '        no standard page heading.'.    APPEND T2.
  T2-R = '                                 '.    APPEND T2.
ENDFORM.
```

```
*---------------------------------------------------------------------*"
*       FORM R_BLANCO                                                 *"
*---------------------------------------------------------------------*"
form r_blanco.
  T2-R = '* ------------------------------------------------------- *'.
  APPEND T2.
ENDFORM.
*---------------------------------------------------------------------*"
*       FORM INSERTAR_PGM                                             *"
*---------------------------------------------------------------------*"
FORM INSERTAR_PGM.

  CALL FUNCTION 'RKD_INSERT_REPORT'
       EXPORTING
            COMMIT         = 'X'
            ENQUEUE_FLAG   = 'X'
            REPORT_NAME = NOMPROG
            REPORT_NAMEH = NOMPROG
            REPORT_NAMEL = NOMPROG
            TADIR_DEVCLASS = '$TMP'
            TEXT           = 'Nuevo programa'
            TRDIR_APPL     = 'S'
            TRDIR_CLAS     = '$TMP'
            TRDIR_DBAPL    = 'S'
            TRDIR_DBNA     = 'D$'
            TRDIR_FIXPT    = 'X'
            TRDIR_RSTAT    = 'T'
            TRDIR_SQLX     = 'R'
            TRDIR_SUBC     = '1'
       TABLES
            CODE_TAB = T2
       EXCEPTIONS
            NO_FREE_NUMBER = 1
            SYNTAX_CHECK = 2
            TRDIR_LOCKED = 3
            OTHERS = 4.

 IF SY-SUBRC EQ 0.
    SELECT SINGLE * FROM TRDIR
           WHERE NAME EQ NOMPROG.
    IF SY-SUBRC EQ 0.
      TRDIR-RLOAD = 'S'.
      TRDIR-EDTX = 'X'.
      TRDIR-SECU = 'YSGNI'.
      UPDATE TRDIR.
      COMMIT WORK.
    ENDIF.
  ENDIF.

 ENDFORM.
```

Lanzadera

```
* ********************************************************************
*  Autor ........................... Jordi Calopa Bosch (JCB)        *
*  Ultima actualización ............ 16.02.2001                      *
*  Entorno de desarrollo ........... SAP/R3 (31H) bajo Unix          *
*  Lenguaje de programación ........ ABAP/4                          *
*                                                                    *
*  Descripción: Esta utilidad sirve para submitir programas Abap     *
*               residentes en la work-station.                       *
*                                                                    *
* ******************************************************************** *

REPORT Yxxxxxxx.

INCLUDE YSU0000.

* *** Declaración de Variables Globales                               *** *
DATA: RESPUESTA(01) TYPE C,
      RETCODE       LIKE SY-SUBRC,
      TXT120(120)   TYPE C.

* *** Declaración de Constantes Globales                            *** *
CONSTANTS: MARCA(01)   TYPE C VALUE 'X',
           SKELETON(8) TYPE C VALUE 'YZZZZZZZ'.

* *** Definición de tablas dinámicas de memoria interna             *** *
DATA: BEGIN OF TBLPGM OCCURS 5000,
       C(72),
      END OF TBLPGM.

* *** Definición de Criterios de Selección                           *** *
SELECTION-SCREEN: BEGIN OF BLOCK 1 WITH FRAME TITLE TEXT-001.
SELECTION-SCREEN: SKIP.
PARAMETERS: PROGRAMA LIKE SY-REPID.
SELECTION-SCREEN: SKIP.
SELECTION-SCREEN: BEGIN OF BLOCK 2 WITH FRAME.
PARAMETERS: VISUAL RADIOBUTTON GROUP G1,
            EDITAR RADIOBUTTON GROUP G1,
            LANZAR RADIOBUTTON GROUP G1.
SELECTION-SCREEN: END OF BLOCK 2.
SELECTION-SCREEN: BEGIN OF BLOCK 3 WITH FRAME.
SELECTION-SCREEN: SKIP.
PARAMETERS: RESTORE AS CHECKBOX.
SELECTION-SCREEN: SKIP.
PARAMETERS: RESTPAT(128) TYPE C DEFAULT 'C:\DATOS\'.
SELECTION-SCREEN: END OF BLOCK 3.
SELECTION-SCREEN: END OF BLOCK 1.

*---------------------------------------------------------------------*"
*       AT SELECTION-SCREEN                                           *"
*---------------------------------------------------------------------*"
AT SELECTION-SCREEN.

  IF PROGRAMA IS INITIAL.
    PROGRAMA = SKELETON.
  ENDIF.
```

```
  IF LANZAR EQ MARCA.
    PROGRAMA = SKELETON.
  ENDIF.

*---------------------------------------------------------------------*"
*       START-OF-SELECTION                                            *"
*---------------------------------------------------------------------*"
START-OF-SELECTION.

* *** EL SIGUIENTE IF NO DEBE SER ELIMINADO JAMAS ***
  IF PROGRAMA EQ 'SAPMSYST'.
    MESSAGE ID 38 TYPE 'I' NUMBER 16
            WITH 'El programa Informado NO es correcto'.
    STOP.
  ENDIF.
* ***

  IF EDITAR EQ MARCA.
    PERFORM RUTINA_EDITAR.
  ELSEIF LANZAR EQ MARCA.
    PERFORM RUTINA_LANZAR.
  ELSE.
    PERFORM RUTINA_VISUALIZAR.
  ENDIF.

END-OF-SELECTION.

*---------------------------------------------------------------------*"
*       FORM RUTINA_VISUALIZAR                                        *"
*---------------------------------------------------------------------*"
FORM RUTINA_VISUALIZAR.

  IF RESTORE = MARCA.
    PERFORM RESTAURAR_DESDE_PC.
  ELSE.
    READ REPORT PROGRAMA INTO TBLPGM.
    RETCODE = SY-SUBRC.
  ENDIF.

  IF RETCODE EQ 0.
    EDITOR-CALL FOR TBLPGM DISPLAY-MODE.
  ENDIF.

ENDFORM.

*---------------------------------------------------------------------*"
*       FORM RUTINA_EDITAR                                            *"
*---------------------------------------------------------------------*"
FORM RUTINA_EDITAR.

PERFORM RUTINA_AVISO1.

  IF RESPUESTA EQ '1'.

    IF RESTORE EQ MARCA.
      PERFORM RESTAURAR_DESDE_PC.
    ELSE.
      READ REPORT PROGRAMA INTO TBLPGM.
      RETCODE = SY-SUBRC.
    ENDIF.
```

```
    IF RETCODE EQ 0.
      EDITOR-CALL FOR TBLPGM.
    ENDIF.

    PERFORM COMPROBAR_SINTAXIS.

    IF RETCODE EQ 0.
      INSERT REPORT PROGRAMA FROM TBLPGM.
    ELSE.
      MESSAGE ID 38 TYPE 'I' NUMBER 016
         WITH 'EL PROGRAMA TIENE ERRORES DE SINTAXIS'.
      INSERT REPORT PROGRAMA FROM TBLPGM.
    ENDIF.

  ENDIF.

ENDFORM.


*---------------------------------------------------------------------*"
*       FORM RUTINA_LANZAR                                            *"
*---------------------------------------------------------------------*"
FORM RUTINA_LANZAR.

  IF RESTORE EQ MARCA.
    PERFORM RESTAURAR_DESDE_PC.
    INSERT REPORT PROGRAMA FROM TBLPGM.
    SUBMIT (PROGRAMA) AND RETURN VIA SELECTION-SCREEN.
  ELSE.
    SUBMIT (PROGRAMA) AND RETURN VIA SELECTION-SCREEN.
  ENDIF.

ENDFORM.

*---------------------------------------------------------------------*"
*       FORM RUTINA_AVISO1                                            *"
*---------------------------------------------------------------------*"
FORM RUTINA_AVISO1.

CALL FUNCTION 'POPUP_FOR_INTERACTION'
       EXPORTING"
            HEADLINE       = '*** ATENCION ***'
            TEXT1          = 'Este es un Editor muy especial y su uso '
            TEXT2          = 'conlleva un elevado riesgo. Si no está  '
            TEXT3          = 'Ud. muy seguro de lo que desea modificar'
            TEXT4          = 'es mejor pulse el botón de CANCELAR.    '
            TEXT5          = '                                        '
            TEXT6          = '                                        '
            TICON          = 'W'
            BUTTON_1       = 'Continuar'
            BUTTON_2       = 'CANCELAR'
            BUTTON_3       = ' '
       IMPORTING
            BUTTON_PRESSED = RESPUESTA
       EXCEPTIONS
            OTHERS = 1.
ENDFORM.

*---------------------------------------------------------------------*"
*       FORM RESTAURAR_DESDE_PC                                       *"
```

```
*---------------------------------------------------------------------*"
FORM RESTAURAR_DESDE_PC.

  CALL FUNCTION 'WS_UPLOAD'
       EXPORTING
            filename = RESTPAT
            FILETYPE            = 'ASC'
       TABLES
            DATA_TAB = TBLPGM
       EXCEPTIONS
            CONVERSION_ERROR = 1
           FILE_OPEN_ERROR = 2
            FILE_READ_ERROR = 3
            INVALID_TABLE_WIDTH = 4
            INVALID_TYPE = 5
            NO_BATCH = 6
            UNKNOWN_ERROR = 7
            OTHERS = 8.

  RETCODE = SY-SUBRC.

ENDFORM.

*---------------------------------------------------------------------*"
*       FORM COMPROBAR_SINTAXIS                                       *"
*---------------------------------------------------------------------*"
FORM COMPROBAR_SINTAXIS.

CLEAR: RETCODE.
.
.
.
ENDFORM.
```

Generador de Skeletons

```
* ********************************************************************** *
*    Autor ............................ Jordi Calopa Bosch (JCB)         *
*    Generación ....................... 23.07.2002                       *
*    Entorno de desarrollo ............ SAP/R3 (31I) bajo Unix           *
*                                                                        *
*   Generador de Skeletons                                               *
*                                                                        *
* ********************************************************************** *
program zxxxxxxx
        LINE-SIZE 120
        NO STANDARD PAGE HEADING.

include zxxxxxxx.

* *** Declaración de tablas y Vistas de base de Datos            *** *
TABLES: DD02L, DD03L, USR03, TRDIR.

* *** Declaración de variables Globales del Programa             *** *
DATA: RESP(01)           TYPE C,
      NAMETAB(10)        TYPE C,
      FOUNFLD(01)        TYPE C VALUE 'F',
      FIELD_CLASIF(30)   TYPE C,
      SKT                LIKE SY-CPROG  VALUE 'Z'.

DATA: BEGIN OF T1 OCCURS 1,
        TABNAME   LIKE DD03L-TABNAME  ,
        FIELDNAME LIKE DD03L-FIELDNAME,
      END OF T1.

DATA: BEGIN OF T2 OCCURS 1,
         R(255) TYPE C,
END OF T2.

*---------------------------------------------------------------------*
*       START-OF-SELECTION                                            *
*---------------------------------------------------------------------*
START-OF-SELECTION.

  skt+01(08) = sy-uname.
  skt+09(01) = '_'.
  PERFORM GENERADOR_PROGRAMAS.

END-OF-SELECTION.

  IF RETCODE EQ 0.
    FORMAT INTENSIFIED ON.
    FORMAT COLOR COL_HEADING.
    ULINE.
    WRITE: /01 SY-VLINE,
           AT SY-LINSZ SY-VLINE.
    WRITE: /01 SY-VLINE,
                'Generado el skelecton correctamente. Nombre',
               SKT COLOR COL_NORMAL,
           AT SY-LINSZ SY-VLINE.
    WRITE: /01 SY-VLINE,
           AT SY-LINSZ SY-VLINE.
```

```
    ULINE.
  ELSE.
    FORMAT INTENSIFIED ON.
    FORMAT COLOR COL_NEGATIVE.
    ULINE.
    WRITE: /01 SY-VLINE,
               'NO SE HA GENERADO NINGUN SKELETON',
           AT SY-LINSZ SY-VLINE.
    ULINE.
  ENDIF.

*---------------------------------------------------------------------*
*       FORM GENERADOR_PROGRAMAS                                      *
*---------------------------------------------------------------------*
FORM GENERADOR_PROGRAMAS.

  DATA: BEGIN OF TM OCCURS 9,
           C1(01),
           C2(80),
        END OF TM.

  TM-C1 = ' '.
  TM-C2 = ' Skeleton de un Programa Batch        '. APPEND TM.
  TM-C2 = ' Skeleton de una Lista Interactiva    '. APPEND TM.
  TM-C2 = ' Skeleton para Listar una Tabla de BD '. APPEND TM.
  TM-C2 = ' Skeleton Programa Nuevo              '. APPEND TM.

  CALL FUNCTION 'K_POPUP_TO_DECIDE'
    EXPORTING
      PAR_TEXT1        = '                      '
      PAR_TITLE        = 'GENERADOR DE SKELETONS'
    IMPORTING
      PAR_REPLY        = RESP
    TABLES
      TAB_LINES        = TM
    EXCEPTIONS
      NOT_ENOUGH_LINES = 1
      TOO_MANY_LINES   = 2.

  RETCODE = SY-SUBRC.

  IF RESP = 'A'.
    RETCODE = 4.
    EXIT.
  ENDIF.

  IF RETCODE EQ 0.

    IF RESP EQ '1'.
      skt+10(05) = 'BATCH'.
      PERFORM RPG_SKELETON_BATCH.

    ELSEIF RESP EQ '2'.
      skt+10(05) = 'LISTA'.
      PERFORM RPG_SKELETON_LISTA.

    ELSEIF RESP EQ '3'.
      skt+10(05) = 'TABLA'.
      PERFORM RPG_TABLAS.
```

```
    ELSEIF RESP EQ '4'.
      skt+10(05) = 'NUEVO'.
      PERFORM RPG_NEWPGM.

    ELSE.
      RETCODE = 4.
    ENDIF.

  ENDIF.

ENDFORM.                    "GENERADOR_PROGRAMAS

*---------------------------------------------------------------------*
*       FORM RPG_SKELETON_BATCH                                       *
*---------------------------------------------------------------------*
FORM RPG_SKELETON_BATCH.

  PERFORM CABECERA_REPORT.
  PERFORM R_BLANCO.
  T2-R = '* Declaración de Tablas y Vistas de BD    '.      APPEND T2.
  PERFORM R_BLANCO.
  T2-R = 'tables: t001.                             '.      APPEND T2.
  T2-R = '                  '.                              APPEND T2.
  T2-R = '                  '.                              APPEND T2.
  PERFORM R_BLANCO.
  T2-R = '* Evento ANTES DE LA PANTALLA DE SELECCION'.      APPEND T2.
  PERFORM R_BLANCO.
  T2-R = 'initialization.'.                                 APPEND T2.
  T2-R = '                  '.                              APPEND T2.
  T2-R = '                  '.                              APPEND T2.
  PERFORM R_BLANCO.
  T2-R = '* Evento DESPUES DE LA PANTALLA DE SELECCION'.    APPEND T2.
  PERFORM R_BLANCO.
  T2-R = 'at selection-screen.'.                            APPEND T2.
  T2-R = '                  '.                              APPEND T2.
  T2-R = '                  '.                              APPEND T2.
  PERFORM R_BLANCO.
  T2-R = '* Evento AL PRINCIPIO DEL PROGRAMA'.              APPEND T2.
  PERFORM R_BLANCO.
  T2-R = 'start-of-selection.'.                             APPEND T2.
  T2-R = '                  '.                              APPEND T2.
  T2-R = '   call function ''Y_UPD_T9S00'' '.               APPEND T2.
  T2-R = '      exceptions                 '.               APPEND T2.
  T2-R = '          others = 1.            '.               APPEND T2.
  T2-R = '                  '.                              APPEND T2.
  T2-R = '   write ''SKELETON BATCH''.'.                    APPEND T2.
  T2-R = '                  '.                              APPEND T2.
  T2-R = '                  '.                              APPEND T2.
  PERFORM R_BLANCO.
  T2-R = '* Evento AL FINAL DEL PROGRAMA    '.              APPEND T2.
  PERFORM R_BLANCO.
  T2-R = 'end-of-selection.'.                               APPEND T2.
  T2-R = '                  '.                              APPEND T2.
  T2-R = '                  '.                              APPEND T2.
  PERFORM R_BLANCO.
  T2-R = '* Evento AL PRINCIPIO DE PAGINA   '.              APPEND T2.
  PERFORM R_BLANCO.
  T2-R = 'top-of-page.'.                                    APPEND T2.
  T2-R = '                  '.                              APPEND T2.
  T2-R = '                  '.                              APPEND T2.
  PERFORM R_BLANCO.
```

```
  T2-R = '* Evento AL FINAL DE PAGINA       '.              APPEND T2.
  PERFORM R_BLANCO.
  T2-R = 'end-of-page.'.                                    APPEND T2.
  T2-R = '                  '.                              APPEND T2.
  T2-R = '                  '.                              APPEND T2.

  PERFORM INSERTAR_PGM.

ENDFORM.                    "RPG_SKELETON_BATCH


*---------------------------------------------------------------------*
*       FORM RPG_SKELETON_LISTA                                       *
*---------------------------------------------------------------------*
FORM RPG_SKELETON_LISTA.

  PERFORM CABECERA_REPORT.
  PERFORM R_BLANCO.
  T2-R = '* Declaración de Tablas y Vistas de BD    '.      APPEND T2.
  PERFORM R_BLANCO.
  T2-R = 'tables: t001.                             '.      APPEND T2.
  T2-R = '                  '.                              APPEND T2.
  T2-R = '                  '.                              APPEND T2.
  PERFORM R_BLANCO.
  T2-R = '* Evento ANTES DE LA PANTALLA DE SELECCION'.      APPEND T2.
  PERFORM R_BLANCO.
  T2-R = 'initialization.'.                                 APPEND T2.
  T2-R = '                  '.                              APPEND T2.
  T2-R = '                  '.                              APPEND T2.
  PERFORM R_BLANCO.
  T2-R = '* Evento DESPUES DE LA PANTALLA DE SELECCION'.    APPEND T2.
  PERFORM R_BLANCO.
  T2-R = 'at selection-screen.'.                            APPEND T2.
  T2-R = '                  '.                              APPEND T2.
  T2-R = '                  '.                              APPEND T2.
  PERFORM R_BLANCO.
  T2-R = '* Evento AL PRINCIPIO DEL PROGRAMA'.              APPEND T2.
  PERFORM R_BLANCO.
  T2-R = 'start-of-selection.'.                             APPEND T2.
  T2-R = '                  '.                              APPEND T2.
  T2-R = '   call function ''Y_UPD_T9S00'' '.               APPEND T2.
  T2-R = '      exceptions                 '.               APPEND T2.
  T2-R = '          others = 1.            '.               APPEND T2.
  T2-R = '                  '.                              APPEND T2.
  T2-R = '   write ''Haga doble click sobre esta línea''.'. APPEND T2.
  T2-R = '                  '.                              APPEND T2.
  T2-R = '                  '.                              APPEND T2.
  PERFORM R_BLANCO.
  T2-R = '* Evento AL FINAL DEL PROGRAMA    '.              APPEND T2.
  PERFORM R_BLANCO.
  T2-R = 'end-of-selection.'.                               APPEND T2.
  T2-R = '                  '.                              APPEND T2.
  T2-R = '                  '.                              APPEND T2.
  PERFORM R_BLANCO.
  T2-R = '* Evento AL PRINCIPIO DE LA PRIMERA PAGINA   '.   APPEND T2.
  PERFORM R_BLANCO.
  T2-R = 'top-of-page.'.                                    APPEND T2.
  T2-R = '                  '.                              APPEND T2.
  T2-R = '                  '.                              APPEND T2.
  PERFORM R_BLANCO.
  T2-R = '* Evento AL PRINCIPIO DE LAS PAGINAS SIGUIENTES'. APPEND T2.
```

```
  PERFORM R_BLANCO.
  T2-R = 'top-of-page during line-selection.'.              APPEND T2.
  T2-R = '                    '.                            APPEND T2.
  T2-R = '                    '.                            APPEND T2.
  PERFORM R_BLANCO.
  T2-R = '* Evento AL FINAL DE PAGINA       '.              APPEND T2.
  PERFORM R_BLANCO.
  T2-R = 'end-of-page.'.                                    APPEND T2.
  T2-R = '                    '.                            APPEND T2.
  T2-R = '                    '.                            APPEND T2.
  PERFORM R_BLANCO.
  T2-R = '* Evento AL SELECCIONAR UNA LINEA '.              APPEND T2.
  PERFORM R_BLANCO.
  T2-R = 'at line-selection.'.                              APPEND T2.
  T2-R = '   if sy-curow eq 1. '.                           APPEND T2.
  T2-R = '   write ''Esto es una LISTA INTERACTIVA    ''.'. APPEND T2.
  T2-R = '   write /''Haga doble click sobre esta línea''.'. APPEND T2.
  T2-R = '   sy-lsind = 0.                                  '. APPEND T2.
  T2-R = '   else.                                          '. APPEND T2.
  T2-R = '   write ''Haga doble click sobre esta línea''.'. APPEND T2.
  T2-R = '   endif. '.                                      APPEND T2.
  T2-R = '                    '.                            APPEND T2.

  PERFORM INSERTAR_PGM.

ENDFORM.                    "RPG_SKELETON_LISTA

*---------------------------------------------------------------------*
*       FORM RPG_TABLAS                                               *
*---------------------------------------------------------------------*
FORM RPG_TABLAS.

  IF NAMETAB IS INITIAL.

    CALL FUNCTION 'POPUP_TO_GET_VALUE'
      EXPORTING
        FIELDNAME           = 'TABNAME'
        TABNAME             = 'DD02L'
        TITEL               = 'Nombre de la'
        VALUEIN             = NAMETAB
      IMPORTING
        VALUEOUT            = NAMETAB
      EXCEPTIONS
        FIELDNAME_NOT_FOUND = 1
        OTHERS              = 2.

  ENDIF.

  TRANSLATE NAMETAB TO UPPER CASE.

  SELECT SINGLE * FROM DD02L WHERE TABNAME EQ NAMETAB AND
                                   AS4LOCAL EQ 'A'    AND
                                   AS4VERS EQ '0000'.
  IF SY-SUBRC NE 0.
    FORMAT INTENSIFIED OFF.
    WRITE: /01 'Tabla NO encontrada', NAMETAB COLOR 6.
    RETCODE = 4.
    STOP.
  ENDIF.

  REFRESH T1.
```

```
SELECT * FROM DD03L WHERE TABNAME EQ DD02L-TABNAME AND
                          AS4LOCAL EQ DD02L-AS4LOCAL.
  IF DD03L-FIELDNAME NE '.INCLUDE'.
    MOVE-CORRESPONDING DD03L TO T1.
    COLLECT T1.
    MOVE 'T' TO FOUNFLD.
  ENDIF.
ENDSELECT.

MOVE T1-FIELDNAME TO FIELD_CLASIF.
IF FOUNFLD EQ 'F'.
  FORMAT INTENSIFIED OFF.
  WRITE: /01 'Campos NO encontrados', NAMETAB COLOR 6.
  RETCODE = 4.
  STOP.
ENDIF.

PERFORM CABECERA_REPORT.

t2-r+00 = 'TABLES:'.
t2-r+10 = nametab.
T2-R+25 = '.'.                                          APPEND T2.
perform r_blanco.
T2-R = 'data: lincab1(40)    TYPE C.'.                  APPEND T2.
perform r_blanco.
T2-R = 'DATA: BEGIN OF TABLA01 OCCURS 1. '.             APPEND T2.
t2-r = '      include structure '.
t2-r+25 = nametab.
T2-R+50 = '.'.                                          APPEND T2.
T2-R = 'DATA: END OF TABLA01.'.                         APPEND T2.
perform r_blanco.
T2-R = 'start-of-selection.'.                           APPEND T2.
perform r_blanco.
T2-R = 'move ''Listado tabla           '' to lincab1.'.
T2-R+20 = NAMETAB(10).
T2-R+33 = ' '' to lincab1.                          '.  APPEND T2.
perform r_blanco.
t2-r = 'SELECT * FROM'.
t2-r+15 = nametab.
T2-R+30 = '.'.                                          APPEND T2.
t2-r = '   MOVE-CORRESPONDING '.
t2-r+25 = nametab.
T2-R+40 = ' TO TABLA01.'.                               APPEND T2.
T2-R = '   APPEND TABLA01.'.                            APPEND T2.
T2-R = 'ENDSELECT.'.                                    APPEND T2.
perform r_blanco.
T2-R = 'END-OF-SELECTION.'.                             APPEND T2.
perform r_blanco.
T2-R = '*  format intensified off.'.                     APPEND T2.
T2-R = '*  format color col_normal.'.                   APPEND T2.
perform r_blanco.
T2-R = '   LOOP AT TABLA01.'.                           APPEND T2.
T2-R = '      write: /01           '.                   APPEND T2.
LOOP AT T1.
  t2-r = '        TABLA01-'.
  move t1-fieldname to t2-r+16.
  MOVE ','          TO T2-R+47.                       APPEND T2.
ENDLOOP.
T2-R = '     space   . '.                               APPEND T2.
T2-R = '   ENDLOOP.     '.                             APPEND T2.
T2-R = '   uline.       '.                             APPEND T2.
```

```
  perform r_blanco.
  T2-R = 'TOP-OF-PAGE.    '.                                    APPEND T2.
  T2-R = '   WRITE: /03 LINCAB1.'.                              APPEND T2.
  T2-R = '   ULINE.       '.                                    APPEND T2.

  PERFORM INSERTAR_PGM.

ENDFORM.                    "RPG_TABLAS

*---------------------------------------------------------------------*
*       FORM RPG_NEWPGM                                               *
*---------------------------------------------------------------------*
FORM RPG_NEWPGM.

  PERFORM CABECERA_REPORT.
  PERFORM INSERTAR_PGM.

ENDFORM.                    "RPG_NEWPGM



*&---------------------------------------------------------------------*
*&      Form  CABECERA_REPORT
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
FORM CABECERA_REPORT.

  DATA: NOMBRE(40).

  SELECT SINGLE * FROM USR03
         WHERE BNAME EQ SY-UNAME.

  IF SY-SUBRC EQ 0.
    CONCATENATE USR03-NAME1 USR03-NAME2 '(' SY-UNAME ')'
                INTO NOMBRE SEPARATED BY ' '.
  ELSE.
    NOMBRE = SY-UNAME.
  ENDIF.

  REFRESH T2.
  T2-R+00 = '* ************************************************* *'.
  APPEND T2.
  T2-R+00 = '* Autor:    '.
  T2-R+14 = NOMBRE.                           APPEND T2.
  T2-R+00 = '* Fecha:    '.
  T2-R+14 = sy-datlo.                         APPEND T2.
  T2-R+00 = '* Hora:     '.
  T2-R+14 = sy-timlo.                         APPEND T2.
  T2-R+00 = '* Host:     '.
  T2-R+14 = SY-HOST.                          APPEND T2.
  T2-R+00 = '* Sistema:    '.
  T2-R+14 = SY-OPSYS.                         APPEND T2.
  T2-R+00 = '* ************************************************* *'.
  APPEND T2.
  T2-R+00 = 'Program'.
  T2-R+09 = SKT.                            APPEND T2.
  T2-R = '         line-size 255'.               APPEND T2.
  T2-R = '         no standard page heading.'.   APPEND T2.
  T2-R = '                                  '.   APPEND T2.
```

```
ENDFORM.                    "CABECERA_REPORT

*&---------------------------------------------------------------------*
*&      Form  r_blanco
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
form r_blanco.

  T2-R = '*'.
  APPEND T2.

ENDFORM.                    "r_blanco

*&---------------------------------------------------------------------*
*&      Form  INSERTAR_PGM
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
FORM INSERTAR_PGM.

  insert report skt from t2.

ENDFORM.                    "INSERTAR_PGM

* ********************************************************************** *
* ***                       FIN DE PROGRAMA                          *** *
* ********************************************************************** *
```

Iconos en la Pantalla de Selección

```
* ********************************************************************** *
*  Autor ..............  Jordi Calopa Bosch (JCB)                         *
*  Entorno ............  SAP/R3 v4.7 - Linux                              *
*                                                                        *
*  Iconos en la pantalla de selección (Barra de Botones)                 *
*                                                                        *
* ********************************************************************** *
program zjcb301
        line-size 120
        no standard page heading.

tables: SSCRFIELDS,   " Campos en las imágenes de selección
        SMP_DYNTXT.   " Menu Painter: Interfase programa para textos dinámicos

parameters: dummy.

selection-screen function key 1.

initialization.
  SMP_DYNTXT-TEXT      = 'mi función'.
  SMP_DYNTXT-ICON_ID   = '@01@'.                " Código del Icono
  SMP_DYNTXT-ICON_TEXT = 'mi función'.
  SMP_DYNTXT-QUICKINFO = 'Info Breve'.
  SSCRFIELDS-FUNCTXT_01 = SMP_DYNTXT.

at selection-screen.
  if sy-ucomm eq 'FC01'.                      " Código de function del botón 1
    submit mi_otro_programa.
  endif.

start-of-selection.
  .
  .
```

Optimización de los accesos a tablas de BD

El programa que mostramos seguidamente realiza dos veces la misma operación de leer todos los pedidos de la sociedad 0101. Primeramente lo hace con la sentencia 'Select * from' tratando la tabla de BD directamente. En el segundo proceso (el tratamiento 2) se realiza la misma lectura pero, en esta ocasión, leyendo únicamente los campos a tratar y cargando una tabla de memoria interna."

Este programa se ha procesado varias veces en el entorno XXX, a diferentes horas del día y el resultado siempre ha sido muy parecido: El segundo método tarda una doceava parte del tiempo que tarda el primero.

```
REPORT YSLXTEST.

TABLES: EKKO.

DATA: HI    LIKE SY-UZEIT,
      HF    LIKE SY-UZEIT,
      HD    LIKE SY-UZEIT,
      WCONT TYPE I VALUE 0.

DATA: BEGIN OF T1 OCCURS 10000,
         BUKRS LIKE EKKO-BUKRS,
      END OF T1.

START-OF-SELECTION.

* *** Tratamiento 1 -     * * * MUY LENTO * * *              *** *
GET TIME.
MOVE SY-UZEIT TO HI.
SELECT * FROM EKKO.
   CHECK EKKO-BUKRS EQ '0101'.
   WCONT = WCONT + 1.
ENDSELECT.
GET TIME.
MOVE SY-UZEIT TO HF.
HD = HF - HI.
WRITE: /01 'Duración', HD.
WRITE: /01 'Registros Tratados', WCONT.
WCONT = 0.
REFRESH T1.

* *** Tratamiento 2 -      * * * RAPIDO * * *                *** *
GET TIME.
MOVE SY-UZEIT TO HI.
SELECT BUKRS FROM EKKO INTO TABLE T1 WHERE BUKRS EQ '0101'.
DESCRIBE TABLE T1 LINES WCONT.
GET TIME.
MOVE SY-UZEIT TO HF.
HD = HF - HI.
WRITE: /01 'Duración', HD.
WRITE: /01 'Registros Tratados', WCONT.
WCONT = 0.
REFRESH T1.
```

Interacción desde Abap con los comandos del sistema operativo DOS

Desde un programa desarrollado en ABAP/4 podemos efectuar llamadas a programas del PC" local mediante la utilización de la función WS_EXECUTE.

Obsérvese que la ejecución se puede realizar de forma síncrona o asíncrona: Si en las variables que exportamos se incluye la variable INFORM, entonces la ejecución es SINCRONA; es decir que SAP espera a que finalice el programa que ha sido llamado. Si no incluimos la variable INFORM entonces la ejecución es ASINCRONA y el programa Abap continua trabajando independientemente de cuando termine el programa del PC.

```
* ************************************************************* *"
* Descripción:  Este programa es un trigger de comandos del PC;      *"
*               es decir que permite disparar comandos del PC bajo   *"
*               sistema operativo DOS-WINDOWS.                       *"
* ************************************************************* *"
REPORT Yxxxxxxx"
       line-size 160"
       no standard page heading."

tables: t001."

DATA: PRM1(25) TYPE C,"
      resp,"
      pg(57)."

data: begin of tm occurs 1,"
         c1(01),"
         c2(80),"
      end of tm."

* ************************************************************* *"
* *** Inicio del programa principal                          *** *"
* ************************************************************* *"
START-OF-SELECTION."

* *** Preparación del menú de selección a displayar             *** *"
TM-C1 = ' '."
   TM-C2 = ' Microsoft WORD                '."
   APPEND TM."
   TM-C1 = ' '."
   TM-C2 = ' Microsoft EXCEL              '."
   APPEND TM."
   TM-C1 = ' '."
   TM-C2 = ' Microsoft ACCESS             '."
   APPEND TM."

* *** Llamada a la función que visualiza el menú                *** *"
   CALL FUNCTION 'K_POPUP_TO_DECIDE'"
      exporting"
         PAR_TEXT1   ="
            '                                '"
         PAR_TITLE   ="
            'Disparador de programas de pc   '"
      importing"
         PAR_REPLY = resp"
```

```
      tables"
         TAB_LINES = tm"
      exceptions"
         NOT_ENOUGH_LINES = 1"
         TOO_MANY_LINES = 2#"

* *** Tratamiento de la respuesta que introdujo el operador       *** *"
   CASE RESP."
     when '1'."
        concatenate 'c:\archivos de programa\'"
                    '\microsoft office\office\winword'  INTO PG."
        PRM1 = 'word'."
     when '2'."
        concatenate 'c:\archivos de programa\'"
'\microsoft office\office\excel   ' INTO PG."
        PRM1 = 'excel'."
     when '3'."
        concatenate 'c:\archivos de programa\'"
                    '\microsoft office\office\msaccess' INTO PG."
        PRM1 = 'access'."
   ENDCASE."

* *** Llamada a la función que dispara programas del PC           *** *"
   CALL FUNCTION 'WS_EXECUTE'"
      exporting"
         program = PG"
        inform         = pg"
      exceptions"
         FRONTEND_ERROR = 1"
         NO_BATCH = 2"
         PROG_NOT_FOUND = 3"
         ILLEGAL_OPTION = 4"
         others = 5."

* ************************************************************** *"
* *** Cabeceras del Listado                                    *** *"
* ************************************************************** *"
top-of-page."
  SY-TITLE = 'Trigger de comandos del PC'."
  perform cabecera using '0170'."
  include ysn0034."
```

Función externa para determinar e informar de años bisiestos

```
FUNCTION Y_BISIESTO."
* ********************************************************************* *"
* Autor ........................ Jordi Calopa Bosch (JCB)              *"
* Fecha Ultima Actualización .... 24/02/2000                           *"
* Entorno de desarrollo ......... SAP/R3 bajo Unix                     *"
*                                                                       *"
*        Función de detección e información de a¤os bisiestos          *"
*       ====================================================           *"
*                                                                       *"
*     Par metro de ENTRADA a la función:  param1 char (04)              *"
*                                                                       *"
*     Par metro de SALIDA de la función:  param2 char (01)              *"
*                                                                       *"
*        - La función devuelve 'S' si el a¤o es bisiesto,               *"
*          devuelve 'N' cuando no lo es y devuelve 'E' cuando           *"
*          el valor que se le envia no es numerico.                     *"
*                                                                       *"
* ********************************************************************* *"

  DATA: ANY TYPE I VALUE 0,
        ENTERO TYPE P,
        RESTO  TYPE P.

  IF PARAM1 CO '0123456789'.
     MOVE PARAM1 TO ANY.
  ELSE.
     PARAM2 = 'E'.
     EXIT.
  ENDIF.

  RESTO = ANY MOD 4.
  IF RESTO NE 0.
    PARAM2 = 'N'.
    EXIT.
  ELSE.
    RESTO = ANY MOD 100.
    IF RESTO NE 0.
      PARAM2 = 'S'.
      EXIT.
    ELSE.
      RESTO = ANY MOD 400.
      IF RESTO = 0.
        PARAM2 = 'S'.
      ELSE.
        PARAM2 = 'N'.
      ENDIF.
    ENDIF.
  ENDIF.

ENDFUNCTION.
```

Utilización de varios punteros sobre una misma tabla de BD

En el siguiente ejemplo se muestra como es posible acceder a una misma tabla ,de BD, utilizando varios punteros diferentes (Tres, en este caso en concreto). ,Para ello NO utilizamos la sentencia SELECT tradicional, sino que realizamos ,la lectura con la sentencia FETCH, con la cual es necesario realizar el OPEN y el CLOSE de cada uno de los punteros.

Obsérvese que primeramente se efectúa una lectura e impresión de dos campos de la tabla de sociedades, pero al encontrar una determinada sociedad (no importa cual) lanzamos la rutina TRATAR_PUNTERO2 que, utilizando otro puntero lee y lista la misma tabla. Estando dentro de esta rutina volvemos a interrumpirla (al encontrar otra sociedad cualquiera) para lanzar la rutina TRATAR_PUNTERO3 que tambien lee y lista la misma tabla.

```
program yxxxxxxx
        line-size 120
        no standard page heading.

tables: t001.

data: puntero1 type cursor,
      puntero2 like puntero1,
      puntero3 like puntero2.

data: begin of area_trabajo.
         include structure t001.
data: end of area_trabajo.

start-of-selection.

   open cursor puntero1 for
        select * from t001.
   do.
      fetch next cursor puntero1 into area_trabajo.
      if sy-subrc <> 0.
         close cursor puntero1.
         exit.
      endif.
      write: /01 area_trabajo-bukrs color col_total,
                 area_trabajo-butxt color col_total.
      if area_trabajo-bukrs eq '0101'.
         perform tratar_puntero2.
      endif.
   enddo.

* ********************************************************************* *
* *** Subrutinas                                                   *** *
* ********************************************************************* *
form tratar_puntero2.
   open cursor puntero2 for
        select * from t001.
   do.
      fetch next cursor puntero2 into area_trabajo.
      if sy-subrc <> 0.
         close cursor puntero2.
         exit.
      endif.
```

```
      write: /01 area_trabajo-bukrs color col_positive,
                 area_trabajo-butxt color col_positive.
      if area_trabajo-bukrs eq '0201'.
         perform tratar_puntero3.
      endif.
 enddo.
endform.

form tratar_puntero3.
   open cursor puntero3 for
        select * from t001.
   do.
      fetch next cursor puntero3 into area_trabajo.
      if sy-subrc <> 0.
         close cursor puntero3.
         exit.
      endif.
      write: /01 area_trabajo-bukrs color col_negative,
                 area_trabajo-butxt color col_negative.
 enddo.
endform.
```

Este ejemplo, en si mismo, no tiene ninguna lógica, pero el listado ilustra ,claramente como se han utilizado los diferentes punteros."

```
Recorrido principal de la tabla
  .
  .
          tratar_puntero2 (segundo recorrido)
                  .
                  .
                  .          tratar_puntero3 (3er recorrido)
                                  .
                                  .
                                  .
                             fin_recorrido_3
                  .
          fin_recorrido_2
  .
  .
final del recorrido
```

Grabación y Lectura de ficheros secuenciales en UNIX

```
* ********************************************************************* *"
* Descripción: En este programa preparamemos y grabaremos un fichero   *"
*              secuencial (en unix) que contendrá el valor de las      *"
*              funciones trigonométricas básicas de 0 a 360 grados.    *"
* ********************************************************************* *"
report yxxxxxxx"
       line-size 94"
       no standard page heading."
"
data: begin of registro,"
         gra(3)     type c,"
         sen(23)    type c,"
         cos(23)    type c,"
         filler(01) type c,"
      end of registro."
"
data: registros_grabados type i value 0."
"
parameters: fichero(64) type c lower case"
                        Default '/home2/c11/fi/jordi'.  "
"
start-of-selection."
"
   perform generar_fichero_unix."
"
* ********************************************************************* *"
* *** rutina de grabación del fichero secuencial                   *** *"
* ********************************************************************* *"
form generar_fichero_unix."
"
* *** declaración de variables locales                             *** *"
   data: contador(3) type n value 0,"
         grados      type f value 0,"
         pi          type f value '3.1415926536',"
         seno        type f value 0,"
         coseno      type f value 0,"
         tangente    type f value 0."
"
   open dataset fichero for output in text mode."
   if sy-subrc eq 0."
      do 360 times."
         contador = contador + 1#"
         grados = (contador * 2 * pi) / 360#"
         seno = sin( grados )."
         coseno = cos( grados )."
         move contador  to registro-gra."
         move seno      to registro-sen."
         move coseno    to registro-cos."
         transfer registro to fichero."
         if sy-subrc eq 0."
            registros_grabados = registros_grabados + 1#"
         endif."
      enddo."
      close dataset fichero."
      if sy-subrc eq 0."
         format intensified off."
```

```
         format color col_total."
         uline."
         write: /01 sy-vline,"
                    'registros grabados',"
                    registros_grabados color col_normal,"
                 94 sy-vline."
         uline."
      else."
         write: /01 'close error' color col_total."
      endif."
   else."
      write: /01 'open error' color col_total."
   endif."
"
endform."



* ********************************************************************* *"
* Descripción: En este programa leeremos y listaremos el fichero       *"
*              secuencial (en unix) que se ha generado en el ejemplo   *"
*              anterior.                                               *"
* ********************************************************************* *"
report yxxxxxxx"
       line-size 94"
       no standard page heading."
"
data: begin of registro,"
         gra(3)     type c,"
         sen(23)    type c,"
         cos(23)    type c,"
         filler(01) type c,"
      end of registro."
"
data: registros_leidos   type i value 0."
"
parameters: fichero(64) type c lower case"
                        Default '/home2/c11/fi/jordi'."
"
start-of-selection."
"
   perform leer_listar_fichero_unix."
"
"
* ********************************************************************* *"
* *** rutina de lectura y listado del fichero secuencial           *** *"
* ********************************************************************* *"
form leer_listar_fichero_unix."
"
* *** declaración de variables locales                             *** *"
   data: work1 type f,"
         work2 type f,"
         tangente like registro-sen."
"
   open dataset fichero for input."
   if sy-subrc ne 0."
      write: /01 'open error'."
      exit."
   endif."
"
```

```
   format intensified off."
   format color col_heading."
   uline."
   do."
      read dataset fichero into registro."
      if sy-subrc ne 0.                              "
         exit."
      endif."
      registros_leidos = registros_leidos + 1#"
      work1 = registro-sen."
      work2 = registro-cos."
      work1 = work1 / work2."
      tangente = work1."
      write: /01 sy-vline,"
                 registro-gra,"
                 registro-sen color col_normal,"
                 registro-cos color col_normal,"
                 tangente     color col_normal,"
             94 sy-vline."
   enddo."
   uline."
"
   close dataset fichero."
   if sy-subrc eq 0."
      format intensified off."
      format color col_total."
      uline."
      write: /01 sy-vline,"
                 'registros leidos  ',"
                 registros_leidos color col_normal,"
             94 sy-vline."
      uline."
   else."
      write: /01 'close error'."
      exit."
   endif."
"
endform."
```

Rutina de conversión de fechas

Presentamos una rutina de conversión de fechas a la cual se le envian 5 parámetros. Uno de entrada (a la rutina) que contiene una fecha en formato SY-DATUM y cuatro parámetros más que la rutina devuelve actualizados. Veamos seguidamente como podemos realizar la llamada a esta rutina:

```
.
.
data: fecha_gre(10) type c,"
      fecha_jul(5)  type c,"
      fecha_co9(8)  type c,"
      diferencia    type i."
"
parameters: entrada like sy-datum."
."
."
  perform gregoriana_juliana_c9 using     entrada"
                                changing fecha_gre"
                                         fecha_jul"
                                         fecha_co9"
                                         diferencia."
."
."
"
"
"
Observese que la rutina actualiza cuatro campos, a saber:"
"
1. El campo G en el cual se devuelve la fecha recibida en formato"
   Gregoriano (DD/MM/AAAA)."
"
2. El campo J en el cual se devuelve la fecha recibida en formato"
   Juliano (AADDD)."
"
3. El campo C en el que se devuelve la fecha en formato AAAAMMDD"
   y a la que se le ha aplicado el complemento a 9."
"
4. El campo D, que es un integer, devuelve la diferencia entre la"
   fecha enviada a la rutina y la fecha del dia. Tendr  valor negativo"
   si la fecha enviada es anterior a la actual, o bien valor positivo"
   si la fecha enviada es posterior a la actual."
"
"
*---------------------------------------------------------------------*"
*       FORM gregoriana_juliana_c9                                    *"
*---------------------------------------------------------------------*"
form gregoriana_juliana_c9 using f changing g j c d."
"
  data: aa(02)    type c,"
        mm(02)    type n,"
        dd(02)    type n,"
        salida(5) type n,"
        out(01)   type c,"
        wf        like sy-datum,"
        f1        like sy-datum,"
        f2        like f1."
"
```

```
  wf = f."
"
  call function 'Y_BISIESTO'                    "
       exporting "
            param1 = wf+00(04)"
       importing"
            PARAM2 = out"
       exceptions"
            others = 1."
"
  if out eq 'E'."
    exit."
  endif."
"
  aa = wf+02(02)."
  mm = wf+04(02)."
  dd = wf+06(02)."
  move aa to salida+00(02)."
"
  add dd to salida."
  if mm gt '11'."
    add 334 to salida."
  elseif mm gt '10'."
    add 304 to salida."
  elseif mm gt '09'."
    add 273 to salida."
  elseif mm gt '08'."
    add 243 to salida."
  elseif mm gt '07'."
    add 212 to salida."
  elseif mm gt '06'."
    add 181 to salida."
  elseif mm gt '05'."
    add 151 to salida."
  elseif mm gt '04'."
    add 120 to salida."
  elseif mm gt '03'."
    add  90 to salida."
  elseif mm gt '02'."
    add  59 to salida."
  elseif mm gt '01'."
    add  31 to salida."
  endif."
  if out eq 'S' and mm gt '02'."
    add 1 to salida."
  endif."

  move salida to j."
  write: wf to g using edit mask '__/__/____'."
  convert date wf into inverted-date c."
  move wf to f1."
  move sy-datum to f2."
  d = f1 - f2."

endform."
```

Comparación de tiempos de acceso (SELECT vs GET)

Hemos efectuado una prueba para comparar el tiempo de acceso a los ficheros maestros de materiales, mediante la utilizacin de la BD lógica por un lado y la utilización de sentencias SELECT por otro."

Nótese que cuando utilizamos los eventos GET con la BD lógica MSM cargamos completamente el buffer del registro mientras que cuando utilizamos las sentencias SELECT solamente cargamos los campos que precisamos para trabajar.

En este orden de ideas observamos que, cuando la cantidad de campos a tratar es reducida entonces el acceso con sentencias SELECT es mucho más rápido que con los eventos GET.

```
,* Programa de Test"
,"
,report yxxxxxxx."
,"
,data: t1 type i,"
,      t2 like t1,"
,      t3 like t1,"
,      t4 like t1."
,"
,tables: mara, marc, mard."
,"
,start-of-selection."
,"
,* *** PRIMERA PARTE ***"
,  get run time field t1."
,  get mara."
,  get marc."
,  get mard."
,"
,end-of-selection."
,"
,  get run time field t2."
,  t2 = t2 - t1."
,  write: /01 'tiempo utilizando GETs', t2."
,"
,* *** SEGUNDA PARTE ***"
,  perform segunda_parte."
,"
,*---------------------------------------------------------------------*"
,*       form segunda_parte                                            *"
,*---------------------------------------------------------------------*"
,form segunda_parte."
,"
,  data: ma-matnr like mara-matnr,"
,        mc-matnr like marc-matnr,"
,        mc-werks like marc-werks,"
,        md-matnr like mard-matnr,"
,        md-werks like mard-werks,"
,        md-lgort like mard-lgort."
,"
,  get run time field t3."
,"
,  select matnr from mara"
```

```
,         into ma-matnr."
,"
,    select matnr werks from marc"
,           into (mc-matnr,mc-werks)"
,           where matnr eq ma-matnr."
,"
,      select matnr werks lgort from mard"
,             into (md-matnr,md-werks,md-lgort)"
,             where matnr eq mc-matnr and"
,                   werks eq mc-werks."
,"
,      endselect."
,    endselect."
,  endselect."
,"
,  get run time field t4."
,  t4 = t4 - t3."
,  write: /01 'tiempo utilizando SELECTs', t4."
,"
,endform."
```

Comparación de Tablas de BD entre diferentes entornos

En ocasiones puede ser necesario comparar el contenido de una determinada tabla de BD, entre diferentes entornos. Para ello disponemos de una función que nos permite leer el contenido de una tabla situada en otro entorno diferente del actual.

En el ejemplo, que mostramos seguidamente, vamos a comparar la tabla T162 del entorno XXX, en el que rueda el programa, con la misma tabla del entorno de producción YYY-200.

```
* Ejemplo"
,"
,* ********************************************************************* *"
,* Descripción: Comparación de tablas de BD entre el entorno actual   %D*"
,*              (C11) y el entorno de producción C13-200, para tablas %D*"
,*              cuya longitud de registro no exceda de 512 bytes.     %D*"
,* ********************************************************************* *"
,report yxxxxxxx"
,       line-size 170"
,       no standard page heading."
,"
,tables: t162."
,"
,data: begin of t1 occurs 25000."
,        include structure t162."
,data: end of t1."
,"
,data: begin of t2 occurs 1000."
,        include structure rfc_db_opt."
,data: end of t2."
,"
,data: begin of t3 occurs 1000."
,        include structure rfc_db_fld."
,data: end of t3."
,"
,data: begin of t4 occurs 1000."
,        include structure tab512."
,data: end of t4."
,"
,data: begin of t5 occurs 1000."
,        include structure t162."
,data: end of t5."
,"
,data: nombre_tabla      like dd02l-tabname     value 'T162',"
,      destino_logico    like rfcdes-rfcdest    value 'C13-200'."
,"
,*---------------------------------------------------------------------*"
,*       START-OF-SELECTION                                            *"
,*---------------------------------------------------------------------*"
,start-of-selection."
,"
,  perform cargar_tabla_produccion."
,"
,  format intensified on."
,  uline."
,"
```

```
,* *** Tratamos la tabla interna (imagen de la de producción) y la  *** *"
,* *** comparamos con la tabla del entorno actual (C11)             *** *"
,  loop at t5."
,"
,    format color col_heading."
,    write: /01 sy-vline,"
,               'C13',"
,               t5-mandt,"
,               t5-flref,"
,               t5-faus1,"
,               t5-faus2,"
,           170 sy-vline."
,"
,    select single * from t162"
,           where flref eq t5-flref."
,"
,    if sy-subrc eq 0."
,      if t5-faus1 eq t162-faus1 and"
,         t5-faus2 eq t162-faus2."
,        format color col_normal."
,      else."
,        format color col_group."
,      endif."
,      write: /01 sy-vline,"
,                 'C11',"
,                 t162-mandt,"
,                 t162-flref,"
,                 t162-faus1,"
,                 t162-faus2,"
,             170 sy-vline."
,    else."
,      format color col_negative."
,      write: /01 sy-vline,"
,                 'NOTF',"
,                 t5-mandt,"
,                 t5-flref,"
,                 t5-faus1,"
,                 t5-faus2,"
,             170 sy-vline."
,    endif."
,  endloop."
,"
,  uline."
,"
,end-of-selection."
,"
,"
,*---------------------------------------------------------------------*"
,*       FORM CARGAR_TABLA_PRODUCCION                                  *"
,*---------------------------------------------------------------------*"
,form cargar_tabla_produccion."
,"
,* *** Esta función devuelve una tabla interna (T4) con el contenido ** *"
,* *** de los registros de la tabla de BD indicada. Cada registro es ** *"
,* *** devuelto en un string de 512 bytes que deberá ser tratado a   ** *"
,* *** posteriori, según convenga.                                   ** *"
,"
,  CALL FUNCTION 'RFC_READ_TABLE' DESTINATION DESTINO_LOGICO"
,       exporting"
,            QUERY_TABLE = nombre_tabla"
,       tables"
```

```
,            Options = t2"
,            FIELDS = t3"
,            data = t4"
,       exceptions"
,            TABLE_NOT_AVAILABLE = 1"
,            TABLE_WITHOUT_DATA = 2"
,            OPTION_NOT_VALID = 3"
,            FIELD_NOT_VALID = 4"
,            NOT_AUTHORIZED = 5"
,            DATA_BUFFER_EXCEEDED = 6"
,            others = 7."
,"
,* *** Seguidamente efectuamos el desglose del string y cargamos una ** *"
,* *** tabla interna con la estructura de la tabla de BD leida del   ** *"
,* *** entorno de producción.                                        ** *"
,"
,  if sy-subrc eq 0."
,    loop at t4."
,      t5-mandt = t4+00(03)."
,      t5-flref = t4+03(04)."
,      t5-faus1 = t4+07(70)."
,      t5-faus2 = t4+77(70)."
,      append t5."
,    endloop."
,  endif."
,"
,endform."
```

Interacción Abap - unix

En anteriores news presentamos una de las formas posibles de ejecutar comandos de Unix desde un programa Abap/4. No obstante en esta ocasión mostramos una forma mucho más sencilla para la consecución del mismo objetivo; ejecutar comandos de unix desde Abap.

```
,* Programa Ejemplo"
,"
,report yxxxxxxx"
,       line-size 250"
,       no standard page heading."
,"
,* *** Definición de Tablas Din micas en Memoria Interna            *** *"
,data: begin of t1 occurs 100,"
,         line(200),"
,      end of t1."
,"
,* *** Definición de Variables Globales del Programa                *** *"
,data: w_comando_unix(250)   type c,"
,      txt120(120)           type c."
,"
,* *** Definición de los Criterios de Selección                     *** *"
,selection-screen: begin of block bl1 with frame title text-001."
,selection-screen: skip."
,parameters: wcom(70) type c lower case obligatory."
,selection-screen: skip."
,selection-screen: end of block bl1."
,"
,"
,start-of-selection."
,"
,  perform ejecuta_comando."
,  "
,end-of-selection."
,"
,  perform listar_t1."
,"
,"
,*---------------------------------------------------------------------*"
,*       FORM EJECUTA_COMANDO                                          *"
,*---------------------------------------------------------------------*"
,form ejecuta_comando."
,"
,  concatenate wcom ' '"
,              into w_comando_unix separated by space."
,"
,  refresh t1."
,"
,  CALL 'SYSTEM' ID 'COMMAND' FIELD W_COMANDO_UNIX"
,                id 'TAB'  FIELD T1-*SYS*."
,"
,endform."
,"
,"
,*---------------------------------------------------------------------*"
,*       FORM LISTAR_T1                                                *"
```

```
,*---------------------------------------------------------------------*"
,form listar_t1."
,"
,  format intensified on."
,  format color col_heading."
,  uline."
,"
,  loop at t1."
,    write: /01 sy-vline,"
,                t1-line,"
,           at sy-linsz sy-vline."
,  endloop."
,"
,  uline."
,"
,endform."
```

Transferencia de datos entre programas Abap mediante -Common part-

Una de las posibles formas de intercambio de datos consiste en utilizar la -Common Part- que es una área de memoria compartida entre programas, que permite transferir información entre ellos.

Seguidamente mostramos dos programas de Ejemplo; El primero que es el que realiza la llamada y el segundo el que contiene la subrutina que ser llamada. Este segundo programa bien pudiera ser únicamente un conjunto de rutinas, con una -Common Part- definida, para ser llamadas desde distintos programas principales.

```
* Programa 1 - Realiza la llamada a la Subrutina del programa 2

REPORT ZXXXXXXX.

DATA: BEGIN OF COMMON PART JCB,
   BEGIN OF T1 OCCURS 1000000,
      registro(255),
   END OF T1,
END OF COMMON PART.

START-OF-SELECTION.

  PERFORM CARGAR_T1.
  PERFORM VISUALIZAR(ZYYYYYYY).

*---------------------------------------------------------------------*
 * Form; CARGAR_T1 *
*---------------------------------------------------------------------*
FORM CARGAR_T1.

  CLEAR: T1.
  REFRESH: T1.
  T1-REGISTRO = 'Primer Registro de Pruebas'.
  APPEND T1.
  T1-REGISTRO = 'Segundo Registro de Pruebas'.
  APPEND T1.
  T1-REGISTRO = 'Tercer Registro de Pruebas'.
  APPEND T1.

ENDFORM.
```

Lógicamente la definición del area de memoria intermedia debe ser exactamente igual en ambos programas, si bien NO es necesario utilizar la misma nomenclatura de campos. En cualquier caso NO es necesario pasar una TABLA pueden ser un conjunto de variables independientes entre si, siempre y cuando se definan exactamente con el mismo orden y formato.

```
* Programa 2 - Contiene la Subrutina que es llamada desde el programa 1

REPORT; ZYYYYYYY
        Line-Size 120
        NO STANDARD PAGE HEADING.

DATA: BEGIN OF COMMON PART JCB,
   BEGIN OF TABLA OCCURS 1000000,
```

```
      registro(255),
   END OF TABLA,
END OF COMMON PART.

*---------------------------------------------------------------------*
* Form; VISUALIZAR *
*---------------------------------------------------------------------*
FORM VISUALIZAR.

  LOOP AT TABLA.
    WRITE: TABLA.
  ENDLOOP.

ENDFORM.
```

Transferencia de datos entre programas Abap mediante - Export / Import -

Otra de las posibles formas de intercambio de datos entre programas consiste en utilizar las sentencias Export e Import. La primera registra un conjunto de variables en la memoria interna y la segunda permite recuperar el contenido de las mismas.

Nótese que es obligatorio que el programa que realiza el Import sea submitido por el ,programa que previamente realiza el Export, porque aparentemente utilizan un área de ,memoria que se resetea al inicio de cada programa.

Seguidamente mostramos dos programas; El primero que es el que realiza el Export ,y la llamada al segundo programa y el segundo programa que contiene el IMPORT y la ,impresión de las variables importadas.

```
* Programa 1"
,"
,REPORT ZXXXXXXX."
,"
,DATA: A(10) TYPE C,"
,      B(10) TYPE C,"
,      C(10) TYPE C."
,"
,START-OF-SELECTION."
,"
,  A = '0123456789'."
,  B = 'ABCDEFGHIJ'."
,  C = 'A1B2C3D4E5'."
,"
,  EXPORT A B C TO MEMORY  ID 'JCB'."
,"
,  SUBMIT ZYYYYYYY."
,"
,"
,* Programa 2"
,"
,REPORT ZYYYYYYY."
,"
,DATA: A(10) TYPE C,"
,      B(10) TYPE C,"
,      C(10) TYPE C."
,"
,START-OF-SELECTION."
,"
,  Import A B C From MEMORY ID 'JCB'."
,  WRITE: /01 A, B, C."
,"
,"
,"
,También es posible utilizar estas sentencias dentro de un mismo programa por
Ejemplo"
,si queremos guardar el contenidos de variables que van a ser reseteadas
temporalmente."
,"
,* Ejemplo 3 - Utilización de Export / Import en un mismo programa"
,"
,REPORT ZJJJJJJJ."
```

```
,"
,DATA: A(10) TYPE C,"
,      B(10) TYPE C,"
,      C(10) TYPE C."
,"
,START-OF-SELECTION."
,"
,  A = '0123456789'."
,  B = 'ABCDEFGHIJ'."
,  C = 'A1B2C3D4E5'."
,"
,  EXPORT A B C TO MEMORY  ID 'JCB'."
,"
,  CLEAR: A, B, C."
,."
,."
,."
,."
,  IMPORT A B C From MEMORY ID 'JCB'."
,  WRITE: /01 A, B, C."
```

SQL Nativo

```
* ********************************************************************* *
*                                                                       *
*  Este programa accede, mediante SQL nativo, al fichero de pedidos     *
*  EKKO, para listar los campos de Mandante, Número de Pedido y         *
*  Sociedad. Al acceder mediante el SQL Nativo, se leen todos los       *
*  registros de la tabla EKKO, sea cual sea el mandante del registro,   *
*  lo que podría ser de utilidad en determinadas ocasiones.             *
*                                                                       *
* ********************************************************************* *
REPORT ZXXXXXXX
       LINE-SIZE 120
       NO STANDARD PAGE HEADING.

DATA: MANDANTE(3) TYPE C,
      PEDIDO(10)  TYPE C,
      SOCIEDAD(4) TYPE C.

START-OF-SELECTION.

  EXEC SQL PERFORMING LISTAR.
    SELECT MANDT, EBELN, BUKRS
           INTO :MANDANTE, :PEDIDO, :SOCIEDAD
           FROM EKKO
           ORDER BY MANDT
  ENDEXEC.

*---------------------------------------------------------------------*
*       FORM LISTAR                                                   *
*---------------------------------------------------------------------*
FORM LISTAR.

  IF MANDANTE EQ SY-MANDT.
    FORMAT INTENSIFIED ON.
  ELSE.
    FORMAT INTENSIFIED OFF.
  ENDIF."

  WRITE: /01 MANDANTE,
             PEDIDO,
             SOCIEDAD.

ENDFORM.
```

SQL Open

```
- Ejemplo 1 - Tratamiento de una tabla declarada en tiempo de ejecución (sin
sentencia TABLES).
report yxxxxxxx.
data: nombre(10) type c value 'T9JCB',
      work_area(100).
start-of-selection.
select descr1 from (nombre) into work_area.
   write: /01 work_area.
endselect."

- Ejemplo 2 - Selección utilizando el wild card %
program yxxxxxxx.
tables: t9jcb.
select * from t9jcb
         where descr1 like 'REVIS%'.        ' Lo que empieza por REVIS
extract datos.
endselect.

- Ejemplo 3 - Selección utilizando los wild card _ y %.
program ysux0800.
tables: ekko.
select * from ekko
          where ebeln like '_4%'.              ' Lo que tenga un 4 en la segunda
pos.
write: /01 ekko-ebeln.
endselect.

- Ejemplo 4 - Utilizando la cláusula GROUP BY
report yxxxxxxx.
tables: t9jcb.
data: codi like t9jcb-codigo,
      acti like t9jcb-actividad.
start-of-selection."
select codigo actividad into (codi, acti) from t9jcb
       group by actividad codigo.             ' Agrupando y Ordenando
  write: /01 codi, acti.
endselect.
write: /01 sy-dbcnt.

- Ejemplo 5 - Selección rápida sobre una tabla interna (Inicializándola)
report yxxxxxxx.
tables: usr03.
data: begin of t1 occurs 1000,
         bname like usr03-bname,             ' código de usuario
         name1 like usr03-name1,             ' nombre PRIMERO
      end of t1.
start-of-selection.
select bname name1 into table t1 from usr03.  ' Refresh t1
sort t1.
loop at t1.
   write: /01 t1-bname,
              t1-name1.
endloop.

- Ejemplo 6 - Selección rápida sobre una tabla interna (SIN Inicializar)
report yxxxxxxx.
```

```
tables: usr03.
data: begin of t1 occurs 1000,
         bname like usr03-bname,             'código de usuario
         name1 like usr03-name1,             'nombre PRIMERO
      end of t1.
start-of-selection.
select * appending corresponding fields       'append t1
         of table t1 from usr03.
sort t1.
loop at t1.
   write: /01 t1-bname,
              t1-name1.
endloop.
```

- Ejemplo 7 - ROLLBACK

```
program yxxxxxxx.
tables: t9jcb.
start-of-selection.
select single for update * from t9jcb where codigo eq 'ZZZZZ'.
if sy-subrc eq 0.
  move 'primera modificaci¢n' to t9jcb-descr1.
  update t9jcb.
  commit work.
  move 'segunda modificaci¢n' to t9jcb-descr1.
  update t9jcb.
  rollback work.         ' Deshace solamente la segunda modificación
endif.
```

- Ejemplo 8 – Insertar un registro en una Tabla

```
report yxxxxxxx.
tables: tvarv.
start-of-selection.
  tvarv-name = 'FI_NUEVA_VARIABLE'.  'informamos los campos clave
  tvarv-type = 'P'.
  tvarv-numb = '0000'.
  insert into tvarv values tvarv.    'insertamos registro
```

- Ejemplo 9 – Average

```
report yxxxxxxx.
data:  cantidad type i,                        ' cantidad
       media    type p decimals 2,         ' Media"
       tecnic   like t9jcb-tecnic.
start-of-selection.
  select tecnic count( * ) avg( horasd )
         into (tecnic, cantidad, media)
         from t9jcb
         where tecnic eq sy-uname
         group by tecnic.
  endselect.
end-of-selection.
  write: / tecnic, cantidad, media.
```