TOP:

```
*&---------------------------------------------------------------------*
-

DATA: ok_code TYPE sy-ucomm.      "variable to capture function code on user
interaction

"variables for login screen (0100)
DATA: io_user     TYPE string,
      io_password TYPE string.

DATA: lv_empname  TYPE z021employee-emp_name.

"structure and work area for register screen (0200)
TYPES: BEGIN OF ty_empreg,
         emp_name TYPE z021employee-emp_name,
         gender   TYPE z021employee-gender,
         email_id TYPE z021employee-email_id,
         set_pass TYPE c LENGTH 30,
         re_pass  TYPE c LENGTH 30,
       END OF ty_empreg.

DATA: ls_empreg TYPE ty_empreg.

"variable for department field (0300)
DATA: io_dept TYPE c LENGTH 30.

"internal table and work area for details table control (0300)
DATA: lt_empdet TYPE STANDARD TABLE OF z021employee,
      ls_empdet TYPE z021employee.

"variables as flags for controlling flow logic
DATA: lv_validemail   TYPE i,
      lv_validpass    TYPE i,
      lv_regsuccess   TYPE i,
      lv_tcfetch      TYPE i,
      lv_emaildisable TYPE i,
      lv_deltc_fetch  TYPE i.

*&SPWIZARD: DECLARATION OF TABLECONTROL 'TC_EMPDET' ITSELF
CONTROLS: tc_empdet TYPE TABLEVIEW USING SCREEN 0300.

*&SPWIZARD: FUNCTION CODES FOR TABSTRIP 'TS_ADMIN'
CONSTANTS: BEGIN OF c_ts_admin,
             tab1 LIKE sy-ucomm VALUE 'TS_ADMIN_FC1',
             tab2 LIKE sy-ucomm VALUE 'TS_ADMIN_FC2',
           END OF c_ts_admin.
*&SPWIZARD: DATA FOR TABSTRIP 'TS_ADMIN'
```

```
CONTROLS:  ts_admin TYPE TABSTRIP.
DATA: BEGIN OF g_ts_admin,
        subscreen   LIKE sy-dynnr,
        prog        LIKE sy-repid VALUE 'Z544_EMP_MANAGE',
        pressed_tab LIKE sy-ucomm,
      END OF g_ts_admin.

DATA: ls_ins_emp TYPE z021employee.

TYPES: BEGIN OF ty_del_emp,
         mark     TYPE c LENGTH 1,
         emp_no   TYPE z021employee-emp_no,
         emp_name TYPE z021employee-emp_name,
         gender   TYPE z021employee-gender,
         email_id TYPE z021employee-email_id,
         dept     TYPE z021employee-dept,
       END OF ty_del_emp.

DATA: lt_del_emp TYPE TABLE OF ty_del_emp,
      ls_del_emp TYPE ty_del_emp.

*&SPWIZARD: DECLARATION OF TABLECONTROL 'TC_DEL_EMP' ITSELF
CONTROLS: tc_del_emp TYPE TABLEVIEW USING SCREEN 0402.

*&SPWIZARD: LINES OF TABLECONTROL 'TC_DEL_EMP'
DATA:     g_tc_del_emp_lines  LIKE sy-loopc.



DATA: gt_data     TYPE TABLE OF zdt_g3order_st,
      gr_alv_grid TYPE REF TO cl_gui_alv_grid,
      gr_container TYPE REF TO cl_gui_custom_container.
```

O01:

```
*----------------------------------------------------------------------*
```

```
MODULE status_0100 OUTPUT.
  SET PF-STATUS 'GUI_0100'.
  SET TITLEBAR 'TITLE_0100'.
ENDMODULE.
*&---------------------------------------------------------------------*
*&      Module  STATUS_0200  OUTPUT
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
MODULE status_0200 OUTPUT.
  SET PF-STATUS 'GUI_0200'.
  SET TITLEBAR 'TITLE_0200'.
ENDMODULE.
*&---------------------------------------------------------------------*
*&      Module  VALIDATE_LOGIN  INPUT
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
MODULE validate_login INPUT.

  IF ok_code = 'FC_LOGIN'.
    PERFORM login_validate.
  ENDIF.

ENDMODULE.
*&---------------------------------------------------------------------*
*&      Module  SCR_MODIFY  OUTPUT
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
MODULE scr_modify OUTPUT.

  LOOP AT SCREEN.

    IF lv_regsuccess IS INITIAL.

      IF screen-name = 'PB_GOLOG'.
        screen-active = 0.

      ENDIF.

    ELSE.
      IF screen-name = 'PB_GOLOG'.
        screen-active = 1.
      ELSE.
        IF screen-name = 'LS_EMPREG-SET_PASS' OR screen-name = 'LS_EMPREG-
RE_PASS'.
          screen-invisible = 0.
        ENDIF.
        screen-input = 0.
      ENDIF.

    ENDIF.
    MODIFY SCREEN.

  ENDLOOP.


ENDMODULE.
*&---------------------------------------------------------------------*
*&      Module  STATUS_0300  OUTPUT
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
MODULE status_0300 OUTPUT.

  "enable/diable Save button only if valid email ID
  IF lv_emaildisable IS INITIAL.
    SET PF-STATUS 'GUI_0300' EXCLUDING 'FC_SAVE'.
  ELSE.
    SET PF-STATUS 'GUI_0300'.
  ENDIF.

  SET TITLEBAR 'TITLE_0300'.

ENDMODULE.

*&SPWIZARD: OUTPUT MODULE FOR TC 'TC_EMPDET'. DO NOT CHANGE THIS LINE!
*&SPWIZARD: UPDATE LINES FOR EQUIVALENT SCROLLBAR
MODULE tc_empdet_change_tc_attr OUTPUT.
  DESCRIBE TABLE lt_empdet LINES tc_empdet-lines.
ENDMODULE.
*&---------------------------------------------------------------------*
*&      Module  FETCH_DATA  OUTPUT
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
MODULE fetch_data OUTPUT.

  IF lv_tcfetch IS INITIAL.
    SELECT * FROM z021employee INTO TABLE lt_empdet
      WHERE dept = io_dept.
    IF sy-subrc IS INITIAL.
      lv_tcfetch = 1.
    ENDIF.
  ENDIF.

ENDMODULE.
```

```
*&---------------------------------------------------------------------*
*&      Module  TC_MODIFY  OUTPUT
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
MODULE tc_modify OUTPUT.

  DATA: ls_col TYPE cxtab_column.

  IF lv_tcfetch IS INITIAL.         "hide tc at start - no records in table
control
    tc_empdet-invisible = 'X'.

  ELSE.
    tc_empdet-invisible = ' '.
  ENDIF.

  "logic to enable disable email field: flag set at FC_CHECK in case of valid
email
  IF lv_emaildisable IS NOT INITIAL.
    LOOP AT tc_empdet-cols INTO ls_col.
      IF ls_col-screen-name = 'LS_EMPDET-EMAIL_ID'.
        ls_col-screen-input = 0.
        MODIFY tc_empdet-cols FROM ls_col INDEX sy-tabix.
      ENDIF.
    ENDLOOP.

  ELSE.
    LOOP AT tc_empdet-cols INTO ls_col.
      IF ls_col-screen-name = 'LS_EMPDET-EMAIL_ID'.
        ls_col-screen-input = 1.
        MODIFY tc_empdet-cols FROM ls_col INDEX sy-tabix.
      ENDIF.
    ENDLOOP.

  ENDIF.

ENDMODULE.
*&---------------------------------------------------------------------*
*&      Module  STATUS_0400  OUTPUT
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
MODULE status_0400 OUTPUT.
  SET PF-STATUS 'GUI_0400'.
  SET TITLEBAR 'TITLE_0400' WITH lv_empname.
ENDMODULE.

*&SPWIZARD: OUTPUT MODULE FOR TS 'TS_ADMIN'. DO NOT CHANGE THIS LINE!
```

```
*&SPWIZARD: SETS ACTIVE TAB
MODULE ts_admin_active_tab_set OUTPUT.

  ok_code = sy-ucomm.

  "To restrict refresh
  IF g_ts_admin-pressed_tab IS INITIAL.
    g_ts_admin-pressed_tab = c_ts_admin-tab1.
  ELSE.
    g_ts_admin-pressed_tab = ts_admin-activetab.
  ENDIF.

*  ts_admin-activetab = g_ts_admin-pressed_tab.
  CASE g_ts_admin-pressed_tab.
    WHEN c_ts_admin-tab1.
      g_ts_admin-subscreen = '0401'.
    WHEN c_ts_admin-tab2.
      g_ts_admin-subscreen = '0402'.
    WHEN OTHERS.
*&SPWIZARD:      DO NOTHING
  ENDCASE.
ENDMODULE.

*&SPWIZARD: OUTPUT MODULE FOR TC 'TC_DEL_EMP'. DO NOT CHANGE THIS LINE!
*&SPWIZARD: UPDATE LINES FOR EQUIVALENT SCROLLBAR
MODULE tc_del_emp_change_tc_attr OUTPUT.
  DESCRIBE TABLE lt_del_emp LINES tc_del_emp-lines.
ENDMODULE.
*&---------------------------------------------------------------------*
*&      Module  TC_DEL_FETCH  OUTPUT
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
MODULE tc_del_fetch OUTPUT.

  IF lv_deltc_fetch IS INITIAL.
    SELECT * FROM z021employee INTO CORRESPONDING FIELDS OF TABLE lt_del_emp.
    IF sy-subrc IS INITIAL.
      lv_deltc_fetch = 1.
    ENDIF.
  ENDIF.

ENDMODULE.
```

I01:

```
*----------------------------------------------------------------------*
MODULE user_command_0100 INPUT.

  IF ok_code = 'FC_GOREG'.
    CALL SCREEN 0200.              "Go to Register screen

  ENDIF.

ENDMODULE.
*&---------------------------------------------------------------------*
*&      Module  EXIT  INPUT
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
MODULE exit INPUT.

  IF ok_code = 'FC_BACK'.
    LEAVE TO SCREEN 0.

  ELSEIF ok_code = 'FC_EXIT'.
    LEAVE PROGRAM.

  ELSEIF ok_code = 'FC_CANCEL'.
    IF sy-dynnr = '0100'.
      CLEAR: io_user, io_password.

    ELSEIF sy-dynnr = '0200'.
      CLEAR ls_empreg.

    ELSEIF sy-dynnr = '0300'.
      CLEAR lv_tcfetch.

    ELSEIF sy-dynnr = '0400'.
      CLEAR: lv_deltc_fetch, g_ts_admin-pressed_tab.

    ENDIF.
    LEAVE SCREEN.

  ENDIF.

ENDMODULE.
*&---------------------------------------------------------------------*
*&      Module  F4HELP_DEPT  INPUT
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
MODULE f4help_dept INPUT.
```

```
  PERFORM f4help_dept.

ENDMODULE.


*----------------------------------------------------------------------*
MODULE user_command_0200 INPUT.

  IF ok_code = 'FC_REG'.
    PERFORM register_user.
  ENDIF.

ENDMODULE.
*&---------------------------------------------------------------------*
*&      Module  VALIDATE_EMAIL  INPUT
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
MODULE validate_email INPUT.

  "validate logic on click of Register button (0200)
  IF ok_code = 'FC_REG'.
    IF ( strlen( ls_empreg-email_id ) < 8 ).
      MESSAGE 'Email should be at least 8 characters' TYPE 'E'.

    ELSEIF ls_empreg-email_id NA '0123456789'.
      MESSAGE 'Email should at least contain a number' TYPE 'E'.

    ELSE.
      lv_validemail = 1.
    ENDIF.

    "validate logic on click of Check button (0300)
  ELSEIF ok_code = 'FC_CHECK'.

    IF ( strlen( ls_empdet-email_id ) < 8 ).
      CLEAR lv_emaildisable.
      MESSAGE 'Email should be at least 8 characters' TYPE 'E'.

    ELSEIF ls_empdet-email_id NA '0123456789'.
      CLEAR lv_emaildisable.
      MESSAGE 'Email should at least contain a number' TYPE 'E'.

    ELSE.
      MESSAGE 'Validated' TYPE 'S'.
      lv_emaildisable = 1.            "set flag to disable email ID column

    ENDIF.
  ENDIF.
```

```
ENDMODULE.
*&---------------------------------------------------------------------*
*&      Module  VALIDATE_PASS  INPUT
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
MODULE validate_pass INPUT.

  IF ok_code = 'FC_REG'.
    IF ls_empreg-set_pass <> ls_empreg-re_pass.
      MESSAGE 'Password and Re-enter password should be same' TYPE 'E'.

    ELSE.
      lv_validpass = 1.
    ENDIF.
  ENDIF.

ENDMODULE.
*&---------------------------------------------------------------------*
*&      Form  REGISTER_USER
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
*  -->  p1        text
*  <--  p2        text
*----------------------------------------------------------------------*
FORM register_user .

  DATA: ls_emp    TYPE z021employee,
        lv_maxemp TYPE z021employee-emp_no,
        lv_pre    TYPE c LENGTH 2,
        lv_post   TYPE string,
        lv_no     TYPE i.

  IF lv_validemail IS NOT INITIAL AND lv_validpass IS NOT INITIAL.

    MOVE-CORRESPONDING ls_empreg TO ls_emp.

    SELECT MAX( emp_no ) FROM z021employee INTO lv_maxemp.
    IF sy-subrc IS NOT INITIAL.
      CLEAR lv_no.
    ELSE.
      SPLIT lv_maxemp AT 'L' INTO lv_pre lv_post.
      lv_no = lv_post.
      lv_no = lv_no + 1.
      lv_post = lv_no.
    ENDIF.
    CONCATENATE 'SL' lv_post INTO ls_emp-emp_no.
    ls_emp-dept = 'OTHERS'.

    INSERT z021employee FROM ls_emp.
    IF sy-subrc IS INITIAL.
      MESSAGE 'Registration Successful' TYPE 'S'.
      lv_regsuccess = 1.
      CLEAR ls_empreg.

    ELSE.
      MESSAGE 'Some error occurred, please try again' TYPE 'E'.
    ENDIF.
  ENDIF.


ENDFORM.
*&---------------------------------------------------------------------*
*&      Module  USER_COMMAND_0300  INPUT
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
MODULE user_command_0300 INPUT.

  "call popup screen
  IF ok_code = 'FC_SAVE'.
    CALL SCREEN 0301 STARTING AT 10 5 ENDING AT 70 15.

    "call Admin screen
  ELSEIF ok_code = 'FC_ADMIN'.
    CALL SCREEN 0400.
  ENDIF.

ENDMODULE.
*&---------------------------------------------------------------------*
*&      Module  IT_MODIFY  INPUT
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
MODULE it_modify INPUT.

  "modify internal table to get value entered by user on screen
  MODIFY lt_empdet FROM ls_empdet INDEX tc_empdet-current_line.
  IF sy-subrc IS NOT INITIAL.
    MESSAGE 'Some error occurred' TYPE 'E'.
  ENDIF.

ENDMODULE.
*&---------------------------------------------------------------------*
*&      Module  USER_COMMAND_0301  INPUT
```

```
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
MODULE user_command_0301 INPUT.

  IF ok_code = 'FC_YES'.
    MODIFY z021employee FROM TABLE lt_empdet.
    IF sy-subrc IS INITIAL.
      MESSAGE 'Details updated successfully' TYPE 'S'.

    ELSE.
      MESSAGE 'Data cannot be updated, please try again' TYPE 'E'.

    ENDIF.
    LEAVE TO SCREEN 0.       "close popup

  ELSEIF ok_code = 'FC_NO'.
    CLEAR lv_emaildisable.
    LEAVE TO SCREEN 0.
  ENDIF.

ENDMODULE.

*&SPWIZARD: INPUT MODULE FOR TS 'TS_ADMIN'. DO NOT CHANGE THIS LINE!
*&SPWIZARD: GETS ACTIVE TAB
MODULE ts_admin_active_tab_get INPUT.
  ok_code = sy-ucomm.
  CASE ok_code.
    WHEN c_ts_admin-tab1.
      g_ts_admin-pressed_tab = c_ts_admin-tab1.
    WHEN c_ts_admin-tab2.
      g_ts_admin-pressed_tab = c_ts_admin-tab2.
    WHEN OTHERS.
*&SPWIZARD:      DO NOTHING
  ENDCASE.
ENDMODULE.
*&---------------------------------------------------------------------*
*&      Module  USER_COMMAND_0110  INPUT
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
MODULE user_command_0110 INPUT.

  DATA: lv_maxemp TYPE z021employee-emp_no,
        lv_pre    TYPE c LENGTH 2,
        lv_post   TYPE string,
        lv_no     TYPE i.

  IF ok_code = 'FC_SUBMIT'.
```

```
    SELECT MAX( emp_no ) FROM z021employee INTO lv_maxemp.
    IF sy-subrc IS NOT INITIAL.
      CLEAR lv_no.
    ELSE.
      SPLIT lv_maxemp AT 'L' INTO lv_pre lv_post.
      lv_no = lv_post.
      lv_no = lv_no + 1.
      lv_post = lv_no.
    ENDIF.

    CONCATENATE 'SL' lv_post INTO ls_ins_emp-emp_no.

    INSERT z021employee FROM ls_ins_emp.
    IF sy-subrc IS INITIAL.
      MESSAGE 'Record inserted' TYPE 'S'.
      CLEAR lv_deltc_fetch.

    ELSE.
      MESSAGE 'Some error occurred, please try again' TYPE 'E'.
    ENDIF.
  ENDIF.

ENDMODULE.

*&SPWIZARD: INPUT MODUL FOR TC 'TC_DEL_EMP'. DO NOT CHANGE THIS LINE!
*&SPWIZARD: MARK TABLE
MODULE tc_del_emp_mark INPUT.
  DATA: g_tc_del_emp_wa2 LIKE LINE OF lt_del_emp.
  IF tc_del_emp-line_sel_mode = 1
  AND ls_del_emp-mark = 'X'.
    LOOP AT lt_del_emp INTO g_tc_del_emp_wa2
      WHERE mark = 'X'.
      g_tc_del_emp_wa2-mark = ''.
      MODIFY lt_del_emp
        FROM g_tc_del_emp_wa2
        TRANSPORTING mark.
    ENDLOOP.
  ENDIF.
  MODIFY lt_del_emp
    FROM ls_del_emp
    INDEX tc_del_emp-current_line
    TRANSPORTING mark.
ENDMODULE.

*&SPWIZARD: INPUT MODULE FOR TC 'TC_DEL_EMP'. DO NOT CHANGE THIS LINE!
*&SPWIZARD: PROCESS USER COMMAND
MODULE tc_del_emp_user_command INPUT.
  ok_code = sy-ucomm.
  PERFORM user_ok_tc USING    'TC_DEL_EMP'
                              'LT_DEL_EMP'
                              'MARK'
                     CHANGING ok_code.
  sy-ucomm = ok_code.
ENDMODULE.
*&---------------------------------------------------------------------*
*&      Module  USER_COMMAND_0402  INPUT
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
MODULE user_command_0402 INPUT.

  IF ok_code = 'FC_DELETE'.

    LOOP AT lt_del_emp INTO ls_del_emp.
      IF ls_del_emp-mark = 'X'.
        DELETE FROM z021employee WHERE emp_no = ls_del_emp-emp_no.
        IF sy-subrc IS INITIAL.
          MESSAGE 'Record deleted' TYPE 'S'.
          CLEAR: ok_code, lv_deltc_fetch.
        ELSE.
          MESSAGE 'Not deleted' TYPE 'E'.
        ENDIF.

      ENDIF.

    ENDLOOP.
  ENDIF.

ENDMODULE.
*&---------------------------------------------------------------------*
*&      Module  VALIDATE_INS  INPUT
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
MODULE validate_ins INPUT.

  IF ok_code = 'FC_SUBMIT'.

    IF ls_ins_emp-emp_name IS INITIAL OR
       ls_ins_emp-gender IS INITIAL OR
       ls_ins_emp-email_id IS INITIAL OR
       ls_ins_emp-dept IS INITIAL.
      MESSAGE 'Please fill all required fields' TYPE 'E'.
    ENDIF.

  ENDIF.
```

```
ENDMODULE.
```

F01

```
*&---------------------------------------------------------------------*
*&      Form  LOGIN_VALIDATE
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
*  -->  p1        text
*  <--  p2        text
*----------------------------------------------------------------------*
FORM login_validate .

  DATA: lv_first    TYPE z021employee-emp_name,
        lv_last     TYPE z021employee-emp_name,
        lv_len      TYPE i,
        lv_password TYPE string.

  SELECT SINGLE emp_name FROM z021employee INTO lv_empname
    WHERE emp_no = io_user.

  IF sy-subrc IS NOT INITIAL.
    MESSAGE 'Invalid Username' TYPE 'E'.

  ELSE.
    SPLIT lv_empname AT ' ' INTO lv_first lv_last.    "Get the first name in
lv_first

    "Get last 3 digits of emp_no field
    lv_len = strlen( io_user ) - 3.
    CONCATENATE lv_first io_user+lv_len(3) INTO lv_password.

    IF io_password = lv_password.
      MESSAGE 'Login Successful' TYPE 'S'.
      CALL SCREEN 0300.

    ELSE.
      MESSAGE 'Invalid Password' TYPE 'E'.

    ENDIF.
  ENDIF.

ENDFORM.
*&---------------------------------------------------------------------*
*&      Form  F4HELP_DEPT
```

```
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
*  -->  p1        text
*  <--  p2        text
*----------------------------------------------------------------------*
FORM f4help_dept .

  TYPES: BEGIN OF ty_dept,
           dept TYPE z021employee-dept,
         END OF ty_dept.

  DATA: lt_dept TYPE STANDARD TABLE OF ty_dept,
        ls_dept TYPE ty_dept.

  ls_dept-dept = 'HR'.
  APPEND ls_dept TO lt_dept.
  CLEAR ls_dept.

  ls_dept-dept = 'Tools'.
  APPEND ls_dept TO lt_dept.
  CLEAR ls_dept.

  ls_dept-dept = 'IT Support'.
  APPEND ls_dept TO lt_dept.
  CLEAR ls_dept.

  ls_dept-dept = 'Development'.
  APPEND ls_dept TO lt_dept.
  CLEAR ls_dept.

  ls_dept-dept = 'Others'.
  APPEND ls_dept TO lt_dept.
  CLEAR ls_dept.

  CALL FUNCTION 'F4IF_INT_TABLE_VALUE_REQUEST'
    EXPORTING
      retfield        = 'DEPT'
      dynpprog        = sy-cprog
      dynpnr          = sy-dynnr
      dynprofield     = 'IO_DEPT'
      value_org       = 'S'
    TABLES
      value_tab       = lt_dept
    EXCEPTIONS
      parameter_error = 1
      no_values_found = 2
      OTHERS          = 3.
  IF sy-subrc <> 0.
    CLEAR lt_dept.
    MESSAGE 'F4 Help cannot be displayed' TYPE 'E'.
  ENDIF.

ENDFORM.

*----------------------------------------------------------------------*
*   INCLUDE TABLECONTROL_FORMS                                         *
*----------------------------------------------------------------------*

*&---------------------------------------------------------------------*
*&      Form  USER_OK_TC                                               *
*&---------------------------------------------------------------------*
 FORM USER_OK_TC USING    P_TC_NAME TYPE DYNFNAM
                          P_TABLE_NAME
                          P_MARK_NAME
                 CHANGING P_OK      LIKE SY-UCOMM.

*&SPWIZARD: BEGIN OF LOCAL DATA----------------------------------------*
   DATA: L_OK              TYPE SY-UCOMM,
         L_OFFSET          TYPE I.
*&SPWIZARD: END OF LOCAL DATA------------------------------------------*

*&SPWIZARD: Table control specific operations                          *
*&SPWIZARD: evaluate TC name and operations                            *
   SEARCH P_OK FOR P_TC_NAME.
   IF SY-SUBRC <> 0.
     EXIT.
   ENDIF.
   L_OFFSET = STRLEN( P_TC_NAME ) + 1.
   L_OK = P_OK+L_OFFSET.
*&SPWIZARD: execute general and TC specific operations                 *
   CASE L_OK.
     WHEN 'INSR'.                      "insert row
       PERFORM FCODE_INSERT_ROW USING    P_TC_NAME
                                         P_TABLE_NAME.
       CLEAR P_OK.

     WHEN 'DELE'.                      "delete row
       PERFORM FCODE_DELETE_ROW USING    P_TC_NAME
                                         P_TABLE_NAME
                                         P_MARK_NAME.
       CLEAR P_OK.

     WHEN 'P--' OR                     "top of list
          'P-'  OR                     "previous page
          'P+'  OR                     "next page
          'P++'.                       "bottom of list
       PERFORM COMPUTE_SCROLLING_IN_TC USING P_TC_NAME
                                             L_OK.
       CLEAR P_OK.
*     WHEN 'L--'.                         "total left
*       PERFORM FCODE_TOTAL_LEFT USING P_TC_NAME.
*
*     WHEN 'L-'.                          "column left
*       PERFORM FCODE_COLUMN_LEFT USING P_TC_NAME.
*
*     WHEN 'R+'.                          "column right
*       PERFORM FCODE_COLUMN_RIGHT USING P_TC_NAME.
*
*     WHEN 'R++'.                         "total right
*       PERFORM FCODE_TOTAL_RIGHT USING P_TC_NAME.
*
     WHEN 'MARK'.                        "mark all filled lines
       PERFORM FCODE_TC_MARK_LINES USING P_TC_NAME
                                         P_TABLE_NAME
                                         P_MARK_NAME   .
       CLEAR P_OK.

     WHEN 'DMRK'.                        "demark all filled lines
       PERFORM FCODE_TC_DEMARK_LINES USING P_TC_NAME
                                           P_TABLE_NAME
                                           P_MARK_NAME .
       CLEAR P_OK.

*     WHEN 'SASCEND'   OR
*          'SDESCEND'.                    "sort column
*       PERFORM FCODE_SORT_TC USING P_TC_NAME
*                                   l_ok.

   ENDCASE.

 ENDFORM.                                " USER_OK_TC

*&---------------------------------------------------------------------*
*&      Form  FCODE_INSERT_ROW                                          *
*&---------------------------------------------------------------------*
 FORM fcode_insert_row
              USING    P_TC_NAME           TYPE DYNFNAM
                       P_TABLE_NAME             .

*&SPWIZARD: BEGIN OF LOCAL DATA----------------------------------------*
   DATA L_LINES_NAME       LIKE FELD-NAME.
   DATA L_SELLINE          LIKE SY-STEPL.
   DATA L_LASTLINE         TYPE I.
   DATA L_LINE             TYPE I.
   DATA L_TABLE_NAME       LIKE FELD-NAME.
   FIELD-SYMBOLS <TC>                  TYPE CXTAB_CONTROL.
   FIELD-SYMBOLS <TABLE>               TYPE STANDARD TABLE.
   FIELD-SYMBOLS <LINES>               TYPE I.
*&SPWIZARD: END OF LOCAL DATA------------------------------------------*

   ASSIGN (P_TC_NAME) TO <TC>.

*&SPWIZARD: get the table, which belongs to the tc                     *
   CONCATENATE P_TABLE_NAME '[]' INTO L_TABLE_NAME. "table body
   ASSIGN (L_TABLE_NAME) TO <TABLE>.                "not headerline

*&SPWIZARD: get looplines of TableControl                              *
   CONCATENATE 'G_' P_TC_NAME '_LINES' INTO L_LINES_NAME.
   ASSIGN (L_LINES_NAME) TO <LINES>.

*&SPWIZARD: get current line                                           *
   GET CURSOR LINE L_SELLINE.
   IF SY-SUBRC <> 0.                   " append line to table
     L_SELLINE = <TC>-LINES + 1.
*&SPWIZARD: set top line                                               *
     IF L_SELLINE > <LINES>.
       <TC>-TOP_LINE = L_SELLINE - <LINES> + 1 .
     ELSE.
       <TC>-TOP_LINE = 1.
     ENDIF.
   ELSE.                               " insert line into table
     L_SELLINE = <TC>-TOP_LINE + L_SELLINE - 1.
     L_LASTLINE = <TC>-TOP_LINE + <LINES> - 1.
   ENDIF.
*&SPWIZARD: set new cursor line                                        *
   L_LINE = L_SELLINE - <TC>-TOP_LINE + 1.

*&SPWIZARD: insert initial line                                        *
   INSERT INITIAL LINE INTO <TABLE> INDEX L_SELLINE.
   <TC>-LINES = <TC>-LINES + 1.
*&SPWIZARD: set cursor                                                 *
   SET CURSOR LINE L_LINE.

 ENDFORM.                              " FCODE_INSERT_ROW

*&---------------------------------------------------------------------*
*&      Form  FCODE_DELETE_ROW                                         *
*&---------------------------------------------------------------------*
 FORM fcode_delete_row
               USING    P_TC_NAME           TYPE DYNFNAM
                        P_TABLE_NAME
                        P_MARK_NAME   .

*&SPWIZARD: BEGIN OF LOCAL DATA----------------------------------------*
```

```
  DATA L_TABLE_NAME       LIKE FELD-NAME.

  FIELD-SYMBOLS <TC>         TYPE cxtab_control.
  FIELD-SYMBOLS <TABLE>      TYPE STANDARD TABLE.
  FIELD-SYMBOLS <WA>.
  FIELD-SYMBOLS <MARK_FIELD>.
*&SPWIZARD: END OF LOCAL DATA------------------------------------------*

  ASSIGN (P_TC_NAME) TO <TC>.

*&SPWIZARD: get the table, which belongs to the tc                     *
  CONCATENATE P_TABLE_NAME '[]' INTO L_TABLE_NAME. "table body
  ASSIGN (L_TABLE_NAME) TO <TABLE>.                "not headerline

*&SPWIZARD: delete marked lines                                        *
  DESCRIBE TABLE <TABLE> LINES <TC>-LINES.

  LOOP AT <TABLE> ASSIGNING <WA>.

*&SPWIZARD: access to the component 'FLAG' of the table header         *
    ASSIGN COMPONENT P_MARK_NAME OF STRUCTURE <WA> TO <MARK_FIELD>.

    IF <MARK_FIELD> = 'X'.
      DELETE <TABLE> INDEX SYST-TABIX.
      IF SY-SUBRC = 0.
        <TC>-LINES = <TC>-LINES - 1.
      ENDIF.
    ENDIF.
  ENDLOOP.

ENDFORM.                              " FCODE_DELETE_ROW

*&---------------------------------------------------------------------*
*&      Form  COMPUTE_SCROLLING_IN_TC
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
*      -->P_TC_NAME  name of tablecontrol
*      -->P_OK       ok code
*----------------------------------------------------------------------*
FORM COMPUTE_SCROLLING_IN_TC USING    P_TC_NAME
                                      P_OK.
*&SPWIZARD: BEGIN OF LOCAL DATA----------------------------------------*
  DATA L_TC_NEW_TOP_LINE     TYPE I.
  DATA L_TC_NAME             LIKE FELD-NAME.
  DATA L_TC_LINES_NAME       LIKE FELD-NAME.
  DATA L_TC_FIELD_NAME       LIKE FELD-NAME.

  FIELD-SYMBOLS <TC>         TYPE cxtab_control.
```

```
   FIELD-SYMBOLS <LINES>      TYPE I.
*&SPWIZARD: END OF LOCAL DATA------------------------------------------*

   ASSIGN (P_TC_NAME) TO <TC>.
*&SPWIZARD: get looplines of TableControl                              *
   CONCATENATE 'G_' P_TC_NAME '_LINES' INTO L_TC_LINES_NAME.
   ASSIGN (L_TC_LINES_NAME) TO <LINES>.


*&SPWIZARD: is no line filled?                                         *
   IF <TC>-LINES = 0.
*&SPWIZARD: yes, ...                                                   *
     L_TC_NEW_TOP_LINE = 1.
   ELSE.
*&SPWIZARD: no, ...                                                    *
     CALL FUNCTION 'SCROLLING_IN_TABLE'
          EXPORTING
               ENTRY_ACT             = <TC>-TOP_LINE
               ENTRY_FROM            = 1
               ENTRY_TO              = <TC>-LINES
               LAST_PAGE_FULL        = 'X'
               LOOPS                 = <LINES>
               OK_CODE               = P_OK
               OVERLAPPING           = 'X'
          IMPORTING
               ENTRY_NEW             = L_TC_NEW_TOP_LINE
          EXCEPTIONS
*              NO_ENTRY_OR_PAGE_ACT  = 01
*              NO_ENTRY_TO           = 02
*              NO_OK_CODE_OR_PAGE_GO = 03
               OTHERS                = 0.
   ENDIF.

*&SPWIZARD: get actual tc and column                                   *
   GET CURSOR FIELD L_TC_FIELD_NAME
              AREA  L_TC_NAME.

   IF SYST-SUBRC = 0.
     IF L_TC_NAME = P_TC_NAME.
*&SPWIZARD: et actual column                                           *
       SET CURSOR FIELD L_TC_FIELD_NAME LINE 1.
     ENDIF.
   ENDIF.

*&SPWIZARD: set the new top line                                       *
   <TC>-TOP_LINE = L_TC_NEW_TOP_LINE.


 ENDFORM.                              " COMPUTE_SCROLLING_IN_TC
```

```
*&---------------------------------------------------------------------*
*&      Form  FCODE_TC_MARK_LINES
*&---------------------------------------------------------------------*
*       marks all TableControl lines
*----------------------------------------------------------------------*
*      -->P_TC_NAME  name of tablecontrol
*----------------------------------------------------------------------*
FORM FCODE_TC_MARK_LINES USING P_TC_NAME
                               P_TABLE_NAME
                               P_MARK_NAME.
*&SPWIZARD: EGIN OF LOCAL DATA-----------------------------------------*
  DATA L_TABLE_NAME       LIKE FELD-NAME.

  FIELD-SYMBOLS <TC>         TYPE cxtab_control.
  FIELD-SYMBOLS <TABLE>      TYPE STANDARD TABLE.
  FIELD-SYMBOLS <WA>.
  FIELD-SYMBOLS <MARK_FIELD>.
*&SPWIZARD: END OF LOCAL DATA------------------------------------------*

  ASSIGN (P_TC_NAME) TO <TC>.

*&SPWIZARD: get the table, which belongs to the tc                     *
   CONCATENATE P_TABLE_NAME '[]' INTO L_TABLE_NAME. "table body
   ASSIGN (L_TABLE_NAME) TO <TABLE>.                "not headerline

*&SPWIZARD: mark all filled lines                                      *
  LOOP AT <TABLE> ASSIGNING <WA>.

*&SPWIZARD: access to the component 'FLAG' of the table header         *
     ASSIGN COMPONENT P_MARK_NAME OF STRUCTURE <WA> TO <MARK_FIELD>.

     <MARK_FIELD> = 'X'.
  ENDLOOP.
ENDFORM.                                          "fcode_tc_mark_lines

*&---------------------------------------------------------------------*
*&      Form  FCODE_TC_DEMARK_LINES
*&---------------------------------------------------------------------*
*       demarks all TableControl lines
*----------------------------------------------------------------------*
*      -->P_TC_NAME  name of tablecontrol
*----------------------------------------------------------------------*
FORM FCODE_TC_DEMARK_LINES USING P_TC_NAME
                                 P_TABLE_NAME
                                 P_MARK_NAME .
*&SPWIZARD: BEGIN OF LOCAL DATA----------------------------------------*
  DATA L_TABLE_NAME       LIKE FELD-NAME.
  FIELD-SYMBOLS <TC>         TYPE cxtab_control.
  FIELD-SYMBOLS <TABLE>      TYPE STANDARD TABLE.
  FIELD-SYMBOLS <WA>.
  FIELD-SYMBOLS <MARK_FIELD>.
*&SPWIZARD: END OF LOCAL DATA------------------------------------------*

  ASSIGN (P_TC_NAME) TO <TC>.

*&SPWIZARD: get the table, which belongs to the tc                     *
   CONCATENATE P_TABLE_NAME '[]' INTO L_TABLE_NAME. "table body
   ASSIGN (L_TABLE_NAME) TO <TABLE>.                "not headerline

*&SPWIZARD: demark all filled lines                                    *
  LOOP AT <TABLE> ASSIGNING <WA>.

*&SPWIZARD: access to the component 'FLAG' of the table header         *
     ASSIGN COMPONENT P_MARK_NAME OF STRUCTURE <WA> TO <MARK_FIELD>.

     <MARK_FIELD> = SPACE.
  ENDLOOP.
ENDFORM.                                          "fcode_tc_mark_lines
```