TABLE CONTROL

EXPLICACION

El “table control” es un control que permite la visualización de los datos de una tabla y con la posibilidad de poder movernos por esos datos. Este control es una versión, digamos, más avanzada de su homogeno “step-loop”, ya que el “step-loop” no permite moverse por esos datos visualizados, como ventaja más significativa.
En una “table control” podemos insertar casi cualquier cosa, desde variables de programas, hasta tablas de diccionario pasando por tablas internas.

El aspecto es similar a de una hoja de calculo, un ejemplo claro de “table control” lo tenemos cuando introducimos datos a través de la vista de una tabla. Para ver como es un “table control” , ejecutamos la transacción “SM30” después introducimos el nombre de una tabla, la que queramos, y pulsamos el botón de visualizar. La pantalla que nos sale es la siguiente:

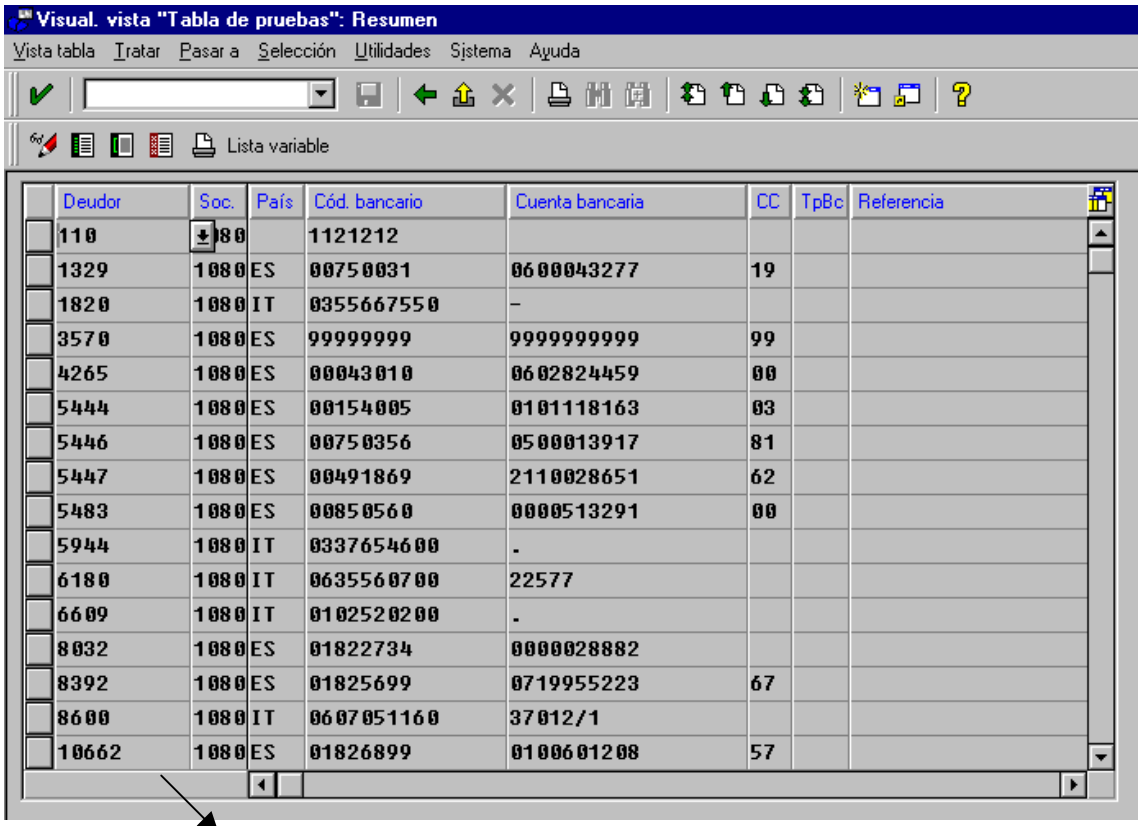


Table control

Este es un ejemplo de table control

DECLARACION

Una de las diferencias entre un “table control” y un “step-loop” es que el “step-loop” no se declará. La sintaxis para declarar una table control es la siguiente:

controls: nom-control type tableview using screen nº dynpro.
En “nom-control” es donde pondremos el nombre del “table control” que utilizaremos. Y en “nº dynpro” indicaremos en que dynpro aparecera el “table control”.

“Tableview” es una tabla con la siguiente estructura:

- Fixed_cols -> Tipo integer. Número inamovible de columnas at the left end of the table. Todas las columnas después de las bloqueadas son movibles y se pueden reordenar en la tabla.
- Lines -> Tipo integer. Número de filas que se pueden visualizar en la tabla.
- Top_line -> Tipo integer. Fila de la tabla, donde se empezará a visualizar.
- Current_line -> Tipo integer, fila actual que puede ser procesada dentro del loop. El indice de este campo es absoluto y se halla haciendo esta operación: TOP_LINE + SYTEPL – 1.
- Left_col -> Tipo integer. Las columnas de la izquierda no bloquedas, desde aquí el usuario puede realizar un scroll de las partes no bloquedas
- Line_sel_mode -> Tipo integer. Activa la selección de una línea. Valores: 0=Nada, 1=Solo una línea (Valor por defecto), 2=Multiples líneas.
- Col_sel_mode -> Tipo integer. Activa la selección de columnas. Valores: 0=Nada, 1=Una sola columna (Valor por defecto), 2)=Multiples líneas.
- Line_selection -> Tipo char 1. Indica si se visualiza la columna de selección de líneas. Esta columna es un simple check box que puede ser examinado desde programa. El sistema pone una “X” cuando el usuario hace un clic.
- H_grid -> Tipo char 1. Indica si se visualiza las lineas de separación horizontales.
- V_grid -> Tipo char 1. Indica si se visualiza las lineas de separación verticales.
- Cols -> Tipo tabla “tab_column” de 10 ocurrencias. Embedded internal table: one table entry for each column in the table.

Los campos de la tabla “tab_column” describen un campo simple and its column en la tabla de pantalla:

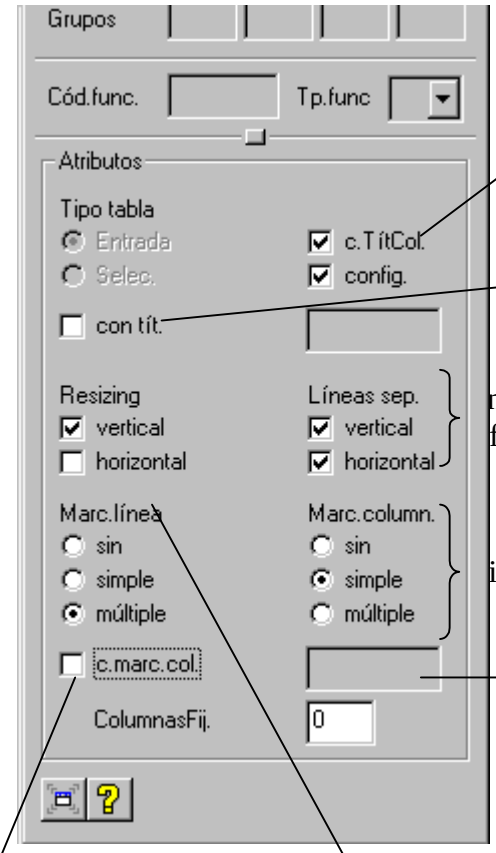
- Screen -> Tipo tabla “screen”. Embedded SCREEN structure: all the fields from a single row of the SCREEN system table.
- Index -> Tipo integer. Indica la posición actual de las columnas, en caso de que el usuario haya reordenado la sequencia de las columnas.
- Selected -> Tipo char 1. Puesto por el sisteme, vale “X” cuando el usuario hace click en una columna.
- Vislenght -> Tipo int1. Número de caracteres visibles del campo. El valor máximo es 255.

Los campos de “lin_sel_mode”, “col_sel_mode”, “line_selection”, “h_grid” y “v_grid” son más faciles manipularlos cuando estemos creando el “table control”.

El único campo que no puede ser inicializado desde programa es “current_line”.

PROPIEDADES DEL TABLE CONTROL EN LA SCREEN PAINTER

Aparte de los atributos que se pueden modificar en un programa, hay algunos que solo se pueden modificar en la “screen painter” y hay otros que se pueden cambiar en ambos sitios.
Para sacar los atributos del “table control”, si no lo esta (para saberlo hemos de fijarnos del recuadro blanco que aparece alrededor del “table control” si aparece es que esta seleccionado) haremos clic en el recuadro que aparece en la parte superior y presionaremos el botón “attributes” , entonces nos saldrá una ventana con los atributos en la parte de debajo de la ventana estan las opciones más interesantes, el recuadro será el siguiente:



Títulos de cabecera

Inidica si el “table control” tendrá título, por defecto no hay título

neas de separación vertical y horizontal, por fecto apareceran estas líneas

ipo de selección de columnas

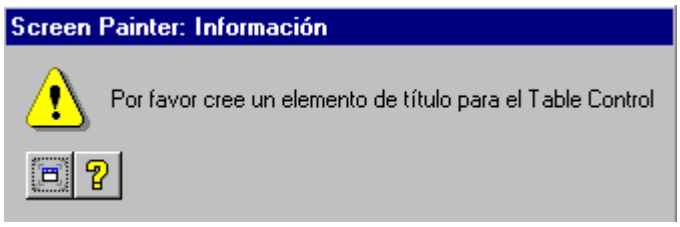
nombre al elemento de marca de líneas

Activar columna de selección de filas

Tipo de selección de filas o líneas

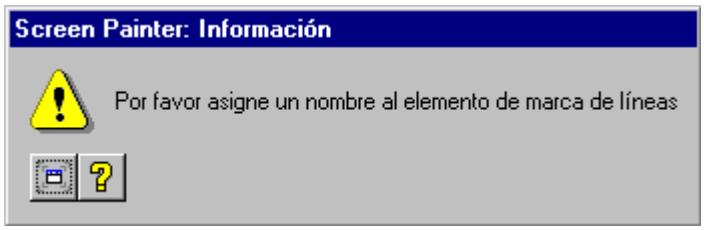
“Marc. Línea” y “Marc. Column” equivalen a los campos, de la tabla “tableview”, “line_sel_mode” y “col_sel_mode” respectivamente y el “c.marc.col” equivale al campo “line_selection”.

Cuando activemos el check box “con.tit” nos saldrá la siguiente ventana:



SAP nos informa que tenemos que hacer un “text” (el icono que aparece con una “T” dibujada, que esta situado a la izquierda de la pantalla).

Cuando activemos el check box “c.marc.col” nos saldrá esta otra ventana:



Esto indica que hemos de poner un nombre al elemento de marca de líneas, el campo esta situado a la derecha del check box.

EJEMPLO

Para ver como funciona un “table control”, realizaré un ejemplo completo de cómo se utiliza.

En este ejemplo se pedira en una pantalla la sociedad, con la sociedad introducida realizaré una búsqueda en la tabla “zztabpru10” el resultado de la busqueda lo guardaré en el “table control”. Una vez en el “table control” podremos: añadir, borrar, modificar registros también podremos ordenar el “table control” de forma ascendente o descendente y también se podrán realizar las funciones típicas como copiar, cortar y pegar.

La tabla de diccionario que utilizo para realizar las búsquedas se llama “ZZTABPRU10” que tiene la siguiente estructura:

ZZTABPRU10		**Tabla de pruebas**	
Clase de tablas	TRANSP	Clase de entrega	A
Modificada por	P0Z5005		
Opciones técnicas			
Clase datos	APPL0	Categoria tamaño	0

Estructura de tabla

Nombre	C	Tipo	Long.	Elem.Datos	Dominio	Texto
MANDT	X	CLNT	3	MANDT	MANDT	Mandante
KUNNR	X	CHAR	10	KUNNR	KUNNR	Número del deudor
BUKRS	X	CHAR	4	BUKRS	BUKRS	Sociedad
BANKS		CHAR	3	BANKS	LAND1	Clave de país del banco
BANKL		CHAR	15	BANKL	BANKL	Código bancario
BANKN		CHAR	18	BANKN	BANKN	Número de cuenta bancaria
BKONT		CHAR	2	BKONT	BKONT	Clae de control de bancos
BVTYP		CHAR	4	BVTYP	BVTYP	Tipo de banco colaborador
BKREF		CHAR	20	BKREF	CHAR20	Referencia para el banco/cuenta
XEZER		CHAR	1	XEZER	XFELD	Indicador: Existe autorización para domiciliación

Relaciones

Nom. campo	Tabla verif.	Cardinalidad	Cl.campos	Clave ext.	Texto
MANDT	T000	:	No especificado		
KUNNR	KNA1	:			
BUKRS	T001	:			
BANKS	T005	:			

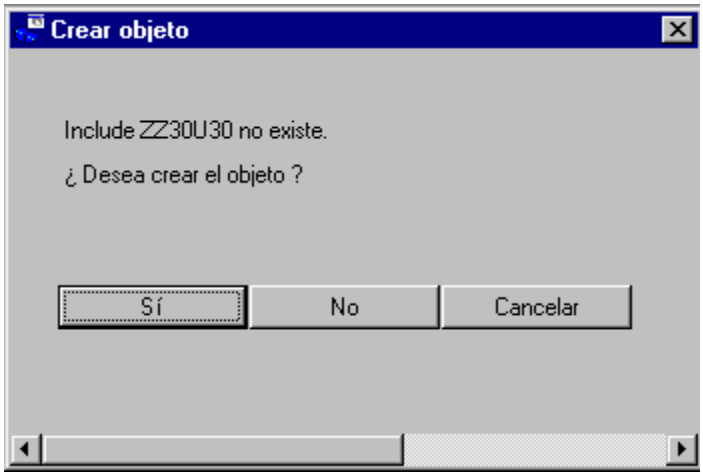
Podeis usar culaquier tabla, pero vigilar ya que nuestro programa puede borrar registros y por ello más vale hacerse una tabla propia cargarla de datos y hacer pruebas con ella.

Tengo que avisar que este ejemplo tiene lagunas a la hora de controlar cosas. Estas lagunas no han sido cubiertas para que el ejemplo no fuera más complicado y entendible por todo el mundo.

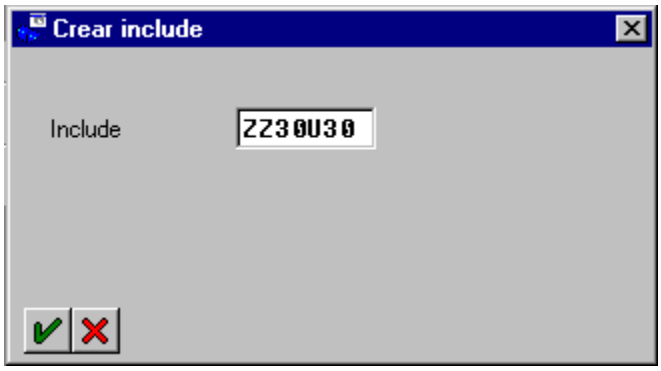
El programa principal constara de cuatro include dondes estarán: declaración de variables, los módulos del PBO, los módulos del PAI y las funciones. Separando cada parte consigo una claridad en la estructuración del programa. En los otros ejemplos no lo he hecho de esta forma porque los programa se entendian perfectamente sin necesidad de realizar includes.

PASO 1 (Crear el programa)

Primero hemos de crear el programa que será de tipo “modulpool” y segundo tenemos que asociarle una transacción a ese programa.
Una vez creado estas dos cosas tenemos que declarar las variables que vamos a utilizar en el programa. Como ello he dicho antes tendremos que crear primero el include donde estarán las variables, tablas, etc..
Para crear la include es sencillo, nos ponemos en una línea de nuestro programa y escribimos : “include zz30d30”, después haremos doble clic sobre el texto escrito y nos saldrá una ventana como esta, puede ser que antes nos salga una ventana pidiendonos si deseamos grabar el programa actual cosa que diremos que sí:



En esta pantalla le diremos que sí. Y nos saldrá otra pantalla parecida a esta:



Aquí de nuevo pulsaremos ENTER y nos saldrá la típica pantalla que nos sale cuando creamos un programa, una vez introducido los datos obligatorios iremos al texto fuente, pulsando el botón “Texto fuente” y la include ha de ser la siguiente:

```
*---------------------------------------------------------------------*
*   include zz30d30                                                   *
*---------------------------------------------------------------------*
tables: zztabpru10.
* defino el control "table-control" en el abap
controls: tabla type tableview using screen 2.
* tabla donde se guardaran los datos del resultado de la búsqueda.
data: tb like zztabpru10 occurs 0 with header line.
* tabla donde se guardaran los registros borrados.
data: tb_bor like zztabpru10 occurs 0 with header line.
* tabla donde se guardaran los movimientos de registros
data: tb_mov like zztabpru10 occurs 0 with header line.
* tabla donde guardaran los registros a mover, copiar. sera como el
* portapapeles de windows.
data: tb_por like zztabpru10 occurs 0 with header line.
* tablas para realizar el insert de línea
data: tb1 like zztabpru10 occurs 0 with header line,
      tb2 like zztabpru10 occurs 0 with header line.
```

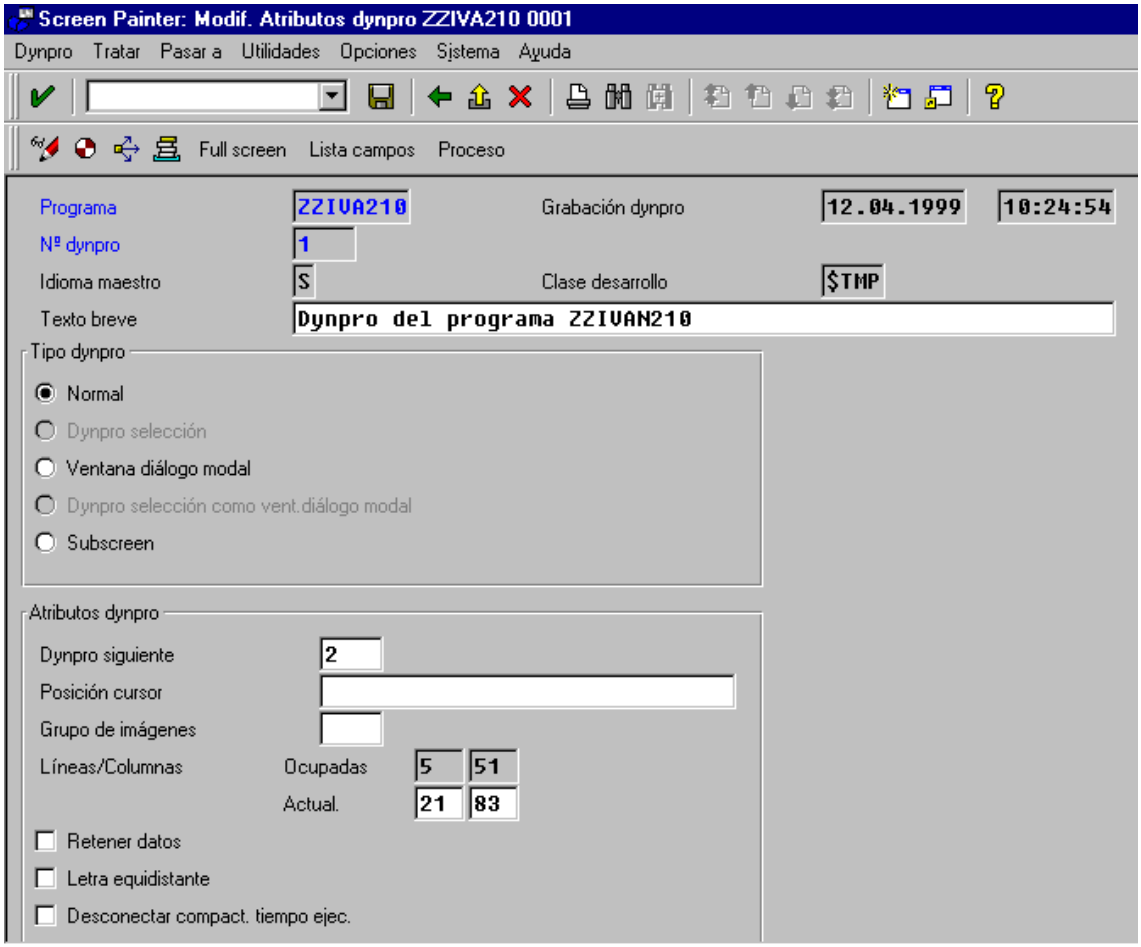
```
data: okcode(4), "control del teclado en la dynpro 1
      lineas type i, "variable para el loop.
* variable donde se pedirá la sociedad por pantalla
      wbukrs like zztabpru10-bukrs value '1080',
      selec(1) type c, " selección de registros en el table-control
      x(5) type n,
      z(5) type n,
      okcode1(4). "control de teclado en la dynpro 2.
```

Después grabaremos y volveremos al programa principal y lo generaremos.

Ahora tocaría crear la pantalla principal donde se pediría la sociedad.

PASO 2 (Pantalla principal)

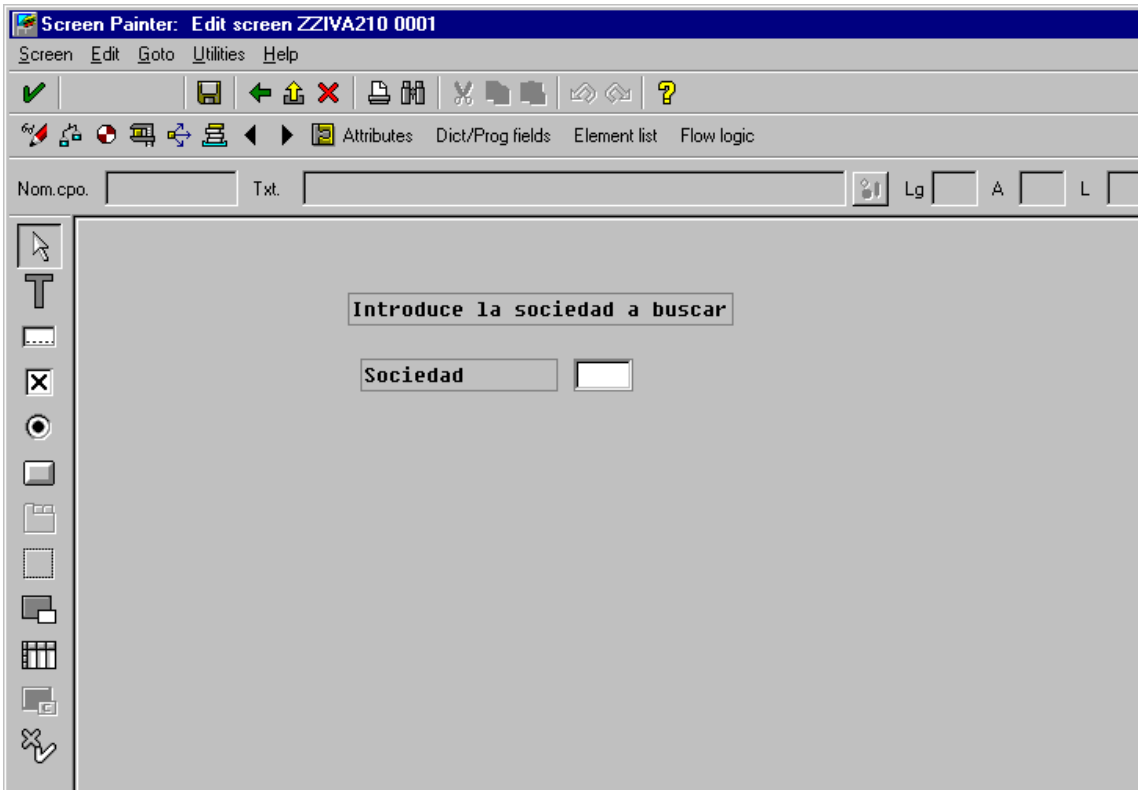
En este paso crearemos la pantalla donde se pedirá la sociedad que utilizaremos para realizar la búsqueda. El número de dynpro que tendrá esta pantalla será la "0001" y los atríbutos de esta dynpro quedarían así:



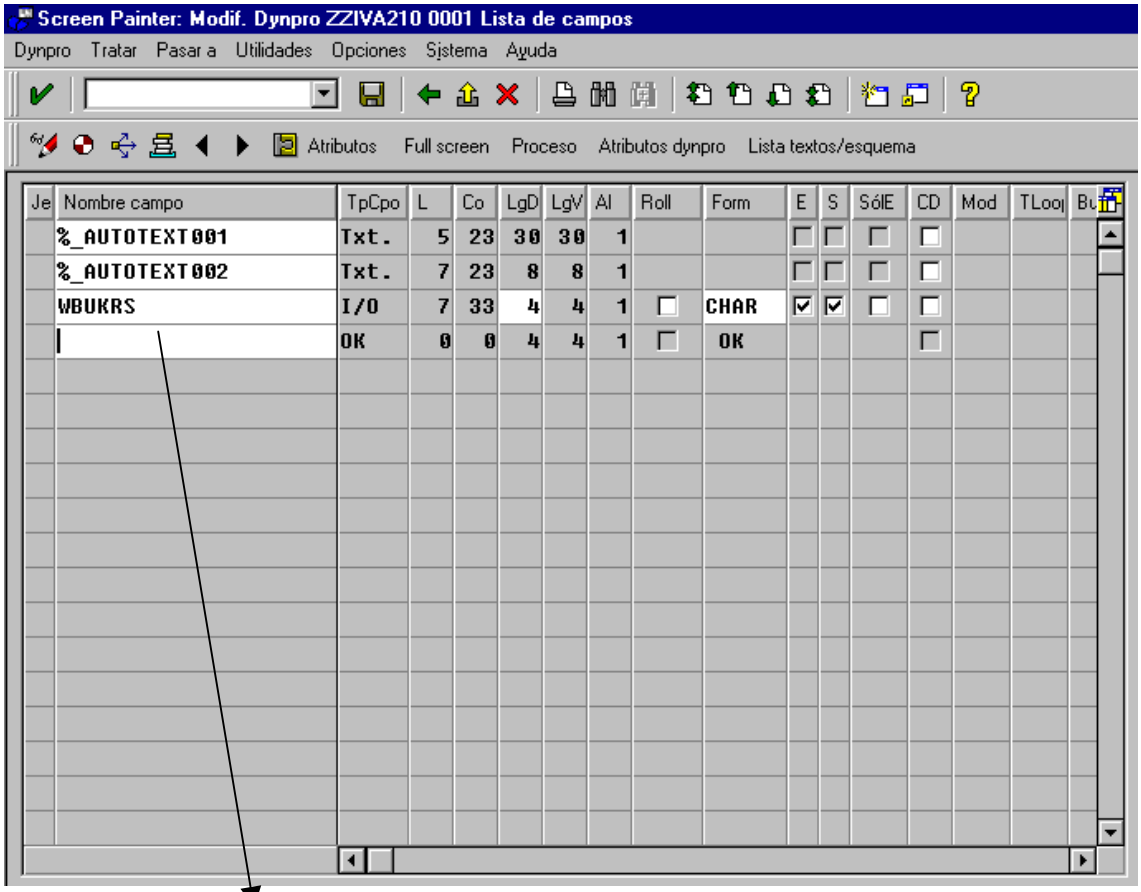
Fijaros que en el campo "Dynpro siguiente" pondremos el número de la siguiente dynpro que será la que tiene el número 2 (Que es donde mostraré el table control), esto

lo hago, por que cuando se pulse ENTER se me vaya automaticamente a la siguiente dynpro si la sociedad esta en la tabla.

Cuando hayamos grabado los atributos nos iremos a la “full screen” pulsando el botón “full screen”, La “full screen” que nos ha de quedar ha de ser esta:



Como veís he creado dos elementos de texto, que se realizan con el segundo botón de la izquierda (El que tiene una “T” dibujada) y después hemos creado objeto para introducir la sociedad que lo hemos traido a través del botón “Dict/Prog fields”, la variable que se trae es la “WBUKRS”.
Una vez introducidos los campos los grabaremos y generaremos la pantalla. Por último tenemos que poner la variable que nos servirá para controlar las teclas, botones, etc.. que pulsemos para ello volveremos atrás y nos iremos a “Lista campos” y al final de la lista habra un campo en blanco, hay escribiremos el nombre de la variable que servira para que nos controle los eventos de la pantalla, la variable será “okcode” que ha de quedar así:

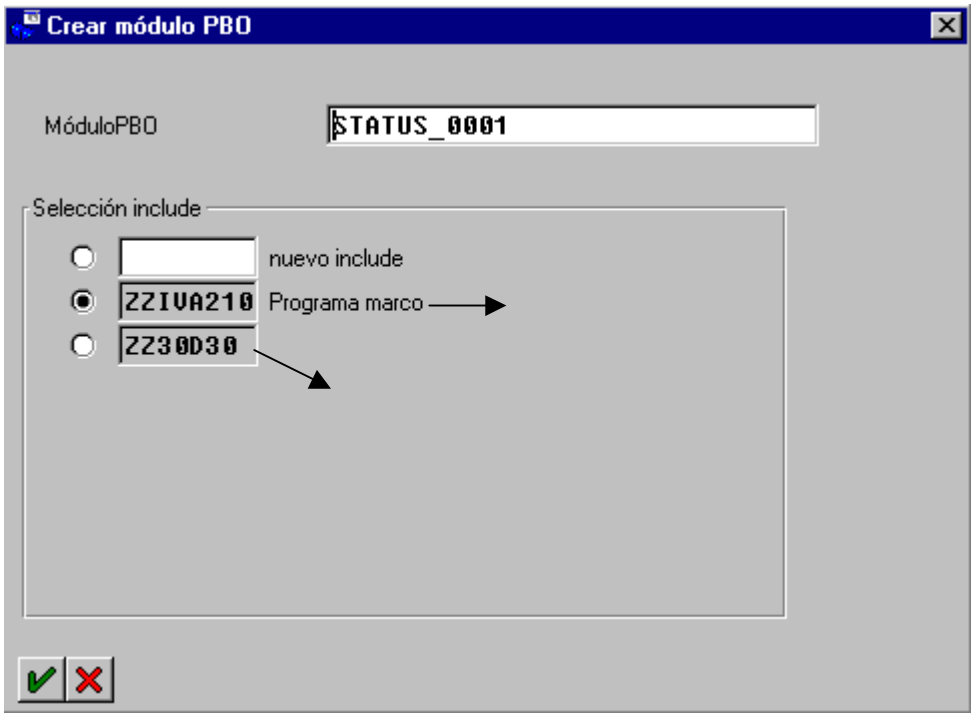


Campo de control del teclado

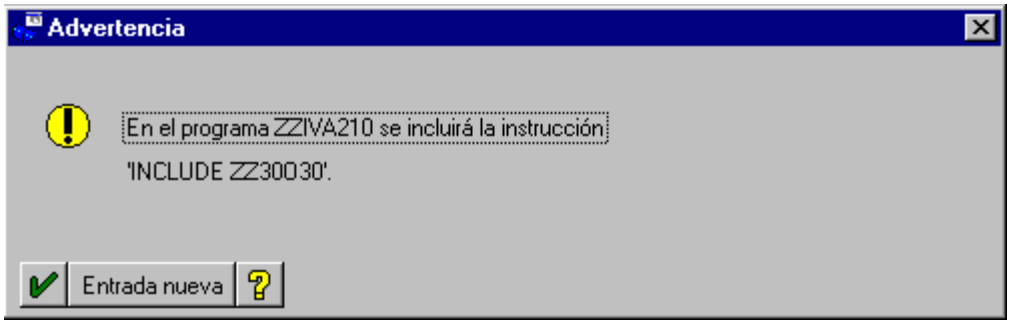
Como vemos al principio este campo esta blanco, solo hay uno por dynpro, siempre esta al final y el tipo de campo es “ok”. Como vemos es dificil no encontrarlo.
Nosotros le asignaremos este campo a la variable “okcode”. Después de hacer esto grabaremos y solo nos faltara crear la lógica de proceso, para ello pulsaremos el botón de “Proceso” y nos saldrá la siguiente pantalla:

```
000010 process before output.
000020 * MODULE STATUS_0001.
000030 *
000040 process after input.
000050 * MODULE USER_COMMAND_0001.
```

Quitaremos los dos asteriscos que estan en los “module” y los crearemos, primero crearemos el modulo del PBO (Process before output), haremos doble clic donde pone “status_001” y nos saldrá una ventana diciendonos que no existe este modulo y si lo deseamos crear, al cual responderemos que si. Un consejo: Si no hemos grabado nos volverá a salir otra ventana con la posibilidad de grabarlo cosa que tambien le diremos que si, después nos saldrá una ventana mostrandonos donde los deseamos crear.

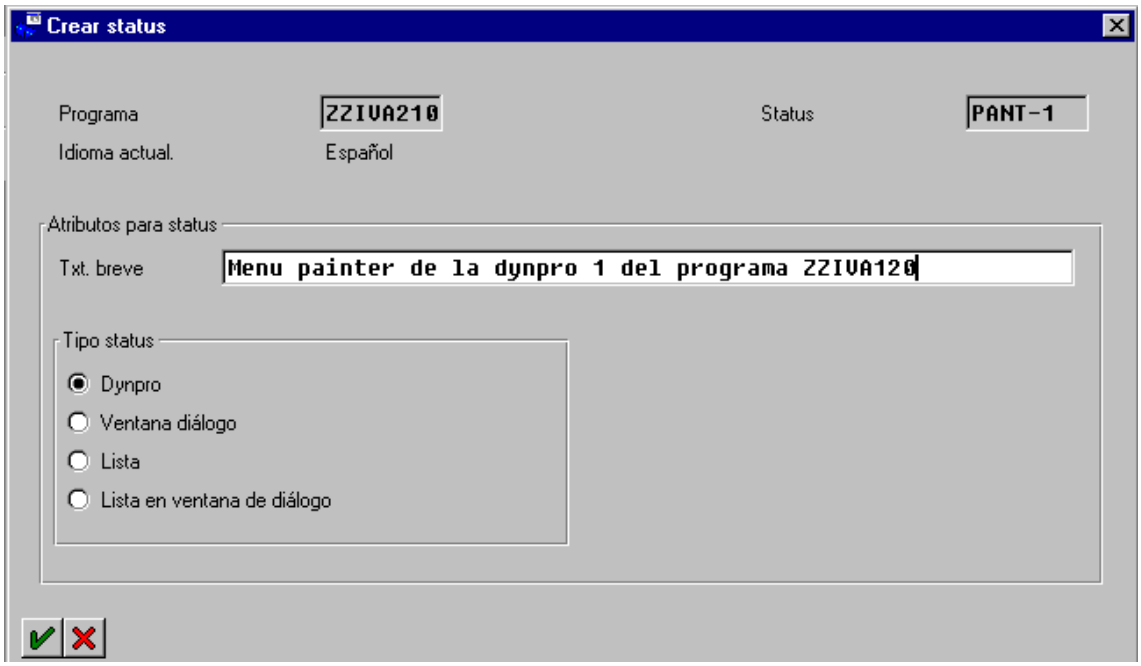


Como vemos podemos crearlos en tres sitios, uno es el programa principal, en la include donde estan declaradas las variables y por último tenemos la posibilidad de crearlo en una nueva include cosa que haremos. Para ello activaremos el radio button del campo “nuevo include” y escribiremos el nombre de la nueva include, en nuestro caso pondremos “ZZ30O30” si ya teneis esta include creada solo tendremos que aumentar la númeracion, por ejemplo: “ZZ40O40”. La “O” que hay en medio no la quitaremos para saber que esa include se guardarán los PBO del programa.
Cuando escribamos el nombre de la include pulsaremos ENTER y nos saldrá esta otra ventana:

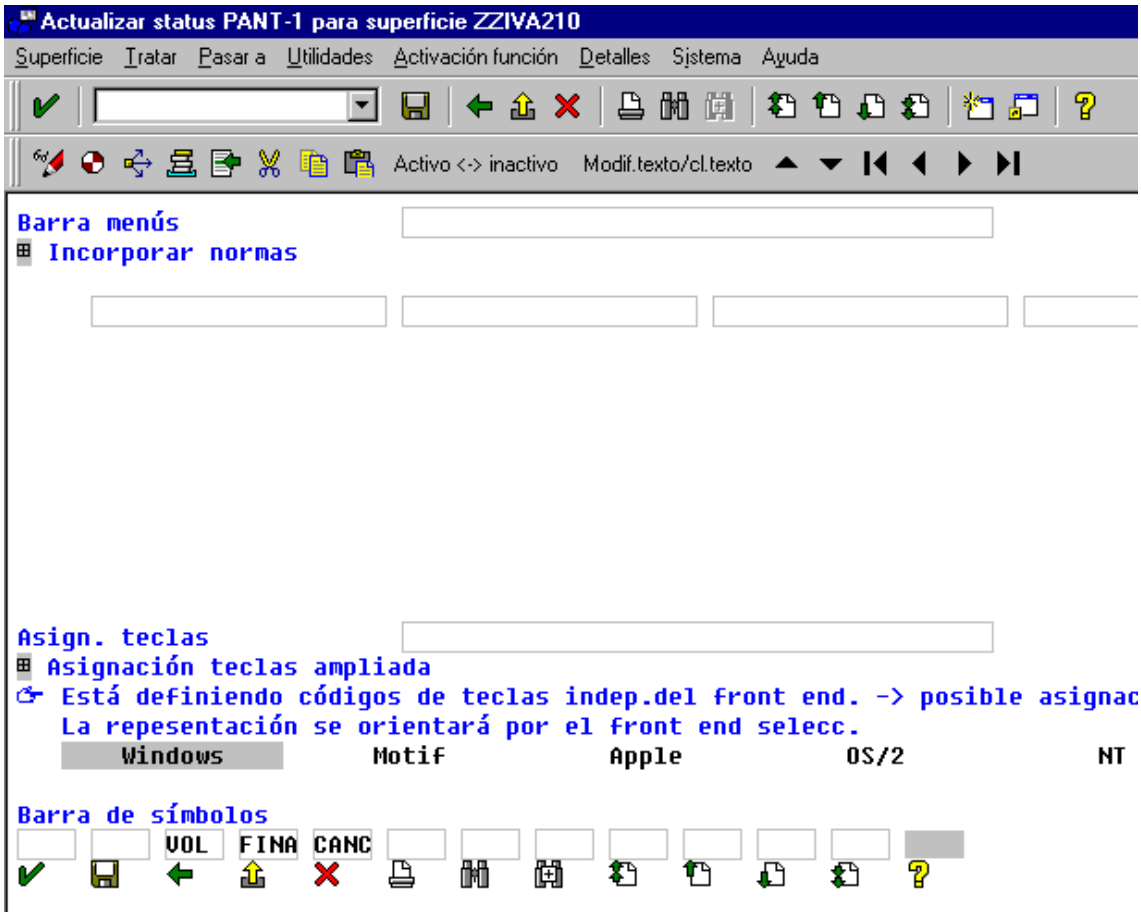


En esta pantalla nos dice que en el programa principal no existe la instrucción “include ZZ30O30”, si pulsamos ENTER nos pondrá la include en el programa principal y nos mostrará el include con el modulo nuestro.

Ahora quitaremos los astericos de las dos instrucciones que hay, y las ‘x’ de la orden “set pf-status” las sustituiremos por “pant-1”, el nombre hay que ponerlo en mayúsculas ya que sino SAP no dará error. Ahora haremos doble clic sobre “pant-1” y nos preguntara si lo deseamos crear le diremos que si y nos saldrá la ventana de “crear status” que ha de quedar como la siguiente:



Una vez puesto los datos pulsaremos “enter” o el botón de confirmar y nos saldrá la pantalla del menu painter, que tambien ha de quedar como la siguiente:



Una vez puesto los datos grabaremos y generaremos el menú painter.

Una vez creado el menú painter crearemos el título de la dynpro, para ello volveremos para atrás y en la orden “set titlebar” sustiuiremos las “x” por “t-1”, nombretan bien ha de estar en mayúsculas haremos doble clic sobre “t-1” y nos preguntara si deseamos crear el objeto le diremos que si y nos saldrá una pantalla donde introduciremos el título, la ventana ha de quedar como esta:

Aquí cada uno que ponga el título que quiera, una vez puesto pulsaremos el botón de “grabar” y ya nos cogera el título. El módulo quedara así:

```
000090 module status_0001 output.
000100   set pf-status 'PANT-1'.
000110   set titlebar 'T-1'.
000130 endmodule.                    " STATUS_0001  OUTPUT
```

Antes de volver a la lógica de proceso grabaremos y generaremos. En la lógica de proceso crearemos otro módulo en el PBO que servira para inicializar las tablas cada vez que se visualize la dynpro 1 y la variable que me controla las teclas y botones que se pulsen, lo inicializo para que no se quede el código anterior y así no me de problemas.
Este módulo lo pondremos antes del "module status_0001" y se llamará "inicializacion". Para crearlo escribiremos "module inicializacion" y haremos doble clic, nos preguntará si lo queremos crear, cosa que le diremos que si. Después nos saldrá la ventana de donde lo queremos crear, hay activaremos el radio button que sale la include "ZZ30O30" (O la que tengamos declarada para guardar los módulo del PBO) y le daremos al "enter" y nos saldrá el include donde escribiremos lo siguiente:

```
000190 module inicializacion output.
000200 clear tb. refresh tb. clear tb1. refresh tb1. clear tb2. refresh tb2.
000210 clear tb_bor. refresh tb_bor. clear tb_mov. refresh tb_mov.
000220 clear tb_por. refresh tb_por. clear okcode.
000230 endmodule.                  " INICIALIZACION  OUTPUT
```

Antés de volver a la lógica de proceso grabaremos y generaremos. La PBO de la dynpro uno quedaría así:

```
000010 process before output.
000020 module inicializacion.
000030 module status_0001.
```

Ahora crearemos el PAI, para ello volveremos atrás, quitaremos el asterisco del "module user_command_0001", haremos doble clic sobre el module y nos preguntará si lo deseamos crear, le diremos que sí. Entonces nos saldrá la ventana que nos pedirá donde deseamos crear el módulo, lo crearemos en nuevo módulo llamado "ZZ30I30", como antés si ya la tenemos creada aumentaremos la numeración pero la "I" del medio no la quitaremos para saber que esa include guardará todos los módulos de PAI. El proceso de creación del include es el mismo que hemos realizado antés para crear el include donde se guardarian los modulos del PBO.
Una vez creado solo tenemos que escrbir el código, que será este:

```
case okcode.
   when 'vol'.
     set screen 0. leave screen.
   when 'fina'.
     set screen 0. leave screen.
   when 'canc'.
     set screen 0. leave screen.
   when space.
```

```
* Busco la sociedad introducida y el resultado lo guardo en la tabla
* principal.
    select * from zztabpru10 into table tb where bukrs = wbukrs.
    if sy-subrc = 0.
* Mostrare a partir del primer registro
      tabla-top_line = 1.
* Indico al table control que muestre todos los registros de la tabla.
      describe table tb lines tabla-lines.
    else.
      message e000(zx) with 'sociedad no econtrada'.
    endif.
 endcase.
 clear okcode.
```

El “when space” lo utilizo para cuando se pulse “enter” se realiza las operaciones que haya debajo y después se carge la dynpro siguiente.

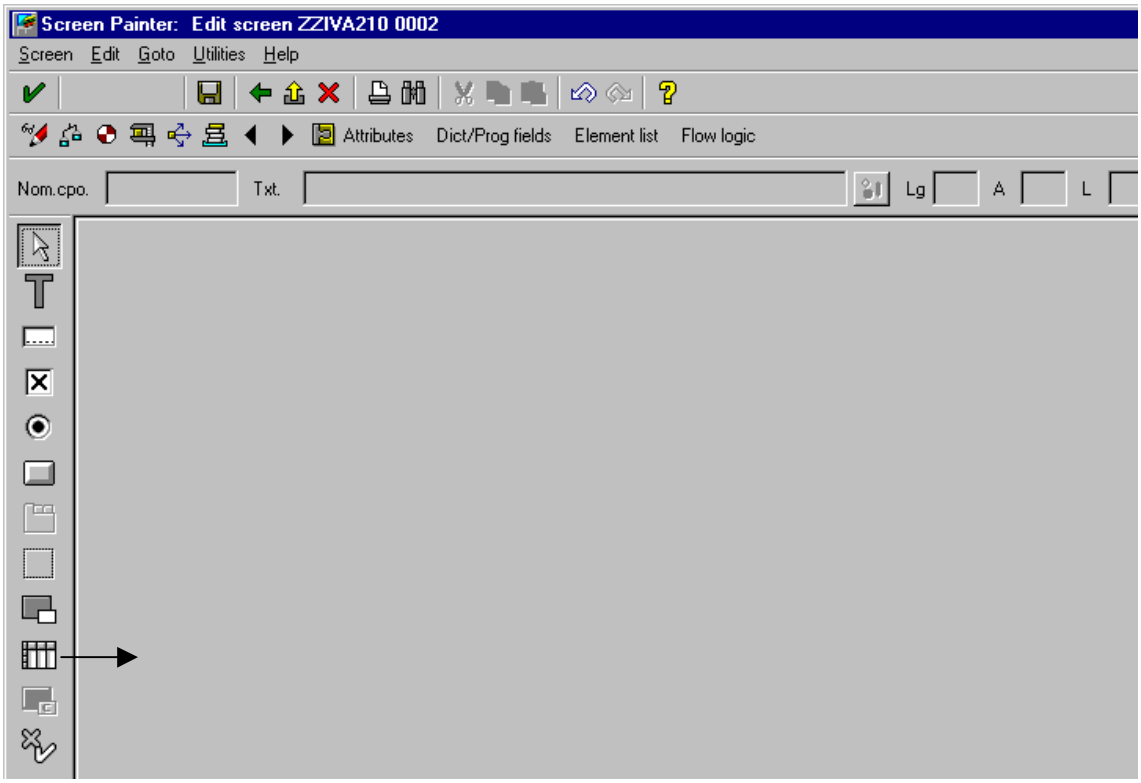
Una vez escrito todo esto grabaremos el programa y lo generaremos.
Volveremos atrás y lo generaremos, sino lo generamos. Nos fallara el programa.

Recordad: Cada vez que modifiquemos la “full screen”, lista de campos, lógica de proceso y los atributos de la dynpro hemos de generar la dynpro.

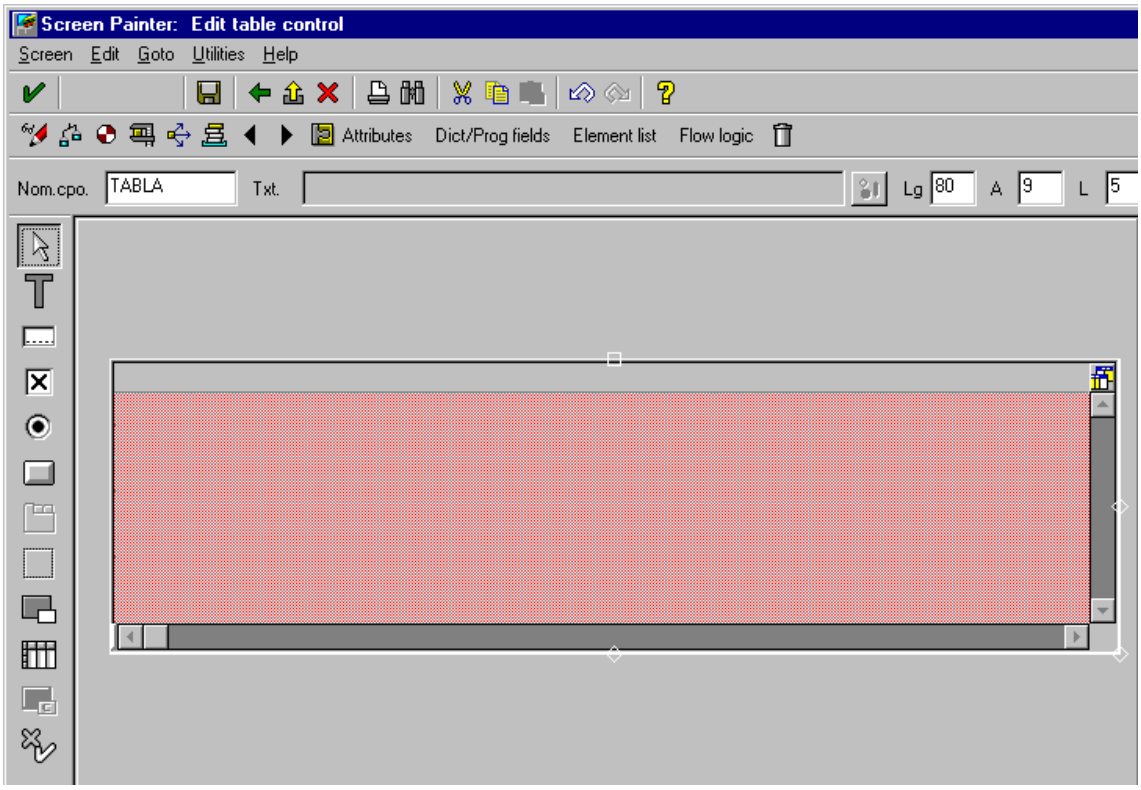
El siguiente paso es crear la pantalla del table control.

PASO 3 (Pantalla del table control)

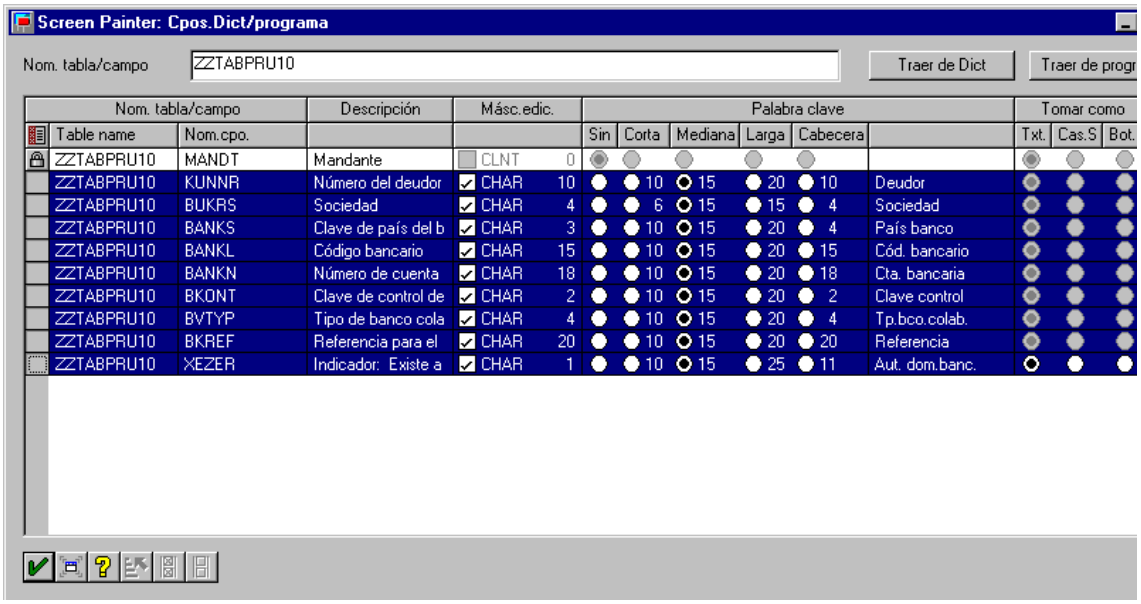
Para ello tendremos que crear una nueva dynpro, que tendrá el número 2 y asociarla al programa principal. Una vez creado y grabado nos iremos a la “full screen” donde pondremos el “table control”.
Inicialmente la pantalla que nos tiene que salir es esta:



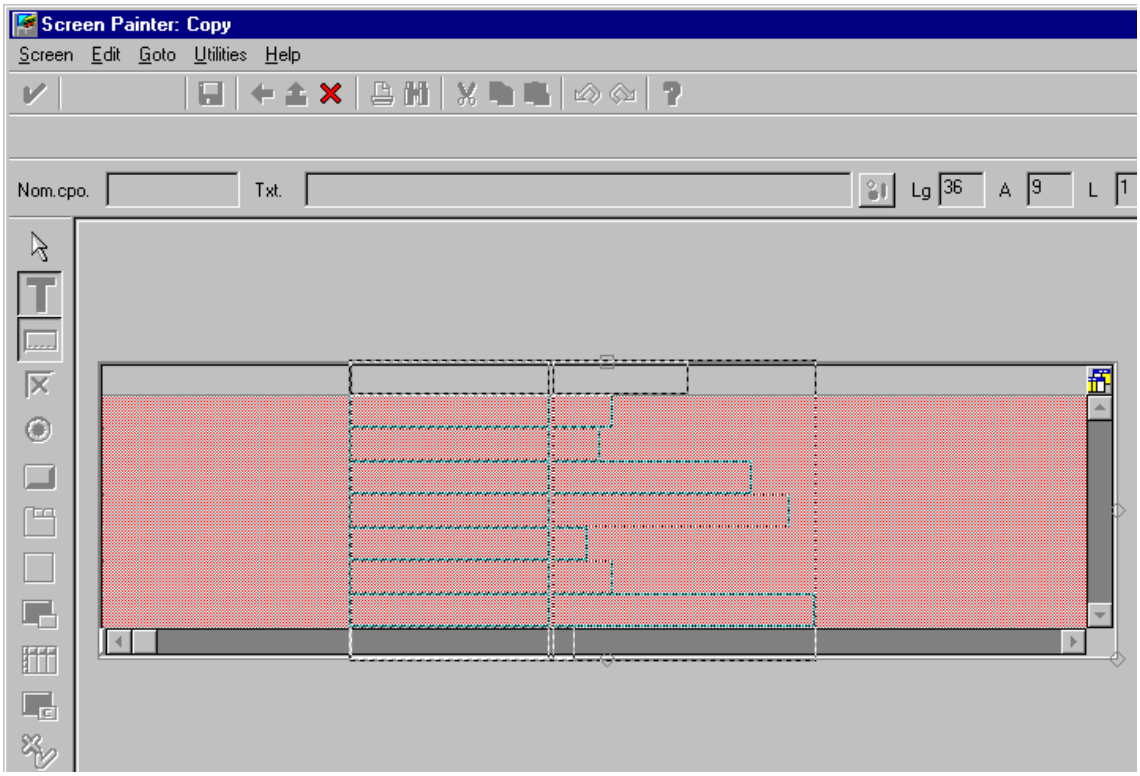
Para insertar un “table control” lo haremos con el botón en que aparece una hoja de calculo. Haremos clic sobre ese botón y lo colocaremos en la pantalla y el pondremos el nombre “tabla” en el campo que aparece arriba con el nombre “nom.cpo”. el resultado sería esta pantalla:



Lo segundo que hay que hacer es poner los campos de la tabla “zztabpru10” dentro del “table control”, para ello presionaremos el botón “Dict/Prog fields” entonces escribiremos el nombre de la tabla y presionaremos “enter” y seleccionaremos todos los campos el resultado sería este:



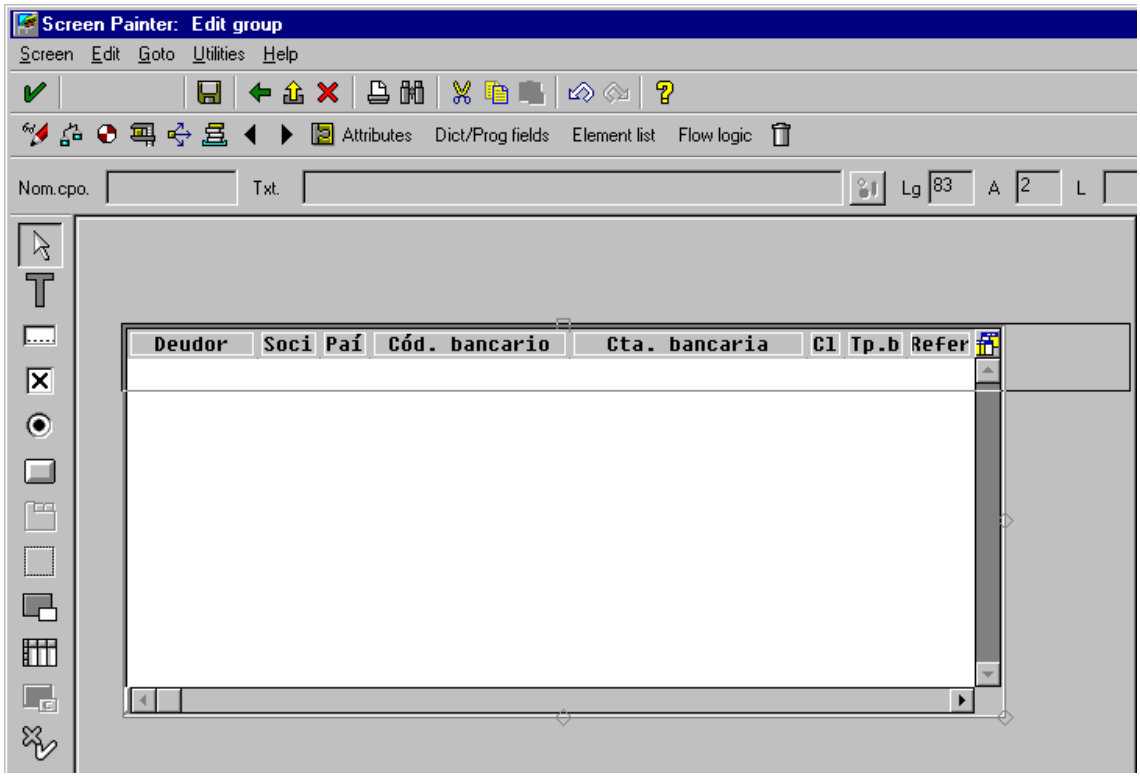
Una vez seleccionados presionaremos el botón de confirmar (esta abajo a la izquierda del todo) y nos saldrá esta pantalla:



Esto que nos sale es la silueta de los campos y sus descripciones, esta silueta la podemos insertar directamente en el “table control” o meterlo en el una zona de la pantalla hacer las operaciones que queramos y después mover campo a campo en el “table control”. Yo lo haré de la segunda forma que es más rápida y más comoda. Si lo hacemos de la segunda forma es hay que poner primero los campos y segundo la descripción de los campos, porque si lo hacemos al revés la descripción cogerá toda la anchura del “table control” y no podremos poner más descripciones. Si lo hacemos correctamente las descripciones se ajustarán al tamaño del campo que no tenga descripción.

Esto se puede hacer de dos formas uno cogiendolo directamente del programa (a través del botón “Dict/prog fields”) y insertandolo en el “table control”

Con el ratón moveros la sliueta de los campos hasta poner el primer campo dentro del table control para soltar esto haremos clic y automáticamente se nos colocará en el “table control”, el resultado puede ser este:



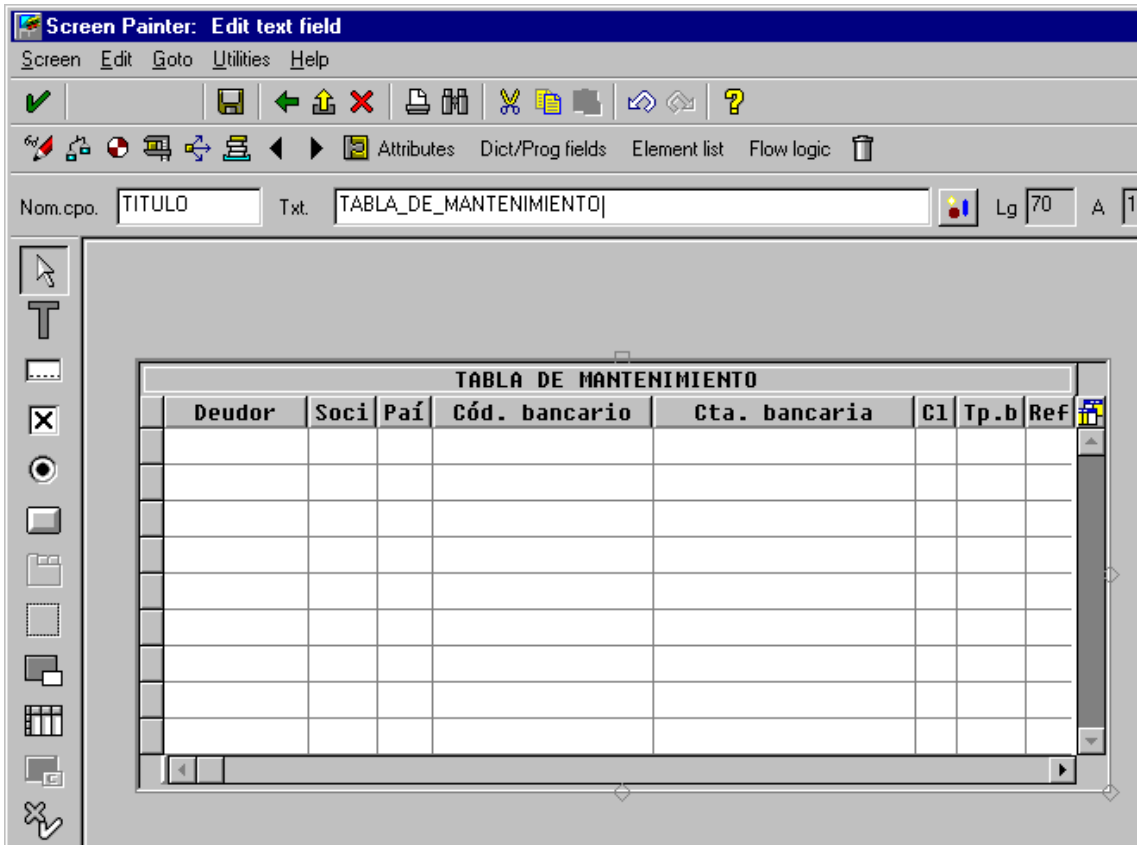
Digo que el resultado puede ser este, ya que a veces nos sale las lineas de separación horizontales y verticales automáticamente.
Lo que si hay que vigilar es que cuando insertamos los campos, SAP tiene todos esos campos seleccionados y si por ejemplo presionamos el botón de borrar nos borraría todos los campos, por ello, una vez isertados haremos un clic en cualquier sitio de la pantalla que no haya nada y se deseleccionaran los campos.

Ahora pondremos las líneas verticales y horizontales, para ello sacaremos la ventana de los atributos del “table control” y activaremos los check box que pone “vertical” y “horizontal”.
También indicaremos que las selección de las filas seán múltiples y las de columna ninguna. Para hacer lo primero activaremos el radio button, que esta debajo del texto “marc.línea”, que pone múltiple y para hacer lo segundo activaremos el radio button, que esta debajo del texto “Marc. Column”, “sin”.
Tambien le pondremos un título al “table control”, para ello activaremos el check box “con.tit” y nos saldrá una ventana de informacion la cual la cerraremos. Y veremos que el “table control” nos ha quedado así:

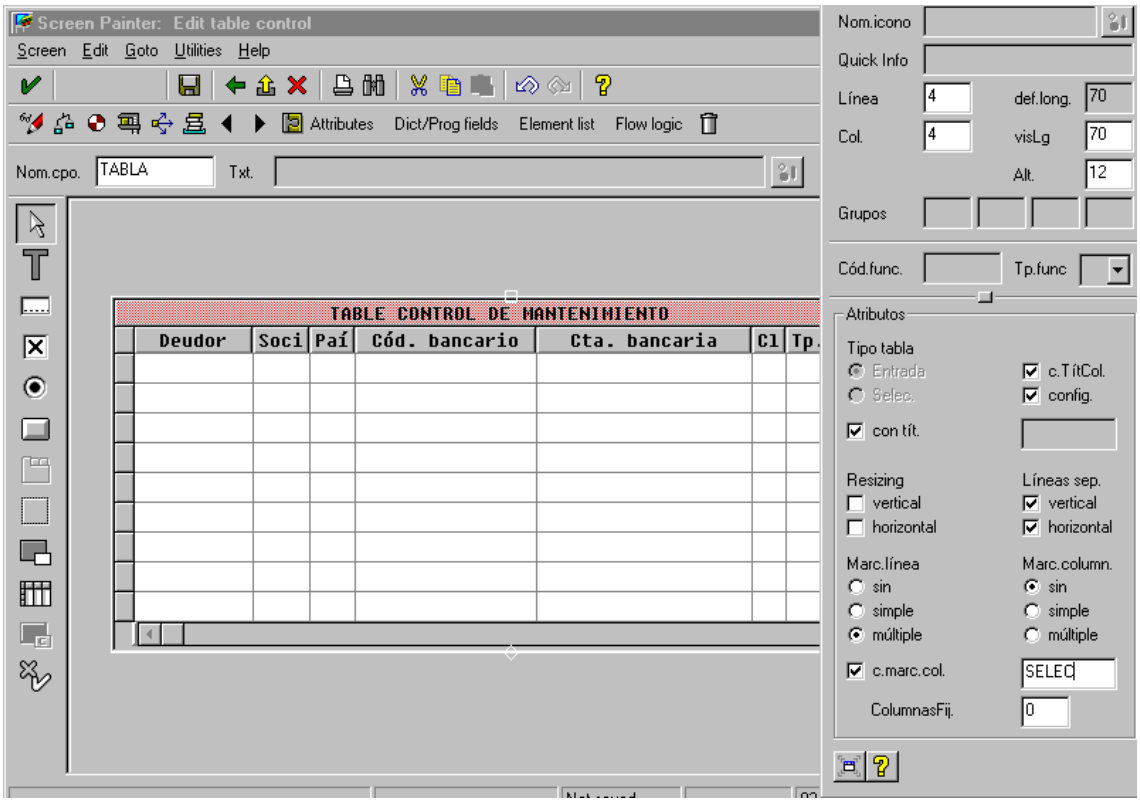
Título del “table control”

En el recuadro que aparece arriba es donde pondremos un “text” que contendrá el título del “table control”, también esto se puede hacer de dos formas insertando el “text” directamente en el “table control” o sino poniendo el texto en una zona de la pantalla escribir el título y después mover ese “text” al “table control”, yo lo haré de la primera manera. En los dos casos el “text” se ajusta a la anchura del “table control” tambien en los dos casos hemos de poner un nombre al “text” ya que sino dará error, si queremos modificar el título del “table control” durante el programa tenemos que declarar una variable con el mismo nombre que le hemos puesto al “text” de una la longitud igual o menor a la anchura del “table control”.

El resultado depués de insertar el “text” y ponerle el título y el nombre que nosotros queremos es el siguiente:

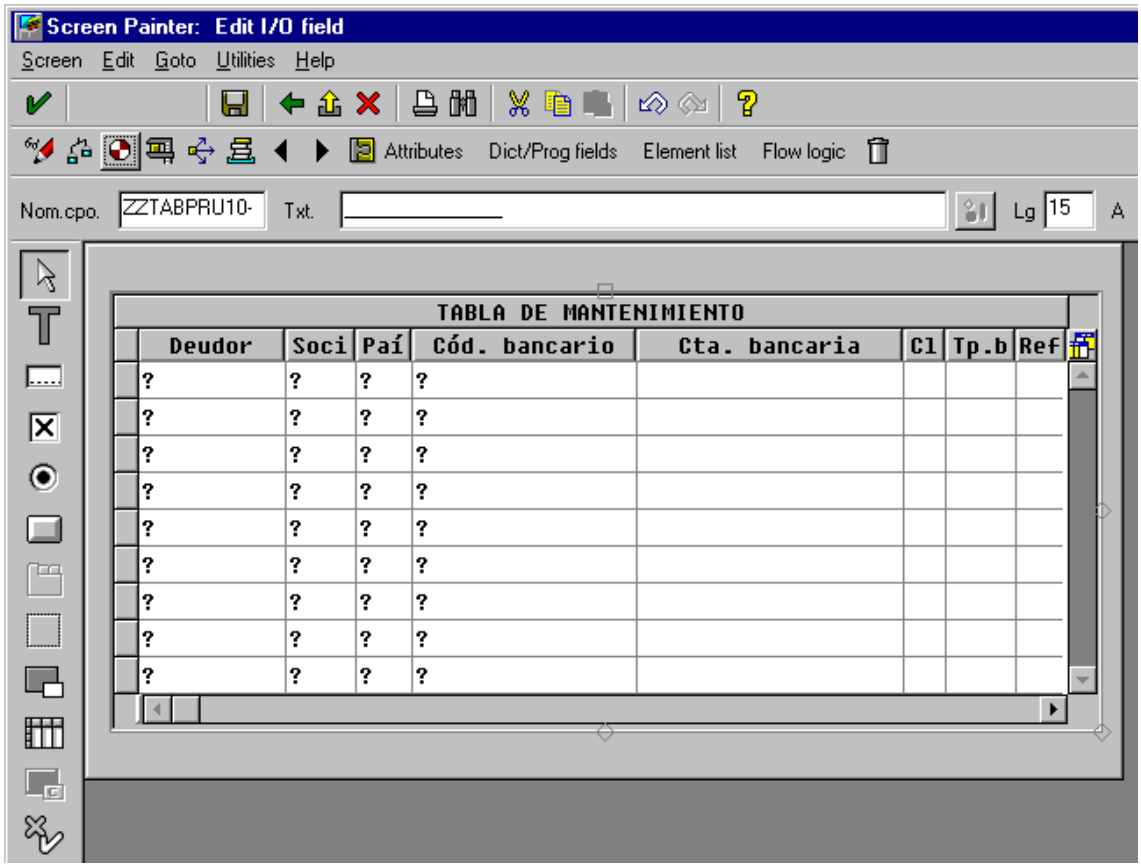


Ya solo nos falta poner la columna para poder seleccionar las filas, para ello activamos el check box “c.marc.col” y nos saldrá el recuadro de información lo cerraremos y en el campo de la derecha pondremos el nombre de la variable, el nombre será “selec”. El resultado seria el siguiente:



No tenemos que olvidarnos es el nombre de una variable que tenemos que declarar como un “char” de un carácter (nosotros si os acordais ya los hemos declarado al principio cuando hemos creado el include para poner las variables, tablas, etc.. que ibamos a utilizar) para poderlo controlar que fila ha sido seleccionada por el usuario.

Ahora vamos hacer que los campo de sociedad, deudor, cuenta y país sean obligatorios, es decir, que siempre se obligado introducir algo. Primero haremos el de la sociedad, para ello hacemos un clic sobre el campo “zztabpru10-bukrs” y después vamos a los atributos de ese campo (botón “attributes) una vez mostrados iremos a la pestaña “Programa” y activaremos el check box “Cpo.entr.obl” y entonces nos saldrá una “?” en la columna de la sociedad, ahora hacer los mismo con los campos “zztabpru10-kunnr” (Deudor), “zztabpru10-banks” (País) y “zztabpru10-bankl” (Cuenta bancaria) el resultado en pantalla será este:



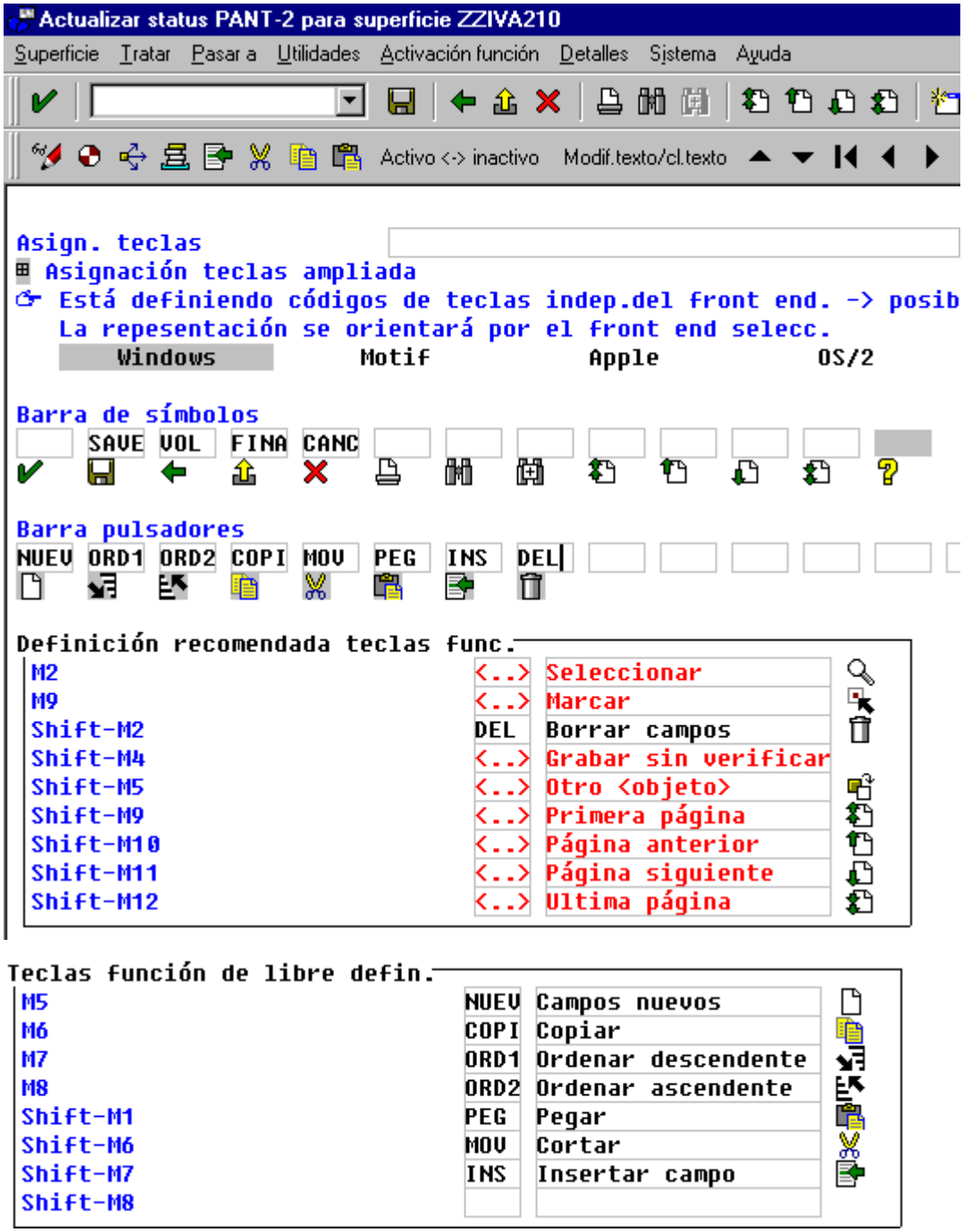
Como vemos en los cuatro campos obligatorios aparece el “?”.

Como último, grabaremos la dynpro pero no la verificaremos ni generaremos, por que dará error al faltar algunas cosas en la lógica de proceso, ya lo haremos más tarde.

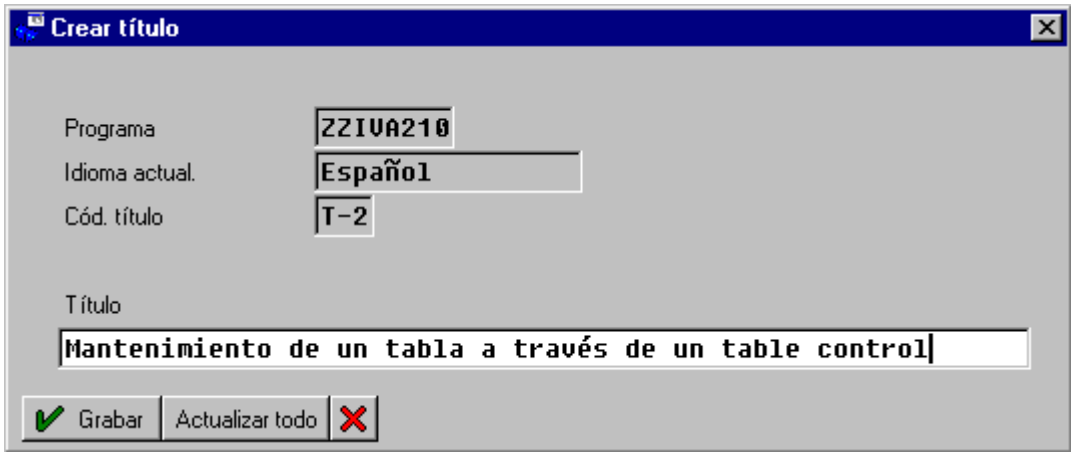
Una vez grabada iremos a la lista de campos de la dynpro y tendremos que asignarle una variable al campo que nos controla los botones, teclas pulsadas, etc.. Ese campo aparecere siempre el último de todos, como ya he explicado antés. El nombre que le pondremos será “okcode1”. Una vez puesto solo nos falta poner la lógica de proceso.

En la lógica de proceso, lo primero que haremos es quitar los asteriscos de los dos “module” y haremos doble clic en el “module” que esta en el PBO y nos saldrá la ventana que nos preguntará si queremos crear el módulo una vez respondido que sí, nos saldrá otra ventana de donde queremos crear el módulo nosotros activaremos el radio button del include “ZZ30O30” (O la que tengamos declarada para guardar los módulo del PBO) y pulsaremos “enter” para que nos cree el módulo.

En el módulo quitaremos los astericos de las dos instrucciones que hay, y las ‘x’ de la orden “set pf-status” las sustituiremos por “pant-2”, el nombre hay que ponerlo en mayúsculas ya que sino SAP no dará error. Ahora haremos doble clic sobre “pant-2” y nos preguntara si lo deseamos crear le diremos que si y nos saldrá la ventana de “crear status” donde pondremos el título al menú painter y después pulsaremos “enter” para que nos vaya a la pantalla de edición del menú painter, que ha de quedar como la siguiente:



Como vemos este ejemplo será bastante completo. Una vez creado todo esto tenemos que grabar el menú painter y generarlo y volveremos hacia atrás para ponerle el título a la dynpro. Como sabemos el título se pone a través de la orden “set titlebar”, por eso cambiaremos las “x” por “T-2”, recordar las letras siempre en mayúsculas, y haremos doble clic sobre “T-2” y nos saldrá la ventana que nos preguntará si los deseamos crear, le diremos que sí y le pondremos nuestro título, en nuestro caso quedará lo siguiente:



Pulsaremos el botón de grabar para que nos cree el título. Después escribiremos la siguiente instrucción: “clear okcode1”, como en la dynpro 1, con esto limpiare la variable de control de las teclas y botones pulsados, para evitar que el código anterior me de problemas. El Módulo quedaría así:

```
000290 module status_0002 output.
000300   clear okcode1.
000310   set pf-status 'PANT-2'.
000320   set titlebar 'T-2'.
```

Antes de volver a la lógica de proceso, grabaremos y generaremos.

En la lógica de proceso escribiremos la primer parte del código para pasar los datos de la tabla interna al “table control”, este código lo escribiremos el PBO (process before output) y antes del “module status_0002”. El PBO ha de quedar así:

```
000010 process before output.
000020  module status_0002.
000030  loop at tb with control tabla cursor lineas.
000040       module visualizar.
000050  endloop.
```

El parámetro “with control” indicamos que los campos que visualizemos se controlaran a través de programa mediante un “table control” , nuestro “table control” se llama “tabla”. Y el parámetro “cursor” servira para guardar el indice del “loop”, la variable que servira para guardar el índice ha de ser de tipo integer o como la variable del sistema “sy-index”.

Y el module visualizar quedara así:

```
000400 module visualizar output.
000410 move-corresponding tb to zztabpru10.
000420 endmodule.                    " VISUALIZAR  OUTPUT
```

Ya sabemos como crear un modulo, haciendo doble clic sobre “visualizar”, decirle que si cuando nos diga si lo deseamos crear y por último le indicaremos que lo queremos crear en la “include ZZ30O30” o la include que tengamos declarada para guardar los módulos del PBO. Una vez escrito este modulo lo grabaremos y generaremos. Os podeis preguntar que por una instrucción no hace falta crear un módulo, pues si que hace falta ya que si ponemos el “move-corresponding...” en la lógica de proceso nos dará error, más concretamente este:

```
Statement "MOVE-CORRESPONDING" is not defined. Please check your spelling.
```

Este error ocurre por que solo podemos poner unas pocas instrucciones en la lógica de proceso como por ejemplo: “loop”, “chain”, etc..

Antés de volver a la lógica de proceso grabaremos y generaremos el módulo:

Ahora en la lógica de proceso hemos de crear el segundo “loop”, este se ha de poner en el PAI (process after input), el PAI quedaría de la siguiente forma:

```
000080 process after input.
000090 loop at tb.
000100
000110 endloop.
000120  module user_command_0002.
```

Este “loop” al principio no tiene ninguna utilidad, pero es obligatorio ponerlo ya que como en el “step-loop” hay que poner dos “loops” uno en el PBO y otro en PAI. El “loop” del PAI lo utilizaré para controlar los registros que el usuario toca. Pero esa parte ya la haremos después.

Ahora crearemos el “module user_command_0002”, para ello haremos doble clic sobre el “user_command_0002”, nos preguntara si lo queremos crear y después nos pedira en que include los deseamos crear, nosotros la crearemos en la include “ZZ30I30” o en la include que tengamos creada para guardar las includes del PAI.
El modulo quedará así.

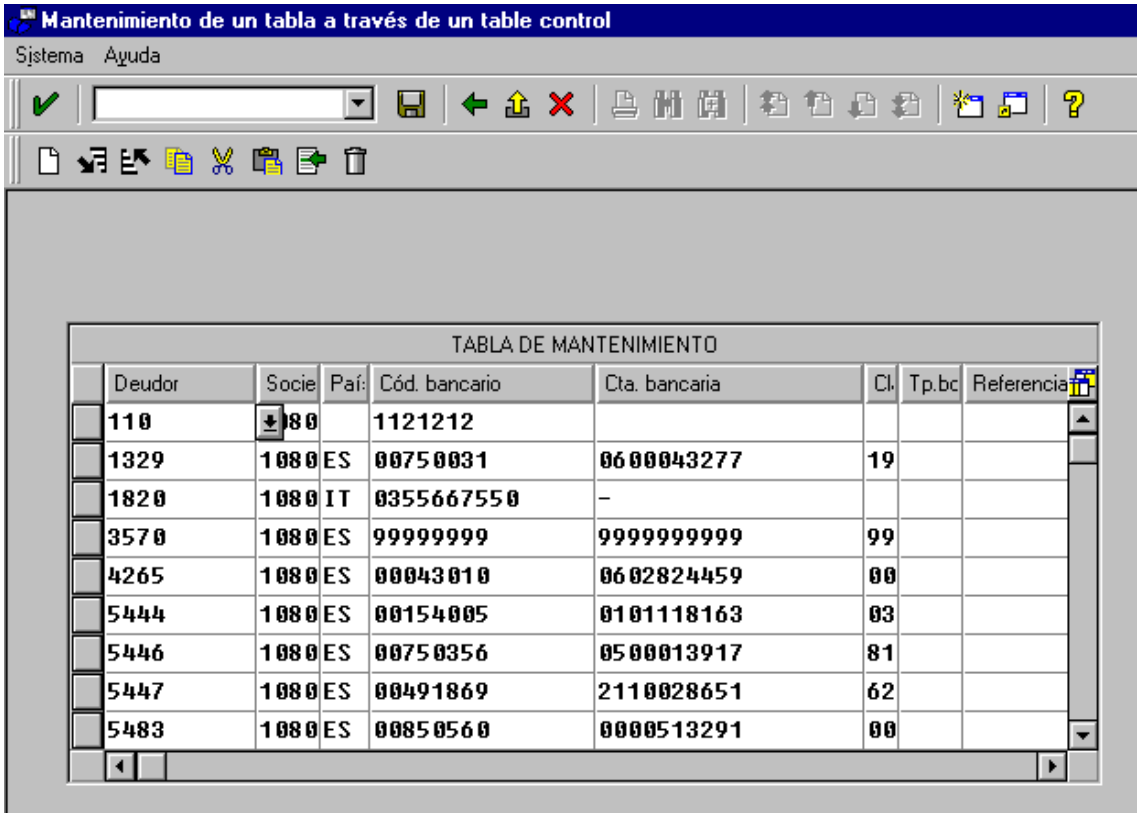
```
000370 module user_command_0002 input.
000380   case okcode1.
000390     when 'VOL'.
000400       set screen 1. leave screen.
000410     when 'FINA'.
000420       set screen 1. leave screen.
000430     when 'CANC'.
000440       set screen 1. leave screen.
000450 endcase.
000470 endmodule.                    " USER_COMMAND_0002  INPUT
```

Este módulo controlo los botones necesarios para volver a la dynpro anterior (que tiene el numero uno), más tarde ya añadiremos el control de los botones que me faltan.

Antes de volver a la lógica de proceso grabaremos y generaremos el módulo. Y la lógica de proceso tambien la generaremos.

PASO 4 (Mini-prueba)

En este caso haremos una prueba de cómo queda nuestro “table control” hasta el momento. Para ellos ejecutaremos la transacción que hemos creado para nuestro “modul-pool”, en mi caso he creado la “ZIV2” por ello pondré en el campo comandos “/NZIV2” y nos saldrá la pantalla principal donde se nos pedirá la sociedad que inicialmente vale “1080” (este valor por defecto lo hemos puesto cuando hemos declarado la variable “wbukrs”), vosotros poner una sociedad que exista, pulsar “enter” y el resultado es este:



Como vemos el “table control” tiene barras de movimiento verticales y horizontales que nos permite ver los datos que hay más abajo y los campos de la derecha que no se ven.

Una vez visto la prueba hay que saber que registro modifica o que fila seleccionada el usuario.

PASO 5 (Control del “table control”)

Para saber que registro o fila selecciona el usuario se utiliza la orden “chain”, como ya hemos visto esta orden controla que campo o campos se modifican. Esta orden se pondriá tendro del “loop” que esta situado en el PAI de la segunda dynpro, el resultado sería este:

```
process after input.
loop at tb.
     chain.
          field: zztabpru10-kunnr, zztabpru10-bukrs, zztabpru10-banks,
                 zztabpru10-bankl, zztabpru10-bankn, zztabpru10-bkont,
                 zztabpru10-bvtyp, zztabpru10-bkref, zztabpru10-xezer,
                 selec.
          module linea on chain-request.
     endchain.
endloop.
 module user_command_0002.
```

Dentro del “chain” he puesto todos los campos del “table control” incluido el campo de elemento de marca de líneas, que se llama “selec”.

La opción “on chain-request” en el “module” significa que entrara if the user entered at least one value in the chain.

Lo bueno que tiene poner el “chain” dentro del “loop” es que si el usuario modifica o selecciona cien registros, cien veces entrará en el módulo “linea” y por tanto podremos controlar todos los registros que el usuario hay modificado y seleccionado el usuario. Si os acordais al crear el “table control” hemos dicho que se pueden seleccionar múltiples filas.

Dentro del “chain” entrará en el módulo cuando pulsemos un botón, el “enter” o alguna tecla de función que tengamos definida.

Después de introducir esto crearemos el módulo, acordaros que irá en el include “ZZ30I30” o el include que tengamos creado para poner los módulos del PAI. El módulo tendrá las siguientes instrucciones:

```
* si el registro esta seleccionado, lo añado a la tabla de borrado
  if selec = 'x'.
    move-corresponding zztabpru10 to tb_mov.
    append tb_mov.
  else.
* tabla-current_line devuelve la posicion donde esta el registro en la
* tabla.
* Me posiciono en el registro de la tabla principal que hay sido
* modificado
    read table tb index tabla-current_line.
* coloco por defecto el mandante
    zztabpru10-mandt = sy-mandt.
* si la busqueda ha tenido éxito borro el registro para añadir el nuevo
* para evitar que los registros anterior se quede en la tabla.
* si no encuentro el registro lo añado.
    if sy-subrc = 0.
* solo se borrara el registro y se añadira a la tabla de borrado,
```

```
* siempre y cuando el deudor y la sociedad no sean espacios
* en blanco. si esos dos campos están en blanco quiere decir que estoy
* añadiendo un nuevo registro en medio del "table control".
      if  tb-kunnr ne ' ' and tb-bukrs ne ' '.
        perform buscar_soc_deudor_cta.
        if sy-subrc ne 0.
* añado el registro modificado en una tabla, esta tabla guardare los
* registros modificados y que borraran de la tabla principal
          move-corresponding tb to tb_bor.
          append tb_bor.
* pongo los nuevos datos al header y modifico el registro de la tabla
          move-corresponding zztabpru10 to tb.
          modify tb index tabla-current_line.
        else.
* solo muestro el mensaje de error cuando los registros el modificado,
* tabla-current_line y el encontrado, sy-tabix, sean diferentes
* si son iguales quiere decir que se ha vuelto al original
          if sy-tabix ne tabla-current_line.
            message e000(zx) with 'datos repetidos'.
          endif.
        endif.
      else.
* primero compruebo de que los datos no estan repetidos
        perform buscar_soc_deudor_cta.
        if sy-subrc ne 0.
* modifico los datos de la tabla
          move-corresponding zztabpru10 to tb.
          modify tb index tabla-current_line.
        else.
       message e000(zx) with 'el deudor ya tiene la cuenta introducida'.
        endif.
      endif.
    else.
      perform buscar_soc_deudor_cta.
      if sy-subrc ne 0.
* si no encuentro el registro en la tabla, lo añado
        move-corresponding zztabpru10 to tb. append tb.
      else.
       message e000(zx) with 'el deudor ya tiene la cuenta introducida'.
      endif.
    endif.
  endif.
```

El funcionamiento es el siguiente: Primero compruebo que el registro, que el usuario ha tocado, esta seleccionado, para saberlo miro si la variable “selec” (que es quien me controla la fila seleccionada), si esta seleccionada “selec” tendrá una “x” mayúscula, si la fila esta selecciona guardo el registro en la tabla “tb_mov”.
Ahora si no esta seleccionada me posiciono en el número de registro de la tabla principal que ha sido modificado, entonces controlo si he me he posicionado bien o no.

Ya que en un “table control” puede quedar registros en blanco al final donde podemos poner nuevos registros en esa zona, este seria el caso donde el indice del “table control” sería superior al número de registros que tiene la tabla interna y entonces daría error. Para controlar si me posiciono o no utilizaré la variable del sistema “sy-subrc” si me devuelve cero es que no hay errores y cualquier valor diferente a cero quiere decir que se ha posicionado mal. Si da error, quiere decir que el usuario ha introducido registros nuevos al final del “table control” y por lo tanto primero buscaré que los datos introducidos al final no están introducidos con anterioridad, si son correctos los añadiré a la tabla interna y le indicare al “table control” el número de registros que hay en la tabla principal, ya que hemos aumentado el tamaño de la tabla y en ciertos casos puede ser que los registros que añadimos al final no salgan.
Si no me da error al posicionarme en el registro hago lo siguiente: Miro si la sociedad y el deudor no sean espacios en blanco, si están en blanco quiere decir que estoy insertando un registro nuevo en medio del “table control” por lo tanto primero busco que los datos no esten repetidos si lo estan muestro un mensaje de error, y si no lo están repetidos muevo el registro nuevo al header de la tabla prinicipal y lo modifico.
Si la sociedad y el deudor no estan en blanco quiere decir que estoy modificando un registro lo que hago es primero buscar si los datos modificados estan repetidos si no lo están muevo el registro modificado (el que esta en la tabla principal) a la tabla de borrado, ¿Por que lo muevo a la tabla de borrado? Pues porque los registros que he modificado en la tabla principal no se modificarian en la de diccionario al no he encontrarse esos registros (los modificados) en la tabla principal, por ello los registros que se modifiquen los guardo en una tabla y cuando vuelque los datos de la tabla principal a la de diccionario, primero borrare los registros modificados de la tabla de diccionario y luego insertaré el resto de registros.. Si estos datos modificados están repetidos puede ser por dos motivos: 1) Que el usuario ha vuelto ha poner los datos que habian antes y que aún no se han puesto en la tabla principal, y 2) Realmente ha puestos unos datos que ya existen, Si ocurre la primera opción lo que hago es ignorar el error y si ocurre la segunda muestro el error.

Cuando hayamos escrito este módulo lo grabaremos y generaremos. Ahora volveremos a la lógica de proceso y la generaremos.

Ahora solo nos toca acabar de poner las instrucciones al módulo “user_command_0002”.

PASO 5 (Los últimos procesos)

En este paso pondremos lo que nos falta en el módulo “user_command_0002”, las cosas que nos falta son los botones que hemos creado para ordenar, modificar, copiar, pegar, etc... los registros del “table control”. El código de este modulo quedaría así:

```
case okcode1.
   when 'vol'.
     set screen 1. leave screen.
   when 'fina'.
     set screen 1. leave screen.
   when 'canc'.
```

```
      set screen 1. leave screen.
    when 'del'.
     perform borrar_registros.
    when 'save'.
     perform grabar_datos.
    when 'ins'.
     perform insertar_linea.
    when 'nuev'.
      clear tb.
      append tb.
      describe table tb lines tabla-lines. " numero de registros totales
* me posiciono en la último registro del table control
      tabla-top_line = tabla-lines.
    when 'mov'.
     perform guardar_portapapeles.
    when 'copi'.
     perform guardar_portapapeles.
    when 'peg'.
     perform pegar_lineas.
    when 'ord1'.
      sort tb descending by kunnr bukrs.
    when 'ord2'.
      sort tb ascending by kunnr bukrs.
  endcase.
```

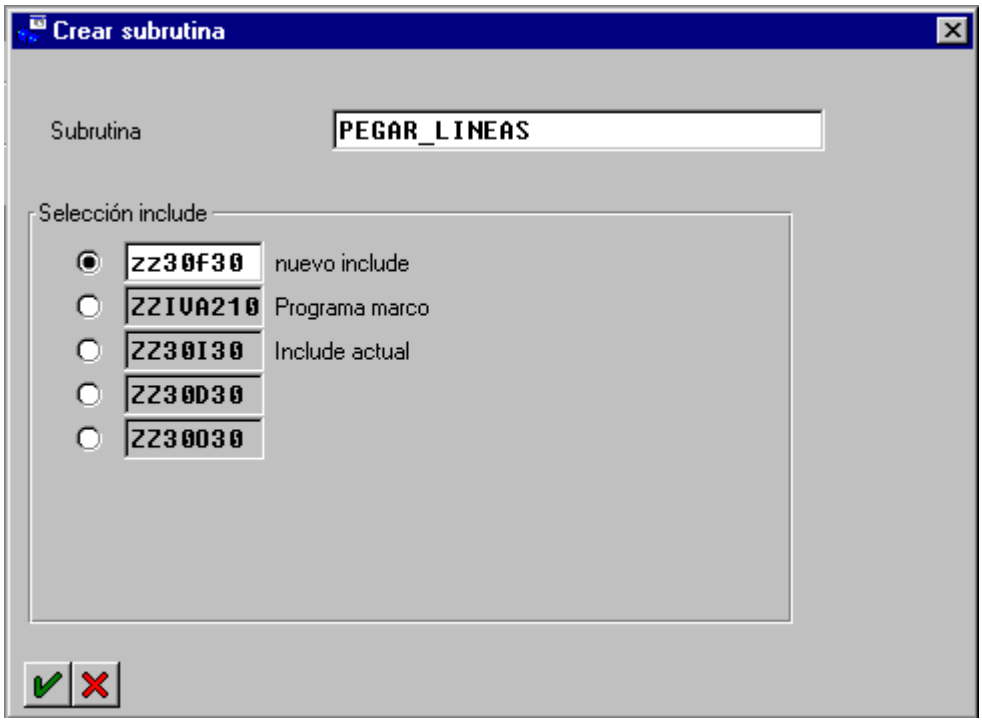
Recordar de poner el nombre del código de función de los botones que hemos creado en mayúsculas ya que SAP pone siempre los códigos de función en mayúsculas.

Los la explicación de los códigos de función es la siguiente:

- SAVE -> Pasa los registros de la tabla principal a la de diccionario.
- INS -> Inserta un registro abajo del registro seleccionado, cuando añado registros he decirle al “table control” el número de registros que tiene la tabla principal.
- NUEV -> Añade un registro al final de la tabla principal y en el “table control” la sensación es que se han añadido campos nuevos pero realmente solo se han añadido uno.
- MOV -> Muevo los registros seleccionados, estos registros estan almacenados en la tabla “tb_mov”, a la tabla del portapapeles, llamada “tb_por”.
- COPI -> Es lo mismo que el “MOV”, pero la diferencia estriba es que mientras al mover los registros borro los registros seleccionados del “table control” cuando copio esos registros no se borrán.
- PEG -> Pega los registros que estan en la tabla portapapeles, “tb_por” una posición más abajo partir del registro seleccionado.
- ORD1 -> Ordena el “table control” de forma descendente.
- ORD2 -> Ordena el “table control” de forma ascendente.

Habeis vista que utilizo procediemientos para realizar las operaciones de copiar, pegar, etc.. Con ello hago las cosas más claras. Los procedimientos estan metidos en la "include ZZ30F30", si ya tenéis esta include creada aumentar la númeracion pero no cambiar la "F" que esta medio, ya que indica que en esta include hay procedimientos.

Para crear la include seguiremos los mismos que hemos hecho para crear la include "ZZ30O30" o la que habéis creado para poner los módulos del PBO. Antes de todo grabaremos este incluide, después haremos doble clic en el procedimiento "pegar_lineas" (si no hemos grabado el include nos preguntara si lo queremos grabar, cosa que diremos que sí) nos saldrá la ventana que nos preguntará si queremeos crear el form le diremos que sí y nos saldrá la ventana que nos dice donde lo queremos crear, nosotros pondremos lo siguiente:



Una vez escrito el nuevo include haremos "enter" y nos saldrá esta otra ventana informandonos que se incluira la nueva include en el programa, si le damos al botón "Entrada nueva" volveremos a la la pantalla anterior y tendremos que introducir el nuevo include, nosotros pulsaremos "enter" y nos creara el módulo y el include. Una vez esto solo tenemos que escribir el código del procedimiento. Ahora pondré el código completo de la include "ZZ30F30" o la que tengas creada para este fin, vosotros solo tenis que crear cada form en la include "ZZ30F30" y escribir el código de ese form.

```
*&---------------------------------------------------------------------*
*&      form  insertar_linea
*&---------------------------------------------------------------------*
* inserto una linea una posición más, del registro seleccionado       *
*----------------------------------------------------------------------*
form insertar_linea.
* limpio el contenido de las tablas que utilizare para los movimientos
  clear tb1. refresh tb1. clear tb2. refresh tb2.
* como se puede seleccionar varios selecciones a la vez, miro todos los
```

```
* registros seleccionados
  loop at tb_mov.
* busco los registros en la tabla principal
    read table tb with key bukrs = tb_mov-bukrs
                           kunnr = tb_mov-kunnr
                           banks = tb_mov-banks
                           bankl = tb_mov-bankl
                           bankn = tb_mov-bankn
                           bkont = tb_mov-bkont
                           bvtyp = tb_mov-bvtyp
                           bkref = tb_mov-bkref
                           xezer = tb_mov-xezer.
    check sy-subrc = 0.
* guardo en que posicion de la tabla estan
    z = sy-tabix.
    x = 1.
    loop at tb.
* guardo los registros desde el inicio de la tabla principal hasta el
* registro encontrado en una tabla
      if x <= z.
        move-corresponding tb to tb1. append tb1.
      else.
* y del siguiente al encontrado hasta el final en otra
        move-corresponding tb to tb2. append tb2.
      endif.
      x = x + 1.
    endloop.
* limpio toda la tabla principal
    refresh tb. clear tb.
* inserto la primera parte en la tabla principal
    append lines of tb1 to tb.
* añado una linea en blanco en la tabla principal
    append tb.
* inserto la segunda parte en la tabla principal
    append lines of tb2 to tb.
  endloop.
* después limpio la tabla de movimientos
  refresh tb_mov. clear tb_mov.
* indico al table control cuantos registros tiene la tabla
  describe table tb lines tabla-lines.
* me pongo en la posición anterior al registro insertado
  tabla-top_line = z.
endform.                              " insertar_linea
*&---------------------------------------------------------------------*
*&      form  guardar_portapapeles
*&---------------------------------------------------------------------*
* guardo los registos seleccionados a la tabla del portapapeles       *
*---------------------------------------------------------------------*
form guardar_portapapeles.
* limpio la tabla donde guardare los registros movidos
```

```
  refresh tb_por. clear tb_por.
* como se pueden mover más de un registro, recorro la tabla de
* movimientos.
  loop at tb_mov.
* primero paso los registros de la tabla movimientos a la portapapeles
    move-corresponding tb_mov to tb_por.
* elimino de la tabla principal los registros seleccionados. si se ha
* seleccionado la opcion de move registros y si los elimino los guardo
* en la tabla de borrado.
    if okcode1 = 'mov'.
      delete tb where bukrs = tb_mov-bukrs and
                          kunnr = tb_mov-kunnr and
                          banks = tb_mov-banks and
                          bankl = tb_mov-bankl.
      move-corresponding tb_mov to tb_bor.
      append tb_bor.
    else.
* si se escoge la opción de copiar pongo en blanco el país y la cuenta
* con ello obligo a que el usuario introduzca esos dos campos a la
* fuerza evitando registros duplicados
      tb_por-banks = ''. tb_por-bankl = ''.
    endif.
* guardo los registros en la tabla de portapapeles
    append tb_por.
  endloop.
* limpio la tabla de movimientos
  refresh tb_mov. clear tb_mov.
* indico al table control el número de registros que tiene la tabla
* principal
  describe table tb lines tabla-lines.
endform.                              " cortar_lineas
*&---------------------------------------------------------------------*
*&      form  borrar_registros
*&---------------------------------------------------------------------*
* borro los registros seleccionados                                    *
*----------------------------------------------------------------------*
form borrar_registros.
* cuando pulso de botón de borrado, recorro la tabla de movimientos
  loop at tb_mov.
* busco en la tabla de movimientos por la sociedad y deudor
    read table tb with key bukrs = tb_mov-bukrs
                           kunnr = tb_mov-kunnr
                           banks = tb_mov-banks
                           bankl = tb_mov-bankl.
* si esta borra el registro.
    check sy-subrc = 0.
* me guardo el indice del registro encontrado
    z = sy-tabix.
* sy-tabix -> me devuelve el número de registro o linea en que se
* encuentra
```

```
    delete tb index sy-tabix.
* guardo el registro en la tabla de borrado, para después borrarla en
* la tabla de diccionario
    move-corresponding tb_mov to tb_bor. append tb_bor.
  endloop.
* limpio la tabla de moviemientos.
  refresh tb_mov. clear tb_mov.
* indico al table control cuantos registros hay en la tabla
  describe table tb lines tabla-lines.
* me posiciono el la posición del último registro borrado
  tabla-top_line = z.
endform.                              " borrar_registros
*&---------------------------------------------------------------------*
*&      form  pegar_lineas
*&---------------------------------------------------------------------*
* copio los registros del portapapeles, en una posición más al registro
* seleccionado
*----------------------------------------------------------------------*
form pegar_lineas.
* limpio el contenido de las tablas que utilizare para los movimientos
  clear tb1. refresh tb1. clear tb2. refresh tb2.
* como se puede se puede seleccionar varios registros a la vez miro
* toda la tabla. y en el primer registro seleccionado pego el contenido
* del portapapeles. el resto de registros seleccionados los ignoro
  loop at tb_mov.
* antes de la búsqueda limpio el header.
    clear tb.
* busco los registros en la tabla de movimientos
    read table tb with key bukrs = tb_mov-bukrs
                           kunnr = tb_mov-kunnr
                           banks = tb_mov-banks
                           bankl = tb_mov-bankl.
    if sy-subrc = 0.
* guardo en que posicion de la tabla esta el registro seleccionado
      z = sy-tabix.
      x = 1.
      loop at tb.
* guardo los registros desde el inicio de la tabla principal hasta el
* registro seleccionado en una tabla
        if x <= z.
          move-corresponding tb to tb1. append tb1.
        else.
* y del siguiente al seleccionado hasta el final en otra tabla
          move-corresponding tb to tb2. append tb2.
        endif.
        x = x + 1.                    " contador de registros añadidos
      endloop.
* limpio toda la tabla principal
      refresh tb. clear tb.
* inserto la primera parte en la tabla principal
```

```
      append lines of tb1 to tb.
* añado el contenido de la tabla del portapapeles a la tabla principal
      append lines of tb_por to tb.
* inserto la segunda parte en la tabla principal
      append lines of tb2 to tb.
    else.
* si da error quiere decir que estoy pegando fuera de los límites de la
* tabla pero no del table control. si esto ocurre insertaré los
* registros justamente después del último registro.
      append lines of tb_por to tb.
    endif.
  endloop.
* limpio la tabla de movimientos.
  clear tb_mov. refresh tb_mov.
* indico al table control cuantos registros tiene la tabla principal
  describe table tb lines tabla-lines.
* me posiciono el la posición del último registro borrado
  tabla-top_line = z.
endform.                              " pegar_lineas
*&---------------------------------------------------------------------*
*&      form  grabar_datos
*&---------------------------------------------------------------------*
* paso los registros de la tabla principal a la de diccionario         *
*----------------------------------------------------------------------*
form grabar_datos.
* si hay algún registro en blanco, cuando añadimos campos, los borro de
* la tabla interna.
  delete tb where kunnr = ''.
* borro los registros que estan en la tabla de borrado
  delete zztabpru10 from table tb_bor.
* modifico la tabla de diccionario.
  clear tb.
  loop at tb.
    clear zztabpru10.
    select single * from zztabpru10 where kunnr = tb-kunnr and
                                  bukrs = tb-bukrs and
                                  banks = tb-banks and
                                  bankl = tb-bankl.
    move-corresponding tb to zztabpru10.
    if sy-subrc ne 0.
      insert zztabpru10.
    else.
      modify zztabpru10.
    endif.
  endloop.
  set screen 1. leave screen.
endform.                              " grabar_datos
*&---------------------------------------------------------------------*
*&      form  buscar_soc_deudor_cta
*&---------------------------------------------------------------------*
```

```
* busco la sociedad, deudor y la cuenta en la tabla principal         *
*---------------------------------------------------------------------*
form buscar_soc_deudor_cta.
  read table tb with key kunnr = zztabpru10-kunnr
                         bukrs = zztabpru10-bukrs
                         bankl = zztabpru10-bankl
                         banks = zztabpru10-banks.
endform.                              " buscar_soc_deudor_cta
```

Una vez creado todos los forms lo grabaremos y generaremos, después volveremos atrás y generaremos el módulo. Después si no hemos generado la lógica de proceso de la segunda dynpro la generaremos.

PASO 6 (Solo falta ejecutarlo)

Ya hemos creado todo el programa y solo nos falta ejecutarlo y mirar como funciona.

El funcionamiento es simple cada vez que queremos hacer algo con un registro o varios primero hemos de seleccionar ese registro a través del botón que hay a la izquierda de cada registro en el “table control”. Y como ya he dicho al principio este ejemplo no cubre ciertas cosas, ya que solo es un ejemplo de cómo funciona este control.