/////////////////////POPUP FONKSİYONU ////////////////////////////

FUNCTION zfi\_f\_odev1.  
\*"----------------------------------------------------------------------  
\*"\*"Local Interface:  
\*"  EXPORTING  
\*"     REFERENCE(GT\_SCRFLDS) TYPE  ZCO\_TT\_ODEV1  
\*"----------------------------------------------------------------------  
  TABLES: zodev\_t\_0001.  
  
  DATA : gs\_data    TYPE zodev\_t\_0001,  
         gt\_data    TYPE TABLE OF zodev\_t\_0001,  
         gs\_scrflds TYPE  LINE OF  ZCO\_TT\_ODEV1.  
  DATA go\_alv TYPE REF TO cl\_salv\_table.  
  DATA: lo\_selections TYPE REF TO cl\_salv\_selections.  
  DATA lt\_rows TYPE salv\_t\_row.  
  DATA ls\_row TYPE i.  
  
  SELECT \* FROM zodev\_t\_0001 INTO TABLE gt\_data.  
  
  TRY.  
      cl\_salv\_table=>factory(  
        IMPORTING  
          r\_salv\_table   =  go\_alv                       " Basis Class Simple ALV Tables  
        CHANGING  
          t\_table        =   gt\_data[]  ).  
    CATCH cx\_salv\_msg. " ALV: General Error Class with Message  
  ENDTRY.  
  
  lo\_selections = go\_alv->get\_selections( ).  
  lo\_selections->set\_selection\_mode( if\_salv\_c\_selection\_mode=>row\_column ).  
  
  go\_alv->set\_screen\_popup(  
     start\_column = 25  
     end\_column  = 100  
     start\_line  = 6  
     end\_line    = 10 ).  
  
  go\_alv->display( ).  
  
  lt\_rows = lo\_selections->get\_selected\_rows( ).  
  
  LOOP AT lt\_rows INTO ls\_row.  
    READ  TABLE gt\_data INTO  gs\_scrflds INDEX ls\_row.  
    APPEND gs\_scrflds TO gt\_scrflds.  
\*\*action  
  ENDLOOP.  
ENDFUNCTION.

////////////////////////////DATA//////////////////////////////////

TABLES : marc , mard, mara, zodev\_t\_0001.  
TABLES sscrfields.  
DATA gt\_scrflds TYPE TABLE OF zodev\_t\_0001.  
DATA wa\_scrflds TYPE zodev\_t\_0001.  
DATA gt\_prog TYPE TABLE OF abaptext.  
DATA wa\_prog LIKE LINE OF gt\_prog.  
DATA: gs\_scrflds LIKE LINE OF gt\_scrflds.  
  
  
  
\*TYPES: BEGIN OF wa\_itab,  
\*         matnr TYPE marc-matnr,  
\*         werks TYPE mard-werks,  
\*         bwtty TYPE marc-bwtty,  
\*         ekgrp TYPE marc-ekgrp,  
\*         lgort TYPE mard-lgort,  
\*         lgpbe TYPE mard-lgpbe,  
\*       END OF wa\_itab.  
  
DATA: gt\_itab    LIKE TABLE OF zodev1\_s\_0001,  
\*      gt\_itab2 TYPE TABLE OF wa\_itab,  
      gt\_fcat1   TYPE lvc\_t\_fcat,  
      ob\_custom1 TYPE REF TO cl\_gui\_custom\_container,  
      ob\_grid1   TYPE REF TO cl\_gui\_alv\_grid.  
  
PARAMETERS: rb1 RADIOBUTTON GROUP  gr1 DEFAULT 'X' USER-COMMAND gr1.  
SELECTION-SCREEN BEGIN OF BLOCK b11 WITH FRAME TITLE TEXT-001.  
  
  SELECT-OPTIONS : s\_werks FOR marc-werks MODIF ID gr1,  
                 s\_matnr FOR marc-matnr MODIF ID gr1,  
                 s\_lgort FOR mard-lgort MODIF ID gr1.  
  
  
SELECTION-SCREEN END OF BLOCK b11.  
  
  
  
PARAMETERS: rb2 RADIOBUTTON GROUP  gr1.  
SELECTION-SCREEN BEGIN OF BLOCK b12 WITH FRAME TITLE TEXT-002.  
  
  SELECT-OPTIONS: s\_mtart FOR mara-mtart MODIF ID gr2,  
                  s\_matkl FOR mara-matkl MODIF ID gr2.  
  
  SELECTION-SCREEN PUSHBUTTON /1(20) TEXT-003 USER-COMMAND ekl.  
SELECTION-SCREEN END OF BLOCK b12.  
  
  
  
INITIALIZATION.  
  
AT SELECTION-SCREEN.  
  CLEAR gt\_scrflds.  
  
  IF sscrfields-ucomm EQ 'EKL'.  
  
    CALL FUNCTION 'ZFI\_F\_ODEV1'  
      IMPORTING  
        gt\_scrflds = gt\_scrflds.  
    clear gt\_prog.  
  
    if not gt\_scrflds is INITIAL.  
    PERFORM ekl\_button.  
        ENDIF.  
  
  ENDIF.

////////////////////////////FORM/////////////////////////////

DATA gt\_prog TYPE TABLE OF abaptext.  
  
\*&———————————————————————\*\* Start of Selection\*&———————————————————————\*  
START-OF-SELECTION.  
  DATA: go\_alv TYPE REF TO cl\_salv\_table.  
  DATA gt\_tmpflds TYPE TABLE OF zodev\_t\_0001.  
  DATA wa\_tmpflds TYPE zodev\_t\_0001.  
  DATA w\_type(60) TYPE c.  
  DATA w\_ctbix(10) TYPE c.  
  DATA w\_desc(10) TYPE c.  
  DATA columns TYPE REF TO cl\_salv\_columns\_table.  
  DATA  column  TYPE REF TO cl\_salv\_column.  
\*Retrive the required fields from the table  
\*  SELECT \* FROM zodev\_t\_0001    INTO TABLE gt\_scrflds.  
\*Build dynamic report code  
  
  SELECT marc~matnr  
   mard~werks  
      bwtty  
      ekgrp  
      lgort  
      lgpbe  
       INTO TABLE gt\_itab  
       FROM mara INNER JOIN marc  
       ON mara~matnr EQ marc~matnr  
   INNER JOIN mard ON  
    marc~matnr EQ mard~matnr  
  AND marc~werks EQ mard~werks  
  WHERE marc~werks IN s\_werks  
  AND marc~matnr IN s\_matnr  
  AND mard~lgort IN s\_lgort  
  AND mara~mtart IN s\_mtart  
  AND mara~matkl IN s\_matkl.  
  
cl\_salv\_table=>factory(  
         IMPORTING  
            r\_salv\_table   =  go\_alv  
         CHANGING  
           t\_table        =   gt\_itab[] ).  
columns = go\_alv->get\_columns( ).  
columns->set\_optimize( ).  
   go\_alv->display( ).  
  
FORM ekl\_button.  
  
  CLEAR gt\_prog.  
  wa\_prog = 'REPORT zdynselscr.'.  
  APPEND wa\_prog TO gt\_prog.  
  wa\_prog = 'TABLES : MARC , MARD, MARA, ZODEV\_T\_0001.                            '.  
  APPEND wa\_prog TO gt\_prog.  
  
  wa\_prog = ' DATA: go\_alv TYPE REF TO cl\_salv\_table.'.  
  APPEND wa\_prog TO gt\_prog.  
  wa\_prog = ' DATA: lo\_selections TYPE REF TO cl\_salv\_selections.'.  
  APPEND wa\_prog TO gt\_prog.  
  wa\_prog = ' DATA: lt\_rows TYPE salv\_t\_row.'.  
  APPEND wa\_prog TO gt\_prog.  
  wa\_prog = ' DATA: ls\_row TYPE i.'.  
  APPEND wa\_prog TO gt\_prog.  
  wa\_prog = ' DATA: gt\_itab  LIKE TABLE OF ZODEV1\_S\_0001. '.  
  APPEND wa\_prog TO gt\_prog.  
  wa\_prog = 'DATA : columns TYPE REF TO cl\_salv\_columns\_table.'.  
  APPEND wa\_prog TO gt\_prog.  
  wa\_prog = 'DATA : column  TYPE REF TO cl\_salv\_column.'.  
  APPEND wa\_prog TO gt\_prog.  
  wa\_prog = 'PARAMETERS: rb1 RADIOBUTTON GROUP  gr1 DEFAULT ''X'' USER-COMMAND gr1.'.  
  APPEND wa\_prog TO gt\_prog.  
  wa\_prog = 'SELECTION-SCREEN BEGIN OF block b11 with FRAME TITLE text-001.       '.  
  APPEND wa\_prog TO gt\_prog.  
  wa\_prog = 'SELECT-OPTIONS : werks for marc-werks MODIF ID gr1,                '.  
  APPEND wa\_prog TO gt\_prog.  
  wa\_prog = '                 matnr for marc-matnr MODIF ID gr1,                '.  
  APPEND wa\_prog TO gt\_prog.  
  wa\_prog = '                 lgort for mard-lgort MODIF ID gr1.                '.  
  APPEND wa\_prog TO gt\_prog.  
  wa\_prog = 'SELECTION-SCREEN END OF BLOCK B11.                                   '.  
  APPEND wa\_prog TO gt\_prog.  
  wa\_prog = 'PARAMETERS: rb2 RADIOBUTTON GROUP  gr1.                              '.  
  APPEND wa\_prog TO gt\_prog.  
  wa\_prog = 'SELECTION-SCREEN BEGIN OF block b12 with FRAME TITLE text-002.       '.  
  APPEND wa\_prog TO gt\_prog.  
  wa\_prog = 'SELECT-OPTIONS: mtart FOR mara-mtart MODIF ID gr2,                '.  
  APPEND wa\_prog TO gt\_prog.  
  wa\_prog = '                matkl FOR mara-matkl MODIF ID gr2.              '.  
  APPEND wa\_prog TO gt\_prog.  
  wa\_prog = 'SELECTION-SCREEN END OF BLOCK B12.                                   '.  
  APPEND wa\_prog TO gt\_prog.  
\*\*\*\*fonksiyon  
  
  
\*\*Code for building TABLES section\* Copy to a temporary internal table  
  gt\_tmpflds[] = gt\_scrflds[].  
\*Delete screen fields for parameters in the temp table  
  DELETE gt\_tmpflds WHERE parasel NE 'S'.  
  SORT gt\_tmpflds BY t\_ad.  
\*Remove duplicates  
  DELETE ADJACENT DUPLICATES FROM gt\_tmpflds COMPARING t\_ad.  
\*Build TABLES… code for table workarea  
\*  LOOP AT gt\_tmpflds INTO wa\_tmpflds.  
\*    CONCATENATE 'tables' wa\_tmpflds-t\_ad '.'      INTO wa\_prog      SEPARATED BY space.  
\*    APPEND wa\_prog TO gt\_prog.  
\*  ENDLOOP.  
\*Build Selection screen field definition  
  LOOP AT gt\_scrflds INTO wa\_scrflds.  
    w\_ctbix = sy-tabix.  
    CONDENSE w\_ctbix.  
    wa\_prog = 'selection-screen BEGIN OF LINE.'.  
    APPEND wa\_prog TO gt\_prog.  
\*Comments field in the selection field  
    CONCATENATE      'selection-screen comment 1(20) cmt' w\_ctbix '.'  
       INTO wa\_prog.  
    APPEND wa\_prog TO gt\_prog.  
\*Concatenate ref. tablename and A\_EX\*  
\*    Eg. : mara / matnr into mara-matnr  
    CONCATENATE wa\_scrflds-t\_ad wa\_scrflds-a\_ad      INTO w\_type      SEPARATED BY '-'.  
\*Code generation for PARAMETERS type  
    IF wa\_scrflds-parasel EQ 'P'.  
      CONCATENATE 'parameters' wa\_scrflds-a\_ad 'like' w\_type '.'        INTO wa\_prog        SEPARATED BY space.  
    ENDIF.  
\*Code generation for SELECT-OPTIONS type  
    IF wa\_scrflds-parasel EQ 'S'.  
      CONCATENATE     'select-options' wa\_scrflds-a\_ad 'for' w\_type '.'   INTO wa\_prog    SEPARATED BY space.  
    ENDIF.  
    APPEND wa\_prog TO gt\_prog.  
    wa\_prog = 'selection-screen END OF LINE.'.  
    APPEND wa\_prog TO gt\_prog.  
  ENDLOOP.  
\*End of code generation for selection screen\* Generate Initialization event code\* Initialize the comments fields defined in the selection screen  
  wa\_prog = 'initialization.'.  
  APPEND wa\_prog TO gt\_prog.  
  LOOP AT gt\_scrflds INTO wa\_scrflds.  
    w\_ctbix = sy-tabix.  
    CONDENSE w\_ctbix.  
    CONCATENATE 'cmt' w\_ctbix      INTO w\_ctbix.  
\*Get the description from the table DDFTX  
    SELECT SINGLE scrtext\_s  
         FROM ddftx  
        INTO w\_desc  WHERE tabname EQ wa\_scrflds-t\_ad  
            AND fieldname EQ wa\_scrflds-a\_ad  
            AND ddlanguage EQ sy-langu.  
    CONCATENATE w\_ctbix ' = '''  
                w\_desc  
                ' ''.'  
      INTO wa\_prog.  
    APPEND wa\_prog TO gt\_prog.  
  ENDLOOP.  
\*Generate Start of selection event code  
  
  wa\_prog = 'start-of-selection.'.  
  APPEND wa\_prog TO gt\_prog.  
\*Data types for shared objects  
  wa\_prog = 'data t\_ranges TYPE ace\_field\_ranges\_t.'.  
  APPEND wa\_prog TO gt\_prog.  
  wa\_prog = 'data wa\_ranges LIKE LINE OF t\_ranges.'.  
  APPEND wa\_prog TO gt\_prog.  
  wa\_prog = 'data t\_fldrange TYPE ace\_generic\_range\_t.'.  
  APPEND wa\_prog TO gt\_prog.  
  wa\_prog = 'data wa\_fldrange TYPE ace\_generic\_range.'.  
  APPEND wa\_prog TO gt\_prog.  
\*Read the input from the selection screen and\*  store it in Shared Objects  
  LOOP AT gt\_scrflds INTO wa\_scrflds.  
\*Assign the A\_EX in the variable  
    CONCATENATE 'wa\_ranges-fieldname = '''   wa\_scrflds-a\_ad  
    ' ''.'  
    INTO wa\_prog SEPARATED BY space.  
    APPEND wa\_prog TO gt\_prog.  
    wa\_prog = 'refresh t\_fldrange.'.  
    APPEND wa\_prog TO gt\_prog.  
\*Processing for SELECT-OPTIONS\*   Loop and move data to the shared memory  
\*    IF wa\_scrflds-PARASEL EQ 'S'.  
\*\*\*\*\*\*\*\*\*\*\*6666666  
\*      CONCATENATE 'loop at ' wa\_scrflds-A\_AD '.' into wa\_prog separated by space.  
\*      APPEND wa\_prog TO gt\_prog.  
\*      wa\_prog = 'clear wa\_fldrange.'.  
\*      APPEND wa\_prog TO gt\_prog.  
\*      CONCATENATE 'move-corresponding ' wa\_scrflds-A\_AD 'to' 'wa\_fldrange.'  
\*        into wa\_prog separated by space.  
\*      APPEND wa\_prog TO gt\_prog.  
\*      wa\_prog = 'append wa\_fldrange TO t\_fldrange.'.  
\*      APPEND wa\_prog TO gt\_prog.  
\*      wa\_prog = 'endloop.'.  
\*      APPEND wa\_prog TO gt\_prog.  
\*      wa\_prog = 'wa\_ranges-fieldrange[] = t\_fldrange[].'.  
\*      APPEND wa\_prog TO gt\_prog.  
\*      wa\_prog = 'append wa\_ranges TO t\_ranges.'.  
\*      APPEND wa\_prog TO gt\_prog.  
\*    ELSEIF wa\_scrflds-PARASEL EQ 'p'.  
\*      wa\_prog = 'clear wa\_fldrange.'.  
\*      APPEND wa\_prog TO gt\_prog.  
\*\*Process if the parameter is not initial  
\*      CONCATENATE 'if ' wa\_scrflds-A\_AD 'is not initial.'  
\*  into wa\_prog separated by space.  
\*      APPEND wa\_prog TO gt\_prog.  
\*      wa\_prog = 'wa\_fldrange-sign = ''i''.'.  
\*      APPEND wa\_prog TO gt\_prog.  
\*      wa\_prog = 'wa\_fldrange-option = ''eq''.'.  
\*      APPEND wa\_prog TO gt\_prog.  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*666666  
\*      CONCATENATE 'wa\_fldrange-low = ' wa\_scrflds-A\_EX '.'  
\*        into wa\_prog separated by space.  
\*      APPEND wa\_prog TO gt\_prog.  
\*      wa\_prog = 'append wa\_fldrange TO t\_fldrange.'.  
\*      APPEND wa\_prog TO gt\_prog.  
\*      wa\_prog = 'endif.'.  
\*      APPEND wa\_prog TO gt\_prog.  
\*      wa\_prog = 'wa\_ranges-fieldrange[] = t\_fldrange[].'.  
\*      APPEND wa\_prog TO gt\_prog.  
\*      wa\_prog = 'append wa\_ranges TO t\_ranges.'.  
\*      APPEND wa\_prog TO gt\_prog.  
\*    ENDIF.  
  ENDLOOP.  
  
  DATA: tabix    TYPE i,  
        l\_where2 TYPE string,  
        l\_where  TYPE string.  
  
  
  wa\_prog = ' SELECT marc~matnr '.  
  APPEND wa\_prog TO gt\_prog.  
  wa\_prog = ' mard~werks '.  
  APPEND wa\_prog TO gt\_prog.  
  wa\_prog = '    bwtty '.  
  APPEND wa\_prog TO gt\_prog.  
  wa\_prog = '    ekgrp '.  
  APPEND wa\_prog TO gt\_prog.  
  wa\_prog = '    lgort '.  
  APPEND wa\_prog TO gt\_prog.  
  wa\_prog = '    lgpbe '.  
  APPEND wa\_prog TO gt\_prog.  
  wa\_prog = '     INTO TABLE gt\_itab '.  
  APPEND wa\_prog TO gt\_prog.  
  wa\_prog = '     FROM mara INNER JOIN marc '.  
  APPEND wa\_prog TO gt\_prog.  
  wa\_prog = '     ON mara~matnr EQ marc~matnr '.  
  APPEND wa\_prog TO gt\_prog.  
  wa\_prog =   ' INNER JOIN mard on'.  
  APPEND wa\_prog TO gt\_prog.  
  wa\_prog = '  marc~matnr EQ mard~matnr '.  
  APPEND wa\_prog TO gt\_prog.  
  wa\_prog =  'AND marc~werks EQ mard~werks '.  
  APPEND wa\_prog TO gt\_prog.  
  CONCATENATE 'where marc~werks IN werks ' l\_where INTO wa\_prog SEPARATED BY space.  
  APPEND wa\_prog TO gt\_prog.  
  wa\_prog = 'and marc~matnr IN matnr'.  
  APPEND wa\_prog TO gt\_prog.  
  wa\_prog = 'and mard~lgort IN lgort'.  
  APPEND wa\_prog TO gt\_prog.  
  wa\_prog = 'and mara~mtart IN mtart'.  
  APPEND wa\_prog TO gt\_prog.  
  wa\_prog = 'and mara~matkl IN matkl'.  
  APPEND wa\_prog TO gt\_prog.  
  
  LOOP AT gt\_scrflds INTO gs\_scrflds.  
    tabix = sy-tabix.  
  
    CONCATENATE gs\_scrflds-t\_ad '~' gs\_scrflds-a\_ad INTO l\_where2.  
  
\*    IF tabix EQ 1.  
\*  
\*      IF gs\_SCRFLDS-parasel = 'P'.  
\*        CONCATENATE l\_where2  'EQ' gs\_SCRFLDS-a\_ad INTO l\_where SEPARATED BY space.  
\*        wa\_prog = l\_where.  
\*        APPEND wa\_prog TO gt\_prog.  
\*      ELSE.  
\*        CONCATENATE l\_where2  'IN' gs\_SCRFLDS-a\_ad INTO l\_where SEPARATED BY space.  
\*        wa\_prog = l\_where.  
\*        APPEND wa\_prog TO gt\_prog.  
\*      ENDIF.  
\*  
\*    ELSE.  
  
    IF gs\_scrflds-parasel = 'P'.  
      CONCATENATE  'and' l\_where2 'EQ' gs\_scrflds-a\_ad INTO l\_where SEPARATED BY space.  
      wa\_prog = l\_where.  
      APPEND wa\_prog TO gt\_prog.  
    ELSE.  
      CONCATENATE  'and' l\_where2 'IN' gs\_scrflds-a\_ad INTO l\_where SEPARATED BY space.  
      wa\_prog = l\_where.  
      APPEND wa\_prog TO gt\_prog.  
    ENDIF.  
  
\*ENDIF.  
  ENDLOOP.  
\*  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*ALV  
  
  wa\_prog =   '  .      cl\_salv\_table=>factory( '.  
  APPEND wa\_prog TO gt\_prog.  
  wa\_prog =   '         IMPORTING '.  
  APPEND wa\_prog TO gt\_prog.  
  wa\_prog =   '            r\_salv\_table   =  go\_alv '.  
  APPEND wa\_prog TO gt\_prog.  
  wa\_prog =   '         CHANGING '.  
  APPEND wa\_prog TO gt\_prog.  
  wa\_prog =   '           t\_table        =   gt\_itab[] ). '.  
  APPEND wa\_prog TO gt\_prog.  
  wa\_prog = 'columns = go\_alv->get\_columns( ).'.  
  APPEND wa\_prog TO gt\_prog.  
  wa\_prog = 'columns->set\_optimize( ).'.  
  APPEND wa\_prog TO gt\_prog.  
  wa\_prog =   '   go\_alv->display( ). '.  
  APPEND wa\_prog TO gt\_prog.  
  
  
  
  
\*  wa\_prog = 'data: area TYPE ref to ZSELAREA2.'.  
\*  APPEND wa\_prog TO gt\_prog.  wa\_prog = 'data root TYPE REF to zcl\_selscr.'.  
\*  APPEND wa\_prog TO gt\_prog.  
\*\*Get ref to the shared memory  
\*  wa\_prog = 'area = ZSELAREA2=>attach\_for\_write( ).'.  
\*  APPEND wa\_prog TO gt\_prog.  
\*  wa\_prog = 'create object root area handle area.'.  
\*  APPEND wa\_prog TO gt\_prog.  
\*\*Store the value in the memory  
\*  wa\_prog = 'root->set\_fields( t\_ranges ).'.  
\*  APPEND wa\_prog TO gt\_prog.  
\*  wa\_prog = 'area->set\_root(  root ).'.  
\*  APPEND wa\_prog TO gt\_prog.  
\*\*Commit and detatch  
\*  wa\_prog = 'area->detach\_commit( ).'.  
\*  APPEND wa\_prog TO gt\_prog.  
\*\*Quit the program: Return to the main code  
\*  wa\_prog = 'leave PROGRAM.'.  
\*  APPEND wa\_prog TO gt\_prog.  
\*Generate the report program  
  INSERT REPORT 'zdynselscr' FROM gt\_prog.  
  COMMIT WORK.  
\*Execute the report  
  SUBMIT zdynselscr VIA SELECTION-SCREEN AND RETURN.  
\*Read the data from the shared memory  
\*  DATA area TYPE REF TO zselarea2.  
\*  area = zselarea2=>attach\_for\_read( ).  
\*  DATA t\_ranges TYPE ace\_field\_ranges\_t.  
\*  t\_ranges = area->root->get\_fields( ).  
\*  area->detach( ).  
\*  TYPES cond(72) TYPE c.  
\*  DATA it\_vbak TYPE TABLE OF vbak.  
\*  DATA wa\_vbak LIKE LINE OF it\_vbak.  
\*  DATA t\_cond TYPE TABLE OF cond WITH HEADER LINE.  
\*  DATA wa\_ranges LIKE LINE OF t\_ranges.  
\*  DATA wa\_fieldrange LIKE LINE OF wa\_ranges-fieldrange.  
\*  DATA dref TYPE REF TO data.  
\*  LOOP AT t\_ranges INTO wa\_ranges.  
\*    WRITE / wa\_ranges-fieldname COLOR COL\_HEADING.  
\*    ULINE AT /8(68).  
\*    LOOP AT wa\_ranges-fieldrange INTO wa\_fieldrange.  
\*      WRITE :/8 sy-vline, 10 wa\_fieldrange-sign, 14 sy-vline, 15 wa\_fieldrange-option,19 sy-vline, 20 wa\_fieldrange-low, 42 sy-vline, 75 sy-vline.  
\*      IF wa\_fieldrange-high IS NOT INITIAL.  
\*        WRITE AT 43 wa\_fieldrange-high.  
\*        WRITE AT 75 sy-vline.  
\*      ENDIF.  
\*    ENDLOOP.  
\*    ULINE AT /8(68).  
\*  ENDLOOP.  
  
ENDFORM.  
\*  ZDYNSELSCR  
  
\*CALL FUNCTION 'RH\_DYNAMIC\_WHERE\_BUILD'  
\*  EXPORTING  
\*    dbtable               = zodev\_t\_0001  
\*  tables  
\*\*Internal table contains the fieldnames and their values used in where clause.  
\*    condtab               =  
\*    where\_clause          =  
\*\* EXCEPTIONS  
\*\*   EMPTY\_CONDTAB         = 1  
\*\*   NO\_DB\_FIELD           = 2  
\*\*   UNKNOWN\_DB            = 3  
\*\*   WRONG\_CONDITION       = 4  
\*\*   OTHERS                = 5  
\*          .  
\*IF sy-subrc <> 0.  
\*\* Implement suitable error handling here  
\*ENDIF.