REPORT  YKK\_PRJ\_REP NO STANDARD PAGE HEADING  
                                   LINE-SIZE 170  
                                   LINE-COUNT 58  
                                   MESSAGE-ID ZPSDVRP..  
TABLES: PROJ,PRPS.  
\*-----------------------------------------------------------------------  
\* T Y P E S  
\*-----------------------------------------------------------------------  
TYPES:  
  BEGIN OF \_PROJ,  
    VBUKR TYPE PS\_VBUKR,  
    VKOKR TYPE PS\_VKOKR,  
    PSPNR TYPE PS\_INTNR,  
    PSPID TYPE PS\_PSPID,  
    POST1 TYPE PS\_POST1,  
    PLSEZ TYPE PS\_PLSEZ,  
    OBJNR TYPE J\_OBJNR,  
  END OF \_PROJ,  
  
  BEGIN OF \_PRPS,  
    PSPNR TYPE PS\_POSNR,  
    POSID TYPE PS\_POSID,  
    POSKI TYPE PS\_POSKI,  
    PRCTR TYPE PRCTR,  
    PRART TYPE PS\_PRART,  
    POST1 TYPE PS\_POST1,  
    PSPHI TYPE PS\_PSPHI,  
    OBJNR TYPE J\_OBJNR,  
    USR00 TYPE USR00PRPS,  
    ERDAT TYPE ERDAT,  
    STUFE TYPE PS\_STUFE,  
  END OF \_PRPS,  
  
  BEGIN OF \_RPSCO,  
    OBJNR TYPE J\_OBJNR,  
    WRTTP TYPE CO\_WRTTP,  
    GJAHR TYPE GJAHR,  
    VORGA TYPE BP\_VORGANG,  
    VERSN TYPE BP\_VERSION,  
    BELTP TYPE BP\_INOUT,  
    WLP00 TYPE BP\_WPL,  
  END OF \_RPSCO,  
  
  BEGIN OF \_OUTPUT,  
    PROJ\_PSPID TYPE PS\_PSPID,  
    PROJ\_POST1 TYPE PS\_POST1,  
    PROJ\_PLSEZ TYPE PS\_PLSEZ,  
    SLS\_PRICE  TYPE BP\_WPL,  
    PRPS\_USR00 TYPE USR00PRPS,  
    POSID      TYPE PS\_POSID,  
    PRPS\_POST1 TYPE PS\_POST1,  
    ORIG\_COST  TYPE BP\_WPL,  
    CURR\_BUDG  TYPE BP\_WPL,  
    END\_COST   TYPE BP\_WPL,  
    ORDR\_COST  TYPE BP\_WPL,  
    ACTL\_COST  TYPE BP\_WPL,  
    ORDR\_PERC(20)  TYPE C,  
    CURR\_ACTL  TYPE BP\_WPL,  
    STATUS     TYPE J\_INACT,  
    PROJ\_TYPE  TYPE PS\_PRATX,  
    COMP\_NAME  TYPE NAME\_1,  
    TASK\_STAT  TYPE J\_STEXT,  
    PROJ\_STAT  TYPE J\_STEXT,  
    TASK\_DATE  TYPE ERDAT,  
    VBUKR TYPE PS\_VBUKR,  
    VKOKR TYPE PS\_VKOKR,  
    PSPNR TYPE PS\_INTNR,  
    PRPS\_POSKI TYPE PS\_POSKI,  
    PRCTR TYPE PRCTR,  
    PSPHI TYPE PS\_PSPHI,  
    OBJNR TYPE J\_OBJNR,  
    KUNNR TYPE NAME1\_GP,  
    PARVW TYPE VTXTK,  
  END OF \_OUTPUT.  
CONSTANTS:C\_MESS\_CL   TYPE SY-MSGID        VALUE 'ZPSDVRP',  
          C\_E         TYPE BAPIRETURN-TYPE VALUE 'E',  
          C\_A         TYPE BAPIRETURN-TYPE VALUE 'A',  
          C\_I         TYPE BAPIRETURN-TYPE VALUE 'I',  
          C\_X         TYPE C               VALUE 'X',  
          C\_W         TYPE BAPIRETURN-TYPE VALUE 'W',  
          C\_YES       TYPE C               VALUE 'X',  
          C\_NO        TYPE C               VALUE '',  
          C\_CSV(5)                         VALUE '\*.CSV',  
          C\_SEP       TYPE C               VALUE ',',  
          C\_PERIOD    TYPE C               VALUE '.',  
          C\_COMMA     TYPE C               VALUE ',',  
          C\_1         TYPE C               VALUE '1',  
          C\_%(1)      TYPE C               VALUE '%',  
          C\_01(2)     TYPE C               VALUE '01',  
          C\_21(2)     TYPE C               VALUE '21',  
          C\_02(2)     TYPE C               VALUE '02',  
          C\_04(2)     TYPE C               VALUE '04',  
          C\_22(2)     TYPE C               VALUE '22',  
          C\_42(2)     TYPE C               VALUE '42',  
          C\_2         TYPE C               VALUE '2',  
          C\_001(3)                         VALUE '001',  
          C\_002(3)                         VALUE '002',  
          C\_I0067(5)                       VALUE 'I0067',  
          C\_I0043(5)                       VALUE 'I0043',  
          C\_I0045(5)                       VALUE 'I0045',  
          C\_I0046(5)                       VALUE 'I0046',  
          C\_ZERO(3)   TYPE C               VALUE '000',  
          C\_0000(4)   TYPE C               VALUE '0000',  
          C\_NUMC(4)                        VALUE 'NUMC',  
          C\_KSTR(4)   TYPE C               VALUE 'KSTR',  
          C\_KSTP(4)   TYPE C               VALUE 'KSTP'.  
\*----------------------------------------------------------------------\*  
\* SELECTION-SCREEN  
\*----------------------------------------------------------------------\*  
  
SELECTION-SCREEN BEGIN OF SCREEN 100.  
SELECTION-SCREEN BEGIN OF BLOCK SELECTION  
                        WITH FRAME TITLE TEXT-001.  
PARAMETERS:  
  P\_VBUKR   TYPE PROJ-VBUKR OBLIGATORY,  
  P\_VKOKR   TYPE PROJ-VKOKR OBLIGATORY,  
  P\_STAT    TYPE JEST-STAT.  
  
SELECT-OPTIONS:  
  S\_PSPID   FOR PROJ-PSPID,  
  S\_POSID   FOR PRPS-POSID,  
  S\_POSKI   FOR PRPS-POSKI,  
  S\_PRCTR   FOR PRPS-PRCTR,  
  S\_PRART   FOR PRPS-PRART.  
SELECTION-SCREEN END OF BLOCK SELECTION.  
  
SELECTION-SCREEN BEGIN OF BLOCK DOWNLOAD  
                          WITH FRAME TITLE TEXT-002.  
PARAMETERS:  
  P\_EXPATH TYPE RLGRAP-FILENAME.  
SELECTION-SCREEN END OF BLOCK DOWNLOAD.  
SELECTION-SCREEN END OF SCREEN 100.  
  
INCLUDE  ZCLASS\_LOG\_UTILITY.  
  
CLASS CL\_VIEW DEFINITION.  
  PUBLIC SECTION.  
    METHODS:  
      DISPLAY\_GRID IMPORTING T\_OP TYPE STANDARD TABLE,  
  
      F4\_HELP\_FILENAME CHANGING VALUE(EXPATH) TYPE RLGRAP-FILENAME,  
      GET\_PARAMS EXPORTING  
                  VALUE(VBUKR) TYPE PROJ-VBUKR  
                  VALUE(VKOKR) TYPE PROJ-VKOKR  
                  VALUE(STAT)  TYPE JEST-STAT  
                  VALUE(EXPATH) TYPE RLGRAP-FILENAME  
                  VALUE(PSPID) TYPE RSELOPTION  
                  VALUE(POSID) TYPE RSELOPTION  
                  VALUE(POSKI) TYPE RSELOPTION  
                  VALUE(PRCTR) TYPE RSELOPTION  
                  VALUE(PRART) TYPE RSELOPTION ,  
      SAVE\_FILE EXPORTING VALUE(T\_LOG) TYPE BAPIRET2\_T ,  
      VALIDATE\_FILENAME EXPORTING VALUE(T\_LOG) TYPE BAPIRET2\_T,  
      WRITE\_LOG IMPORTING VALUE(EXPATH) TYPE RLGRAP-FILENAME  
                EXPORTING VALUE(T\_LOG) TYPE BAPIRET2\_T .  
  
    DATA:  
      R\_GRID     TYPE REF TO CL\_SALV\_TABLE,  
      T\_OUTPUT   TYPE TABLE OF \_OUTPUT,  
      I\_RESULT   TYPE TABLE OF SOLISTI1.  
  PRIVATE SECTION.  
    METHODS:  BUILD\_GRID,  
              CREATE\_CSV\_HDR,  
              CREATE\_GRID,  
              CREATE\_HEADER,  
              EDIT\_COLUMNS,  
              GET\_FUNCTIONS.  
ENDCLASS.  
  
CLASS CL\_VIEW IMPLEMENTATION.  
  
  METHOD F4\_HELP\_FILENAME.  
    CALL FUNCTION 'KD\_GET\_FILENAME\_ON\_F4'  
      EXPORTING  
        PROGRAM\_NAME  = SYST-REPID  
        DYNPRO\_NUMBER = SYST-DYNNR  
        FIELD\_NAME    = EXPATH  
\*       STATIC        = ' '  
\*       MASK          = ' '  
\*       FILEOPERATION = 'R'  
      CHANGING  
        FILE\_NAME     = EXPATH  
\*       LOCATION\_FLAG = 'P'  
      EXCEPTIONS  
        MASK\_TOO\_LONG = 1  
        OTHERS        = 2.  
    IF SY-SUBRC <> 0.  
    ELSE.  
      IF EXPATH IS NOT INITIAL.  
        TRANSLATE EXPATH TO UPPER CASE.                   "#EC SYNTCHAR  
        IF EXPATH CP C\_CSV.  
        ELSE.  
          CONCATENATE EXPATH C\_CSV INTO EXPATH.  
        ENDIF.  
      ENDIF.  
    ENDIF.  
  ENDMETHOD.  
  METHOD VALIDATE\_FILENAME.  
    CONSTANTS:  
      C\_CSV2(3)    VALUE 'CSV'.  
    DATA:  
      LV\_DIR     TYPE STRING,     " For Directory  
      LV\_BOL     TYPE ABAP\_BOOL,  " Result  
      LV\_FNAME   TYPE STRING,     " File Name  
      LV\_FNAME1  TYPE STRING,     " File Name  
      LV\_ERR\_MESS TYPE SYMSGV,  
      LV\_EXT     TYPE STRING.     " File Extension  
  
  
    CALL FUNCTION 'SO\_SPLIT\_FILE\_AND\_PATH'  
      EXPORTING  
        FULL\_NAME     = P\_EXPATH  
      IMPORTING  
        STRIPPED\_NAME = LV\_FNAME  
        FILE\_PATH     = LV\_DIR  
      EXCEPTIONS  
        X\_ERROR       = 1  
        OTHERS        = 2.  
    IF SY-SUBRC <> 0.  
      LCL\_LOG\_UTILITY=>POPULATE\_LOG( EXPORTING  
                                       I\_TYPE   = C\_I  
                                       I\_CL     = C\_MESS\_CL  
                                       I\_NUMBER = 003  
                                       I\_PAR1   = SY-MSGV1  
                                       I\_PAR2   = SY-MSGV2  
                                       I\_PAR3   = SY-MSGV3  
                                       I\_PAR4   = SY-MSGV4  
                                     CHANGING  
                                       C\_LOG    = T\_LOG ).  
    ENDIF.  
  
    CALL METHOD CL\_GUI\_FRONTEND\_SERVICES=>DIRECTORY\_EXIST  
      EXPORTING  
        DIRECTORY            = LV\_DIR  
      RECEIVING  
        RESULT               = LV\_BOL  
      EXCEPTIONS  
        CNTL\_ERROR           = 1  
        ERROR\_NO\_GUI         = 2  
        WRONG\_PARAMETER      = 3  
        NOT\_SUPPORTED\_BY\_GUI = 4  
        OTHERS               = 5.  
    IF SY-SUBRC = 0.  
    ELSE.  
      IF LV\_BOL IS INITIAL.  
        LV\_ERR\_MESS = LV\_DIR.  
        LCL\_LOG\_UTILITY=>POPULATE\_LOG( EXPORTING  
                                         I\_TYPE   = C\_E  
                                         I\_CL     = C\_MESS\_CL  
                                         I\_NUMBER = 015  
                                         I\_PAR1   = LV\_ERR\_MESS  
                                       CHANGING  
                                         C\_LOG    = T\_LOG ).  
      ELSE.  
        TRANSLATE LV\_FNAME TO UPPER CASE.                 "#EC SYNTCHAR  
        SPLIT LV\_FNAME AT '.' INTO LV\_FNAME1 LV\_EXT.  
        IF LV\_EXT IS INITIAL.  
          LV\_ERR\_MESS = LV\_DIR.  
          LCL\_LOG\_UTILITY=>POPULATE\_LOG( EXPORTING  
                                           I\_TYPE   = C\_E  
                                           I\_CL     = C\_MESS\_CL  
                                           I\_NUMBER = 016  
                                         CHANGING  
                                           C\_LOG    = T\_LOG ).  
        ELSE.  
          IF LV\_EXT CP C\_CSV OR LV\_EXT EQ C\_CSV2.  
  
          ELSE.  
            LV\_ERR\_MESS = LV\_EXT.  
            LCL\_LOG\_UTILITY=>POPULATE\_LOG( EXPORTING  
                                             I\_TYPE   = C\_E  
                                             I\_CL     = C\_MESS\_CL  
                                             I\_NUMBER = 017  
                                             I\_PAR1   = LV\_ERR\_MESS  
                                           CHANGING  
                                             C\_LOG    = T\_LOG ).  
  
          ENDIF.  
        ENDIF.  
      ENDIF.  
    ENDIF.  
  ENDMETHOD.  
  METHOD BUILD\_GRID.  
    EDIT\_COLUMNS( ).  
    GET\_FUNCTIONS( ).  
    CREATE\_HEADER( ).  
  ENDMETHOD.  
  METHOD CREATE\_CSV\_HDR.  
  
    FIELD-SYMBOLS:  
      <FS\_RESULT> TYPE SOLISTI1.  
  
    APPEND INITIAL LINE TO I\_RESULT ASSIGNING <FS\_RESULT>.  
    CONCATENATE  
      'Project'  
      'Project Description'  
      'Task'  
      'Task Description'  
      'Partner Function'  
      'Customer'  
      'Department Total Code'  
      'Original Cost'  
      'Current Budget'  
      'Ordered Cost'  
      'Actual Cost'  
      'Projected End Cost'  
      'Division'  
      'Snapshot Date'  
      'Project Status'  
      'Task Status'  
      'Task Date' INTO <FS\_RESULT>-LINE SEPARATED BY C\_SEP.  
  
  ENDMETHOD.  
  METHOD CREATE\_GRID.  
    TRY.  
        CL\_SALV\_TABLE=>FACTORY( IMPORTING  
                                  R\_SALV\_TABLE = R\_GRID  
                                CHANGING  
                                  T\_TABLE = T\_OUTPUT ).  
      CATCH CX\_SALV\_MSG.  
    ENDTRY.  
  ENDMETHOD.  
  METHOD CREATE\_HEADER.  
  
    DATA:  
      LR\_CONTENT\_HDR TYPE REF TO CL\_SALV\_FORM\_ELEMENT,  
      LR\_CONTENT TYPE REF TO CL\_SALV\_FORM\_ELEMENT,  
      LR\_HEADER  TYPE REF TO CL\_SALV\_FORM\_HEADER\_INFO,  
      LR\_LAYOUT\_DATA\_GRID TYPE REF TO CL\_SALV\_FORM\_LAYOUT\_DATA\_GRID,  
      LR\_LAYOUT\_LABEL\_GRID TYPE REF TO CL\_SALV\_FORM\_LAYOUT\_DATA\_GRID,  
      L\_TEXT     TYPE STRING,  
      LR\_GRID    TYPE REF TO CL\_SALV\_FORM\_LAYOUT\_GRID,  
      LR\_GRID\_1  TYPE REF TO CL\_SALV\_FORM\_LAYOUT\_GRID,  
      LR\_GRID\_2  TYPE REF TO CL\_SALV\_FORM\_LAYOUT\_GRID,  
      LR\_LABEL   TYPE REF TO CL\_SALV\_FORM\_LABEL,  
      V\_LINES    TYPE I,  
      LR\_TEXT    TYPE REF TO CL\_SALV\_FORM\_TEXT.  
  
    DATA:  
      S\_OUTPUT  TYPE \_OUTPUT.  
  
    READ TABLE T\_OUTPUT INTO S\_OUTPUT INDEX 1.  
  
  
    CREATE OBJECT LR\_GRID.  
    LR\_GRID->CREATE\_HEADER\_INFORMATION( ROW     = 1  
                                        COLUMN  = 2  
                                        TEXT    = 'V\_COMPANY\_NAME' ).  
    "colspan = 2 ).  
    LR\_GRID\_1 = LR\_GRID->CREATE\_GRID(  
                                      ROW    = 2  
                                      COLUMN = 1 ).  
  
  
    LR\_LABEL = LR\_GRID\_1->CREATE\_LABEL( ROW     = 2  
                                        COLUMN  = 1  
                                        TEXT    = 'Report Name: ' ).  
    LR\_LAYOUT\_LABEL\_GRID ?= LR\_LABEL->GET\_LAYOUT\_DATA( ).  
    LR\_LAYOUT\_LABEL\_GRID->SET\_WIDTH( '20' ).  
  
    LR\_GRID\_1->CREATE\_TEXT( ROW     = 2  
                            COLUMN  = 6  
                            TEXT    = ' ' ).  
    LR\_TEXT = LR\_GRID\_1->CREATE\_TEXT( ROW     = 2  
                                       COLUMN  = 2  
                                       TEXT    = SY-REPID  
                                       TOOLTIP = 'Report Name' ).  
  
    LR\_LABEL = LR\_GRID\_1->CREATE\_LABEL( ROW     = 3  
                                        COLUMN  = 1  
                                        TEXT    = 'Run Date: ' ).  
  
    LR\_TEXT = LR\_GRID\_1->CREATE\_TEXT( ROW     = 3  
                                      COLUMN  = 2  
                                      TEXT    = SY-DATUM  
                                      TOOLTIP = 'Run Date' ).  
\*    lr\_label->set\_label\_for( lr\_text ).  
    LR\_LABEL = LR\_GRID\_1->CREATE\_LABEL( ROW     = 4  
                                        COLUMN  = 1  
                                        TEXT    = 'Run Time: ' ).  
  
    LR\_TEXT = LR\_GRID\_1->CREATE\_TEXT( ROW     = 4  
                                      COLUMN  = 2  
                                      TEXT    = SY-UZEIT  
                                      TOOLTIP = 'Run Time' ).  
  
    LR\_LABEL = LR\_GRID\_1->CREATE\_LABEL( ROW     = 4  
                                        COLUMN  = 3  
                                        TEXT    = 'Beginning Project: ' ).  
  
    LR\_TEXT = LR\_GRID\_1->CREATE\_TEXT( ROW     = 4  
                                      COLUMN  = 4  
                                      TEXT    = S\_OUTPUT-PROJ\_PSPID  
                                      TOOLTIP = 'Project' ).  
  
    LR\_LABEL = LR\_GRID\_1->CREATE\_LABEL( ROW     = 5  
                                        COLUMN  = 1  
                                        TEXT    = 'Project Type: ' ).  
  
  
    LR\_TEXT = LR\_GRID\_1->CREATE\_TEXT( ROW     = 5  
                                      COLUMN  = 2  
                                      TEXT    = S\_OUTPUT-PROJ\_TYPE  
                                      TOOLTIP = 'Project Type' ).  
  
  
    DESCRIBE TABLE T\_OUTPUT LINES V\_LINES.  
    READ TABLE T\_OUTPUT INTO S\_OUTPUT INDEX V\_LINES.  
    LR\_LABEL = LR\_GRID\_1->CREATE\_LABEL( ROW     = 5  
                                        COLUMN  = 3  
                                        TEXT    = 'Ending Project: ' ).  
  
    LR\_TEXT = LR\_GRID\_1->CREATE\_TEXT( ROW     = 5  
                                      COLUMN  = 4  
                                      TEXT    = S\_OUTPUT-PROJ\_PSPID  
                                      TOOLTIP = 'Project' ).  
  
    LR\_CONTENT = LR\_GRID.  
    R\_GRID->SET\_TOP\_OF\_LIST( LR\_CONTENT ).  
  
  ENDMETHOD.  
  METHOD DISPLAY\_GRID.  
    T\_OUTPUT[] = T\_OP[].  
    CREATE\_GRID( ).  
    BUILD\_GRID( ).  
    R\_GRID->DISPLAY( ).  
  
  ENDMETHOD.  
  METHOD EDIT\_COLUMNS.  
    DATA:  
      COLUMNS TYPE REF TO CL\_SALV\_COLUMNS\_TABLE,  
      COLUMN  TYPE REF TO CL\_SALV\_COLUMN.  
  
    TRY .  
        COLUMNS = R\_GRID->GET\_COLUMNS( ).  
  
        COLUMN = COLUMNS->GET\_COLUMN( COLUMNNAME = 'PROJ\_PSPID' ).  
        COLUMN->SET\_OUTPUT\_LENGTH( '12' ).  
        COLUMN->SET\_LONG\_TEXT( 'Order #' ).  
        COLUMN->SET\_MEDIUM\_TEXT( 'Order #' ).  
        COLUMN->SET\_SHORT\_TEXT( 'Order #' ).  
  
        COLUMN = COLUMNS->GET\_COLUMN( COLUMNNAME = 'PROJ\_POST1' ).  
        COLUMN->SET\_LONG\_TEXT( 'Project' ).  
        COLUMN->SET\_MEDIUM\_TEXT( 'Project' ).  
        COLUMN->SET\_SHORT\_TEXT( 'Project' ).  
  
        COLUMN = COLUMNS->GET\_COLUMN( COLUMNNAME = 'PROJ\_PLSEZ' ).  
        COLUMN->SET\_OUTPUT\_LENGTH( '12' ).  
        COLUMN->SET\_LONG\_TEXT( 'Project End Date' ).  
        COLUMN->SET\_MEDIUM\_TEXT( 'Proj End Dt.' ).  
        COLUMN->SET\_SHORT\_TEXT( 'Proj End' ).  
  
        COLUMN = COLUMNS->GET\_COLUMN( COLUMNNAME = 'SLS\_PRICE' ).  
        COLUMN->SET\_OUTPUT\_LENGTH( '16' ).  
        COLUMN->SET\_LONG\_TEXT( 'Sales Price' ).  
        COLUMN->SET\_MEDIUM\_TEXT( 'Sales Price' ).  
        COLUMN->SET\_SHORT\_TEXT( 'Sles Price' ).  
  
        COLUMN = COLUMNS->GET\_COLUMN( COLUMNNAME = 'PRPS\_USR00' ).  
        COLUMN->SET\_OUTPUT\_LENGTH( '7' ).  
        COLUMN->SET\_LONG\_TEXT( 'Dept Code' ).  
        COLUMN->SET\_MEDIUM\_TEXT( 'Dept Code' ).  
        COLUMN->SET\_SHORT\_TEXT( 'Dept Code' ).  
  
        COLUMN = COLUMNS->GET\_COLUMN( COLUMNNAME = 'POSID' ).  
        COLUMN->SET\_OUTPUT\_LENGTH( '12' ).  
        COLUMN->SET\_LONG\_TEXT( 'Task #' ).  
        COLUMN->SET\_MEDIUM\_TEXT( 'Task #' ).  
        COLUMN->SET\_SHORT\_TEXT( 'Task #' ).  
  
        COLUMN = COLUMNS->GET\_COLUMN( COLUMNNAME = 'PRPS\_POST1' ).  
        COLUMN->SET\_OPTIMIZED( ).  
        COLUMN->SET\_LONG\_TEXT( 'Task Description' ).  
        COLUMN->SET\_MEDIUM\_TEXT( 'Task Descrpt' ).  
        COLUMN->SET\_SHORT\_TEXT( 'Task Des.' ).  
  
        COLUMN = COLUMNS->GET\_COLUMN( COLUMNNAME = 'ORIG\_COST' ).  
        COLUMN->SET\_OPTIMIZED( ).  
        COLUMN->SET\_LONG\_TEXT( 'Original Cost' ).  
        COLUMN->SET\_MEDIUM\_TEXT( 'Orig Cost' ).  
        COLUMN->SET\_SHORT\_TEXT( 'Orig Cost' ).  
  
        COLUMN = COLUMNS->GET\_COLUMN( COLUMNNAME = 'CURR\_BUDG' ).  
        COLUMN->SET\_OPTIMIZED( ).  
        COLUMN->SET\_LONG\_TEXT( 'Current Budget' ).  
        COLUMN->SET\_MEDIUM\_TEXT( 'Currnt Budgt' ).  
        COLUMN->SET\_SHORT\_TEXT( 'Curr. Budg' ).  
  
        COLUMN = COLUMNS->GET\_COLUMN( COLUMNNAME = 'END\_COST' ).  
        COLUMN->SET\_OPTIMIZED( ).  
        COLUMN->SET\_LONG\_TEXT( 'Project End Cost' ).  
        COLUMN->SET\_MEDIUM\_TEXT( 'Proj End Cost' ).  
        COLUMN->SET\_SHORT\_TEXT( 'Proj End' ).  
  
        COLUMN = COLUMNS->GET\_COLUMN( COLUMNNAME = 'ORDR\_COST' ).  
        COLUMN->SET\_OPTIMIZED( ).  
        COLUMN->SET\_LONG\_TEXT( 'Ordered Cost' ).  
        COLUMN->SET\_MEDIUM\_TEXT( 'Ordered Cost' ).  
        COLUMN->SET\_SHORT\_TEXT( 'Ordr Cost' ).  
  
        COLUMN = COLUMNS->GET\_COLUMN( COLUMNNAME = 'ACTL\_COST ' ).  
        COLUMN->SET\_OPTIMIZED( ).  
        COLUMN->SET\_LONG\_TEXT( 'Actual Cost' ).  
        COLUMN->SET\_MEDIUM\_TEXT( 'Actual Cost' ).  
        COLUMN->SET\_SHORT\_TEXT( 'Act. Cost' ).  
  
        COLUMN = COLUMNS->GET\_COLUMN( COLUMNNAME = 'ORDR\_PERC' ).  
        COLUMN->SET\_ALIGNMENT( CL\_SALV\_COLUMN=>RIGHT ).  
        COLUMN->SET\_OPTIMIZED( ).  
        COLUMN->SET\_LONG\_TEXT( 'Ordered cx\_salv\_not\_found' ).  
        COLUMN->SET\_SHORT\_TEXT( 'Ordered %' ).  
  
        COLUMN = COLUMNS->GET\_COLUMN( COLUMNNAME = 'CURR\_ACTL' ).  
        COLUMN->SET\_OPTIMIZED( ).  
        COLUMN->SET\_LONG\_TEXT( 'Current Budget less Cost' ).  
        COLUMN->SET\_MEDIUM\_TEXT( 'Budget less Cost' ).  
        COLUMN->SET\_SHORT\_TEXT( 'Budg. Cost' ).  
  
        COLUMN = COLUMNS->GET\_COLUMN( COLUMNNAME = 'TASK\_STAT' ).  
        COLUMN->SET\_OPTIMIZED( ).  
        COLUMN->SET\_OUTPUT\_LENGTH( '6' ).  
        COLUMN->SET\_LONG\_TEXT( 'Status' ).  
        COLUMN->SET\_MEDIUM\_TEXT( 'Status' ).  
        COLUMN->SET\_SHORT\_TEXT( 'Status' ).  
  
        COLUMN = COLUMNS->GET\_COLUMN( COLUMNNAME = 'PROJ\_TYPE' ).  
        COLUMN->SET\_VISIBLE( ' ' ).  
        COLUMN = COLUMNS->GET\_COLUMN( COLUMNNAME = 'COMP\_NAME' ).  
        COLUMN->SET\_VISIBLE( ' ' ).  
        COLUMN = COLUMNS->GET\_COLUMN( COLUMNNAME = 'STATUS' ).  
        COLUMN->SET\_VISIBLE( ' ' ).  
        COLUMN = COLUMNS->GET\_COLUMN( COLUMNNAME = 'PROJ\_STAT' ).  
        COLUMN->SET\_VISIBLE( ' ' ).  
        COLUMN = COLUMNS->GET\_COLUMN( COLUMNNAME = 'TASK\_DATE' ).  
        COLUMN->SET\_VISIBLE( ' ' ).  
        COLUMN = COLUMNS->GET\_COLUMN( COLUMNNAME = 'VBUKR' ).  
        COLUMN->SET\_VISIBLE( ' ' ).  
        COLUMN = COLUMNS->GET\_COLUMN( COLUMNNAME = 'VKOKR' ).  
        COLUMN->SET\_VISIBLE( ' ' ).  
        COLUMN = COLUMNS->GET\_COLUMN( COLUMNNAME = 'PSPNR' ).  
        COLUMN->SET\_VISIBLE( ' ' ).  
        COLUMN = COLUMNS->GET\_COLUMN( COLUMNNAME = 'PRPS\_POSKI' ).  
        COLUMN->SET\_VISIBLE( ' ' ).  
        COLUMN = COLUMNS->GET\_COLUMN( COLUMNNAME = 'PRCTR ' ).  
        COLUMN->SET\_VISIBLE( ' ' ).  
        COLUMN = COLUMNS->GET\_COLUMN( COLUMNNAME = 'PSPHI' ).  
        COLUMN->SET\_VISIBLE( ' ' ).  
        COLUMN = COLUMNS->GET\_COLUMN( COLUMNNAME = 'OBJNR' ).  
        COLUMN->SET\_VISIBLE( ' ' ).  
        COLUMN = COLUMNS->GET\_COLUMN( COLUMNNAME = 'KUNNR' ).  
        COLUMN->SET\_VISIBLE( ' ' ).  
        COLUMN = COLUMNS->GET\_COLUMN( COLUMNNAME = 'PARVW' ).  
        COLUMN->SET\_VISIBLE( ' ' ).  
      CATCH CX\_SALV\_NOT\_FOUND.  
  
    ENDTRY.  
  ENDMETHOD.  
  METHOD GET\_FUNCTIONS.  
    DATA: R\_FUNCTIONS TYPE REF TO CL\_SALV\_FUNCTIONS\_LIST.  
  
    TRY .  
        R\_FUNCTIONS = R\_GRID->GET\_FUNCTIONS( ).  
      CATCH CX\_SALV\_MSG.  
  
    ENDTRY.  
    R\_FUNCTIONS->SET\_ALL( 'X' ).  
  ENDMETHOD.  
  METHOD GET\_PARAMS.  
    CALL SELECTION-SCREEN 100.  
    IF SY-SUBRC = 0.  
      VBUKR  = P\_VBUKR.  
      VKOKR  = P\_VKOKR.  
      STAT   = P\_STAT.  
      EXPATH = P\_EXPATH.  
      PSPID[]  = S\_PSPID[].  
      POSID[]  = S\_POSID[].  
      POSKI[]  = S\_POSKI[].  
      PRCTR[]  = S\_PRCTR[].  
      PRART[]  = S\_PRART[].  
    ENDIF.  
  ENDMETHOD.  
  METHOD SAVE\_FILE.  
    DATA:  
      LV\_ORIG\_COST     TYPE STRING,  
      LV\_CURR\_BUDG     TYPE STRING,  
      LV\_ORDR\_COST     TYPE STRING,  
      LV\_ACTL\_COST     TYPE STRING,  
      LV\_END\_COST      TYPE STRING,  
      LV\_FNAME TYPE STRING.  
  
    FIELD-SYMBOLS:  
      <FS\_RESULT> TYPE SOLISTI1,  
      <FS\_OUTPUT> TYPE \_OUTPUT.  
    CREATE\_CSV\_HDR( ).  
  
  
    LOOP AT T\_OUTPUT ASSIGNING <FS\_OUTPUT>.  
      LV\_ORIG\_COST = <FS\_OUTPUT>-ORIG\_COST.  
      LV\_CURR\_BUDG = <FS\_OUTPUT>-CURR\_BUDG.  
  
      LV\_ORDR\_COST = <FS\_OUTPUT>-ORDR\_COST \* 100.  
      CONCATENATE LV\_ORDR\_COST C\_% INTO LV\_ORDR\_COST.  
      LV\_ACTL\_COST = <FS\_OUTPUT>-ACTL\_COST.  
      LV\_END\_COST  = <FS\_OUTPUT>-END\_COST.  
  
      APPEND INITIAL LINE TO I\_RESULT ASSIGNING <FS\_RESULT>.  
      CONCATENATE <FS\_OUTPUT>-PROJ\_PSPID  
                  <FS\_OUTPUT>-PROJ\_POST1  
                  <FS\_OUTPUT>-POSID  
                  <FS\_OUTPUT>-PRPS\_POST1  
                  <FS\_OUTPUT>-PARVW  
                  <FS\_OUTPUT>-KUNNR  
                  <FS\_OUTPUT>-PRPS\_USR00  
                  LV\_ORIG\_COST  
                  LV\_CURR\_BUDG  
                  LV\_ORDR\_COST  
                  LV\_ACTL\_COST  
                  LV\_END\_COST  
                  <FS\_OUTPUT>-PRCTR  
                  SY-DATUM  
                  <FS\_OUTPUT>-PROJ\_STAT  
                  <FS\_OUTPUT>-TASK\_STAT  
                  <FS\_OUTPUT>-TASK\_DATE INTO <FS\_RESULT>-LINE SEPARATED BY C\_SEP.  
  
      CLEAR:  
        LV\_ORIG\_COST,  
        LV\_CURR\_BUDG,  
        LV\_ORDR\_COST,  
        LV\_ACTL\_COST,  
        LV\_END\_COST.  
    ENDLOOP.  
  
    CLEAR LV\_FNAME.  
    LV\_FNAME = P\_EXPATH.  
    IF LV\_FNAME IS NOT INITIAL.  
\*     To Download the Internal Table Data to Specified Location  
      CALL METHOD CL\_GUI\_FRONTEND\_SERVICES=>GUI\_DOWNLOAD  
        EXPORTING  
          FILENAME                  = LV\_FNAME  
          FILETYPE                  = 'ASC'  
\*         APPEND                    = 'X'  
          WRITE\_FIELD\_SEPARATOR     = 'X'  
\*         HEADER                    = '00'  
\*         TRUNC\_TRAILING\_BLANKS     = SPACE  
\*         WRITE\_LF                  = 'X'  
\*         COL\_SELECT                = SPACE  
\*         COL\_SELECT\_MASK           = SPACE  
\*         DAT\_MODE                  = SPACE  
          CONFIRM\_OVERWRITE         = 'X'   "SPACE  
\*         NO\_AUTH\_CHECK             = SPACE  
\*         CODEPAGE                  = SPACE  
\*         IGNORE\_CERR               = ABAP\_TRUE  
\*         REPLACEMENT               = '#'  
\*         WRITE\_BOM                 = SPACE  
          TRUNC\_TRAILING\_BLANKS\_EOL = ''  
\*         WK1\_N\_FORMAT              = SPACE  
\*         WK1\_N\_SIZE                = SPACE  
\*         WK1\_T\_FORMAT              = SPACE  
\*         WK1\_T\_SIZE                = SPACE  
\*        IMPORTING  
\*         FILELENGTH                =  
        CHANGING  
          DATA\_TAB                  = I\_RESULT  
        EXCEPTIONS  
          FILE\_WRITE\_ERROR          = 1  
          NO\_BATCH                  = 2  
          GUI\_REFUSE\_FILETRANSFER   = 3  
          INVALID\_TYPE              = 4  
          NO\_AUTHORITY              = 5  
          UNKNOWN\_ERROR             = 6  
          HEADER\_NOT\_ALLOWED        = 7  
          SEPARATOR\_NOT\_ALLOWED     = 8  
          FILESIZE\_NOT\_ALLOWED      = 9  
          HEADER\_TOO\_LONG           = 10  
          DP\_ERROR\_CREATE           = 11  
          DP\_ERROR\_SEND             = 12  
          DP\_ERROR\_WRITE            = 13  
          UNKNOWN\_DP\_ERROR          = 14  
          ACCESS\_DENIED             = 15  
          DP\_OUT\_OF\_MEMORY          = 16  
          DISK\_FULL                 = 17  
          DP\_TIMEOUT                = 18  
          FILE\_NOT\_FOUND            = 19  
          DATAPROVIDER\_EXCEPTION    = 20  
          CONTROL\_FLUSH\_ERROR       = 21  
          NOT\_SUPPORTED\_BY\_GUI      = 22  
          ERROR\_NO\_GUI              = 23  
          OTHERS                    = 24.  
      IF SY-SUBRC <> 0.  
        LCL\_LOG\_UTILITY=>POPULATE\_LOG( EXPORTING  
                                         I\_TYPE   = C\_I  
                                         I\_CL     = C\_MESS\_CL  
                                         I\_NUMBER = 003  
                                         I\_PAR1   = SY-MSGV1  
                                         I\_PAR2   = SY-MSGV2  
                                         I\_PAR3   = SY-MSGV3  
                                         I\_PAR4   = SY-MSGV4  
                                       CHANGING  
                                         C\_LOG    = T\_LOG ).  
        MESSAGE  S003(ZDW\_DEV\_OU) WITH 'File Successfully'(010) 'Downloaded to :'(011) LV\_FNAME .  
      ENDIF.  
    ELSE.  
      MESSAGE  S001(ZDW\_DEV\_OU) WITH 'File not found'(012).  
    ENDIF.  
  ENDMETHOD.  
  METHOD WRITE\_LOG.  
  
    CONSTANTS:  
      C\_NO\_DATA(29)    VALUE 'No valid data records to load'.  
  
    DATA:LV\_RECORDS TYPE I,  
         LS\_RETURN  TYPE BAPIRET2.  
  
    IF T\_LOG IS INITIAL AND T\_OUTPUT[] IS INITIAL.  
      LS\_RETURN-TYPE = C\_E.  
      LS\_RETURN-MESSAGE = C\_NO\_DATA.  
      APPEND LS\_RETURN TO T\_LOG.  
    ENDIF.  
    LV\_RECORDS = LINES(  T\_OUTPUT[] ).  
    LCL\_LOG\_UTILITY=>WRITE\_LOG( EXPORTING  
                                  I\_PROG\_LOG  = T\_LOG  
                                  I\_RECS      = LV\_RECORDS  
                                  I\_FILENAME1 = EXPATH ).  
  
  
  ENDMETHOD.  
ENDCLASS.  
CLASS CL\_DBOBJECT\_IO DEFINITION.  
  PUBLIC SECTION.  
    METHODS CONSTRUCTOR.  
    DATA:  
          T\_PROJ  TYPE TABLE OF \_PROJ,  
          T\_PRPS  TYPE TABLE OF \_PRPS,  
          T\_PSTAT TYPE TABLE OF JEST,  
          T\_TSTAT TYPE TABLE OF JEST,  
          T\_TPART TYPE TABLE OF TPART,  
          T\_KNA1  TYPE TABLE OF KNA1,  
          T\_VBAP  TYPE TABLE OF VBAP,  
          T\_VBPA  TYPE TABLE OF VBPA,  
          T\_RPSCO TYPE TABLE OF \_RPSCO,  
          T\_TJ02T TYPE TABLE OF TJ02T,  
          T\_COEP  TYPE TABLE OF COEP,  
          T\_BPGE  TYPE TABLE OF BPGE,  
          T\_BPHI  TYPE TABLE OF BPHI,  
          T\_TCJ1T TYPE TABLE OF TCJ1T,  
          T\_COOI  TYPE TABLE OF COOI.  
    DATA:  
          VBUKR TYPE PROJ-VBUKR,  
          VKOKR TYPE PROJ-VKOKR,  
          STAT  TYPE JEST-STAT,  
          PSPID TYPE RSELOPTION,  
          POSID TYPE RSELOPTION,  
          POSKI TYPE RSELOPTION,  
          PRCTR TYPE RSELOPTION,  
          PRART TYPE RSELOPTION,  
          COMP\_NAME TYPE T880-NAME1,  
          EXPATH TYPE RLGRAP-FILENAME.  
ENDCLASS.  
CLASS CL\_DBOBJECT\_IO IMPLEMENTATION.  
  METHOD CONSTRUCTOR.  
    CLEAR:T\_PROJ[],  
          T\_PRPS[],  
          T\_PSTAT[],  
          T\_TSTAT[],  
          T\_TPART[],  
          T\_KNA1[],  
          T\_VBAP[],  
          T\_VBPA[],  
          T\_RPSCO[],  
          T\_TJ02T[],  
          T\_COEP[],  
          T\_BPGE[],  
          T\_TCJ1T[],  
          T\_COOI[],  
          VBUKR,  
          VKOKR,  
          STAT,  
          PSPID,  
          POSID,  
          POSKI,  
          PRCTR,  
          PRART.  
  ENDMETHOD.  
ENDCLASS.  
CLASS CL\_PERSISTDB DEFINITION.  
  PUBLIC SECTION.  
    METHODS:  
     FETCH\_DATA CHANGING RC\_DATA TYPE REF TO CL\_DBOBJECT\_IO.  
ENDCLASS.  
  
CLASS CL\_PERSISTDB IMPLEMENTATION.  
  METHOD FETCH\_DATA.  
  
    DATA:  
      LV\_COMPCODE    TYPE RCOMP\_D.  
  
    UNPACK RC\_DATA->VBUKR TO LV\_COMPCODE.  
    SELECT SINGLE NAME1 INTO RC\_DATA->COMP\_NAME FROM T880  
      WHERE RCOMP = LV\_COMPCODE.  
  
    IF RC\_DATA->PSPID[] IS NOT INITIAL.  
      SELECT VBUKR VKOKR PSPNR PSPID POST1 PLSEZ OBJNR FROM PROJ  
        INTO CORRESPONDING FIELDS OF TABLE RC\_DATA->T\_PROJ  
        WHERE PSPID IN RC\_DATA->PSPID AND  
              VBUKR  = RC\_DATA->VBUKR AND  
              VKOKR  = RC\_DATA->VKOKR.  
  
      SELECT POSID POSKI PRCTR PRART POST1 PSPHI OBJNR USR00 PSPNR ERDAT STUFE FROM PRPS  
        INTO CORRESPONDING FIELDS OF TABLE RC\_DATA->T\_PRPS  
        FOR ALL ENTRIES IN RC\_DATA->T\_PROJ  
        WHERE PSPHI  = RC\_DATA->T\_PROJ-PSPNR AND  
              PRCTR IN RC\_DATA->PRCTR      AND  
              PRART IN RC\_DATA->PRART      AND  
              POSKI IN RC\_DATA->POSKI.  
    ELSE.  
      IF RC\_DATA->POSID[] IS NOT INITIAL.  
        SELECT POSID POSKI PRCTR PRART POST1 PSPHI OBJNR USR00 PSPNR ERDAT STUFE  
          FROM PRPS  
          INTO CORRESPONDING FIELDS OF TABLE RC\_DATA->T\_PRPS  
          WHERE POSID IN RC\_DATA->POSID  AND  
                PRCTR IN RC\_DATA->PRCTR  AND  
                PRART IN RC\_DATA->PRART  AND  
                POSKI IN RC\_DATA->POSKI.  
  
        SELECT VBUKR VKOKR PSPNR PSPID POST1 PLSEZ OBJNR  
          FROM PROJ  
          INTO CORRESPONDING FIELDS OF TABLE RC\_DATA->T\_PROJ  
          FOR ALL ENTRIES IN RC\_DATA->T\_PRPS  
          WHERE PSPNR  = RC\_DATA->T\_PRPS-PSPHI AND  
                VBUKR  = RC\_DATA->VBUKR      AND  
                VKOKR  = RC\_DATA->VKOKR.  
  
        SELECT POSID POSKI PRCTR PRART POST1 PSPHI OBJNR USR00 PSPNR ERDAT STUFE  
          FROM PRPS  
          INTO CORRESPONDING FIELDS OF TABLE RC\_DATA->T\_PRPS  
          FOR ALL ENTRIES IN RC\_DATA->T\_PROJ  
          WHERE PSPHI  = RC\_DATA->T\_PROJ-PSPNR AND  
                PRCTR IN RC\_DATA->PRCTR      AND  
                PRART IN RC\_DATA->PRART      AND  
                POSKI IN RC\_DATA->POSKI.  
      ELSE.  
        IF RC\_DATA->POSKI IS NOT INITIAL.  
          SELECT POSID POSKI PRCTR PRART POST1 PSPHI OBJNR USR00 PSPNR ERDAT STUFE  
            FROM PRPS  
            INTO CORRESPONDING FIELDS OF TABLE RC\_DATA->T\_PRPS  
            WHERE POSKI IN RC\_DATA->POSKI  AND  
                  PRCTR IN RC\_DATA->PRCTR  AND  
                  PRART IN RC\_DATA->PRART.  
  
          SELECT VBUKR VKOKR PSPNR PSPID POST1 PLSEZ OBJNR FROM PROJ  
            INTO CORRESPONDING FIELDS OF TABLE RC\_DATA->T\_PROJ  
            FOR ALL ENTRIES IN RC\_DATA->T\_PRPS  
            WHERE PSPNR  = RC\_DATA->T\_PRPS-PSPHI AND  
                  VBUKR  = RC\_DATA->VBUKR      AND  
                  VKOKR  = RC\_DATA->VKOKR.  
  
          SELECT POSID POSKI PRCTR PRART POST1 PSPHI OBJNR USR00 PSPNR ERDAT STUFE  
            FROM PRPS  
            INTO CORRESPONDING FIELDS OF TABLE RC\_DATA->T\_PRPS  
            FOR ALL ENTRIES IN RC\_DATA->T\_PROJ  
            WHERE PSPHI  = RC\_DATA->T\_PROJ-PSPNR AND  
                  PRCTR IN RC\_DATA->PRCTR      AND  
                  PRART IN RC\_DATA->PRART      AND  
                  POSKI IN RC\_DATA->POSKI.  
        ENDIF.  
      ENDIF.  
    ENDIF.  
  
    IF RC\_DATA->T\_PROJ IS NOT INITIAL.  
      SELECT \* FROM JEST  
        INTO TABLE RC\_DATA->T\_PSTAT  
        FOR ALL ENTRIES IN RC\_DATA->T\_PROJ  
        WHERE OBJNR = RC\_DATA->T\_PROJ-OBJNR  
        .  
    ENDIF.  
  
    IF RC\_DATA->T\_PRPS IS NOT INITIAL.  
      SELECT \* FROM JEST  
        INTO TABLE RC\_DATA->T\_TSTAT  
        FOR ALL ENTRIES IN RC\_DATA->T\_PRPS  
        WHERE OBJNR = RC\_DATA->T\_PRPS-OBJNR  
        .  
  
      SELECT OBJNR WRTTP GJAHR VORGA VERSN BELTP WLP00  
        FROM RPSCO  
        INTO TABLE RC\_DATA->T\_RPSCO  
        FOR ALL ENTRIES IN RC\_DATA->T\_PRPS  
        WHERE OBJNR = RC\_DATA->T\_PRPS-OBJNR.  
  
      SELECT \*  
        FROM TPART  
        INTO TABLE RC\_DATA->T\_TPART  
        ORDER BY SPRAS PARVW.  
  
      SELECT \*  
        FROM VBAP  
        INTO TABLE RC\_DATA->T\_VBAP  
        FOR ALL ENTRIES IN RC\_DATA->T\_PRPS  
        WHERE PS\_PSP\_PNR = RC\_DATA->T\_PRPS-PSPNR.  
  
      SELECT \*  
        FROM VBPA  
        INTO TABLE RC\_DATA->T\_VBPA  
        FOR ALL ENTRIES IN RC\_DATA->T\_VBAP  
        WHERE VBELN = RC\_DATA->T\_VBAP-VBELN.  
  
      SELECT \*  
        FROM KNA1  
        INTO TABLE RC\_DATA->T\_KNA1.  
  
      SELECT \* FROM TJ02T INTO TABLE RC\_DATA->T\_TJ02T.  
  
      SELECT \*  
        FROM COEP  
        INTO TABLE RC\_DATA->T\_COEP  
        FOR ALL ENTRIES IN RC\_DATA->T\_PRPS  
        WHERE OBJNR = RC\_DATA->T\_PRPS-OBJNR AND  
              KOKRS = P\_VKOKR  
        .  
  
      SELECT \*  
        FROM BPGE  
        INTO TABLE RC\_DATA->T\_BPGE  
        FOR ALL ENTRIES IN RC\_DATA->T\_PRPS  
        WHERE OBJNR = RC\_DATA->T\_PRPS-OBJNR.  
  
      SELECT \*  
        FROM COOI  
        INTO TABLE RC\_DATA->T\_COOI  
        FOR ALL ENTRIES IN RC\_DATA->T\_PRPS  
        WHERE OBJNR = RC\_DATA->T\_PRPS-OBJNR  
        .  
  
      SELECT \*  
        FROM TCJ1T  
        INTO TABLE RC\_DATA->T\_TCJ1T  
        FOR ALL ENTRIES IN RC\_DATA->T\_PRPS  
        WHERE PRART = RC\_DATA->T\_PRPS-PRART.  
  
      SELECT \* FROM BPHI  
        INTO TABLE RC\_DATA->T\_BPHI  
       FOR ALL ENTRIES IN RC\_DATA->T\_PRPS  
       WHERE OBJNR = RC\_DATA->T\_PRPS-OBJNR AND  
             WRTTP = C\_01.  
    ENDIF.  
  ENDMETHOD.  
ENDCLASS.  
  
CLASS CL\_MODEL DEFINITION.  
  PUBLIC SECTION.  
    METHODS:  
      CONSTRUCTOR,  
      ACCESS\_DATA      RETURNING VALUE(RVAL) TYPE BOOLEAN,  
      TBL\_CONT\_MSGS,  
      SORT\_TABLE,  
      BUILD\_OUTPUT,  
      CHECK\_FOR\_ERRORS IMPORTING VALUE(T\_ERRTAB) TYPE BAPIRET2\_T  
                       RETURNING VALUE(RVAL) TYPE BOOLEAN,  
      FILTER\_WBS\_ELEMS IMPORTING I\_STR1 TYPE ANY  
                                 I\_STR2 TYPE ANY  
                       RETURNING VALUE(RVAL) TYPE BOOLEAN,  
      GET\_CUSTDATA     IMPORTING I\_STR1  TYPE ANY  
                       EXPORTING E\_KUNNR TYPE NAME1\_GP  
                                 E\_PARVW TYPE VTXTK,  
      GET\_STATTXT       IMPORTING I\_TASK TYPE J\_OBJNR OPTIONAL  
                                  I\_PROJ TYPE J\_OBJNR OPTIONAL  
                        RETURNING VALUE(RVAL) TYPE J\_STEXT,  
      GET\_ACTUALS       IMPORTING I\_VAR  TYPE ANY  
                        CHANGING  I\_STR1 TYPE ANY,  
  
      GET\_ORDERED       IMPORTING I\_VAR  TYPE ANY  
                        CHANGING  I\_STR1 TYPE ANY,  
  
      GET\_VERSION       IMPORTING I\_STR1 TYPE ANY  
                        RETURNING VALUE(RVAL) TYPE I,  
      CONV\_ELEMENTS     CHANGING STR1 TYPE ANY,  
      CHECK\_VALUES     IMPORTING I\_STR TYPE ANY  
                       RETURNING VALUE(RVAL) TYPE BOOLEAN.  
    DATA:  
        R\_DB\_IO TYPE REF TO CL\_DBOBJECT\_IO,  
        R\_PERSIST\_DB TYPE REF TO CL\_PERSISTDB,  
        T\_LOG TYPE TABLE OF BAPIRET2,  
        V\_ERROR\_CHECK,  
        V\_RECORDS TYPE I,  
        T\_OUTPUT TYPE TABLE OF \_OUTPUT.  
  PRIVATE SECTION.  
    METHODS:  
      DB\_CALL CHANGING RC\_DATA TYPE REF TO CL\_DBOBJECT\_IO.  
ENDCLASS.  
CLASS CL\_MODEL IMPLEMENTATION.  
  METHOD CONSTRUCTOR.  
    CREATE OBJECT R\_DB\_IO.  
  ENDMETHOD.  
  METHOD CHECK\_FOR\_ERRORS.  
  
    FIELD-SYMBOLS:  
      <FS\_ILOG> TYPE BAPIRET2.  
  
    CLEAR RVAL.  
    LOOP AT T\_ERRTAB ASSIGNING <FS\_ILOG>.  
      IF <FS\_ILOG>-TYPE = C\_E.  
        RVAL = C\_YES.  
        RETURN.  
      ENDIF.  
    ENDLOOP.  
  
  ENDMETHOD.  
  METHOD ACCESS\_DATA.  
    DB\_CALL( CHANGING RC\_DATA = R\_DB\_IO ).  
    SORT\_TABLE(  ).  
    RVAL = CHECK\_FOR\_ERRORS( EXPORTING T\_ERRTAB = T\_LOG ).  
  ENDMETHOD.  
  METHOD CONV\_ELEMENTS.  
  
    FIELD-SYMBOLS:  
      <STR>  TYPE \_OUTPUT.  
  
    ASSIGN STR1 TO <STR>.  
    CALL FUNCTION 'CONVERSION\_EXIT\_PROJN\_OUTPUT'  
      EXPORTING  
        INPUT  = <STR>-PROJ\_PSPID  
      IMPORTING  
        OUTPUT = <STR>-PROJ\_PSPID.  
  
    CALL FUNCTION 'CONVERSION\_EXIT\_PROJN\_OUTPUT'  
      EXPORTING  
        INPUT  = <STR>-POSID  
      IMPORTING  
        OUTPUT = <STR>-POSID.  
  
  ENDMETHOD.  
  METHOD FILTER\_WBS\_ELEMS.  
  
    DATA:  
      V\_POS1  TYPE STRING,  
      V\_POS2  TYPE STRING,  
      V\_POS3  TYPE STRING,  
      V\_WBS   TYPE PS\_POSID,  
      V\_WLP00 TYPE BP\_WPL,  
      V\_CONTR TYPE I,  
      LV\_VERSION       TYPE BP\_VERSION,  
      S\_STR   TYPE \_PRPS.  
  
    FIELD-SYMBOLS:  
      <FS\_RPSCO> TYPE \_RPSCO,  
      <FS\_TASK>  LIKE LINE OF ME->R\_DB\_IO->T\_TSTAT.  
  
    S\_STR = I\_STR1.  
    CALL FUNCTION 'CONVERSION\_EXIT\_ABPSP\_OUTPUT'  
      EXPORTING  
        INPUT  = S\_STR-PSPNR  
      IMPORTING  
        OUTPUT = V\_WBS.  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
\*\*\*WBS element A.XXXXXXX.002.XXX or E.XXXXXXX.002.XXX  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
    SPLIT V\_WBS AT C\_PERIOD INTO V\_POS1 V\_POS2.  
    IF V\_POS1 = C\_E OR V\_POS1 = C\_A.  
      CLEAR:  
        V\_POS1.  
      SPLIT V\_POS2 AT C\_PERIOD INTO V\_POS1 V\_POS3.  
      IF V\_POS3(3) = C\_002.  
  
\*        V\_CONTR = V\_CONTR + 1.  
\*\*\*Check Status  
  
        READ TABLE ME->R\_DB\_IO->T\_TSTAT ASSIGNING <FS\_TASK> WITH KEY OBJNR = S\_STR-OBJNR  
                                                                     BINARY SEARCH.  
        IF SY-SUBRC = 0.  
          IF ( <FS\_TASK>-STAT = C\_I0045 AND <FS\_TASK>-INACT = C\_X ) OR  
             ( <FS\_TASK>-STAT = C\_I0046 AND <FS\_TASK>-INACT = C\_X ).  
            RVAL = C\_NO.  
          ELSE.  
            RVAL = C\_YES.  
          ENDIF.  
        ELSE.  
          RVAL = C\_NO.  
        ENDIF.  
\*        IF V\_CONTR IS INITIAL.  
\*          RVAL = C\_NO.  
\*        ENDIF.  
      ELSE.  
        RVAL = C\_NO.  
      ENDIF.  
    ELSE.  
      RVAL = C\_NO.  
    ENDIF.  
  
  ENDMETHOD.  
  METHOD GET\_CUSTDATA.  
  
    DATA:  
      S\_STR   TYPE \_PRPS.  
    FIELD-SYMBOLS:  
      <TPART> TYPE TPART,  
      <KNA1>  TYPE KNA1,  
      <VBAP>  TYPE VBAP,  
      <VBPA>  TYPE VBPA.  
  
    S\_STR = I\_STR1.  
    READ TABLE ME->R\_DB\_IO->T\_VBAP  
      ASSIGNING <VBAP>  
      WITH KEY PS\_PSP\_PNR = S\_STR-PSPNR  
      BINARY SEARCH.  
    IF SY-SUBRC = 0.  
      READ TABLE ME->R\_DB\_IO->T\_VBPA  
        ASSIGNING <VBPA>  
        WITH KEY VBELN = <VBAP>-VBELN  
        BINARY SEARCH.  
      IF SY-SUBRC = 0.  
        READ TABLE ME->R\_DB\_IO->T\_TPART  
          ASSIGNING <TPART> WITH KEY SPRAS = SY-LANGU  
                                     PARVW = <VBPA>-PARVW  
          BINARY SEARCH.  
        IF SY-SUBRC = 0.  
          E\_PARVW = <TPART>-VTEXT.  
        ENDIF.  
        READ TABLE ME->R\_DB\_IO->T\_KNA1  
          ASSIGNING <KNA1> WITH KEY KUNNR = <VBPA>-KUNNR  
          BINARY SEARCH.  
        IF SY-SUBRC = 0.  
          E\_KUNNR = <KNA1>-NAME1.  
        ENDIF.  
      ENDIF.  
    ENDIF.  
  ENDMETHOD.  
  METHOD GET\_ACTUALS.  
  
    DATA:  
      LV\_OBJNR TYPE J\_OBJNR.  
  
    FIELD-SYMBOLS:  
      <S\_STR>       TYPE \_OUTPUT,  
      <COEP>        TYPE COEP.  
  
    LV\_OBJNR = I\_VAR.  
    ASSIGN I\_STR1 TO <S\_STR>.  
    LOOP AT ME->R\_DB\_IO->T\_COEP  
         ASSIGNING <COEP>  
         WHERE OBJNR = LV\_OBJNR.  
      <S\_STR>-ACTL\_COST = <S\_STR>-ACTL\_COST + <COEP>-WTGBTR.  
    ENDLOOP.  
  
  
  ENDMETHOD.  
  METHOD GET\_ORDERED.  
    DATA:  
      LV\_OBJNR TYPE J\_OBJNR.  
  
    FIELD-SYMBOLS:  
      <S\_STR>       TYPE \_OUTPUT,  
      <BPGE>        TYPE BPGE,  
      <COOI>        TYPE COOI.  
  
    LV\_OBJNR = I\_VAR.  
    ASSIGN I\_STR1 TO <S\_STR>.  
    LOOP AT ME->R\_DB\_IO->T\_COOI ASSIGNING <COOI> WHERE OBJNR = LV\_OBJNR.  
      <S\_STR>-ORDR\_COST = <S\_STR>-ORDR\_COST + <COOI>-WHGBTR.  
    ENDLOOP.  
    <S\_STR>-ORDR\_COST = <S\_STR>-ORDR\_COST + <S\_STR>-ACTL\_COST.  
  
  ENDMETHOD.  
  METHOD GET\_STATTXT.  
    FIELD-SYMBOLS:  
      <FS\_TASK>     TYPE JEST,  
      <TJ02T>    TYPE TJ02T.  
  
    DATA:  
      LV\_ANW\_STAT\_EXISTING TYPE  XFELD,  
      LV\_E\_STSMA TYPE J\_STSMA,  
      LV\_LINE    TYPE J\_STEXT,  
      LV\_STAT  TYPE J\_STATUS,  
      LV\_STONR   TYPE J\_STONR.  
  
  
    IF I\_TASK IS NOT INITIAL.  
      READ TABLE ME->R\_DB\_IO->T\_TSTAT ASSIGNING <FS\_TASK> WITH KEY OBJNR = I\_TASK  
                                                    STAT  = C\_I0043 BINARY SEARCH.  
      IF SY-SUBRC = 0 AND <FS\_TASK>-INACT IS INITIAL.  
        LV\_STAT = C\_I0043.  
      ELSE.  
        READ TABLE ME->R\_DB\_IO->T\_TSTAT ASSIGNING <FS\_TASK> WITH KEY OBJNR = I\_TASK  
                                                    STAT  = C\_I0067 BINARY SEARCH.  
        IF SY-SUBRC = 0 AND <FS\_TASK>-INACT IS INITIAL.  
          LV\_STAT = C\_I0067.  
        ELSE.  
          CALL FUNCTION 'STATUS\_TEXT\_EDIT'  
            EXPORTING  
              OBJNR             = I\_TASK  
              SPRAS             = SY-LANGU  
            IMPORTING  
              ANW\_STAT\_EXISTING = LV\_ANW\_STAT\_EXISTING  
              E\_STSMA           = LV\_E\_STSMA  
              LINE              = LV\_LINE  
              USER\_LINE         = RVAL  
              STONR             = LV\_STONR.  
          IF RVAL IS INITIAL.  
            RVAL = LV\_LINE.  
          ENDIF.  
        ENDIF.  
      ENDIF.  
    ELSEIF I\_PROJ IS NOT INITIAL.  
      CALL FUNCTION 'STATUS\_TEXT\_EDIT'  
        EXPORTING  
          OBJNR             = I\_PROJ  
          SPRAS             = SY-LANGU  
        IMPORTING  
          ANW\_STAT\_EXISTING = LV\_ANW\_STAT\_EXISTING  
          E\_STSMA           = LV\_E\_STSMA  
          LINE              = LV\_LINE  
          USER\_LINE         = RVAL  
          STONR             = LV\_STONR.  
    ELSE.  
      READ TABLE ME->R\_DB\_IO->T\_TJ02T ASSIGNING <TJ02T> WITH KEY ISTAT = LV\_STAT  
                                                    SPRAS = SY-LANGU BINARY SEARCH.  
      IF SY-SUBRC = 0.  
        RVAL = <TJ02T>-TXT30.  
      ENDIF.  
    ENDIF.  
  ENDMETHOD.  
  METHOD GET\_VERSION.  
  
    DATA:  
      LV\_DTYP(4),  
      LS\_STR   TYPE \_PROJ.  
  
    FIELD-SYMBOLS:  
      <FS\_BPHI>  TYPE BPHI.  
  
    LS\_STR = I\_STR1.  
  
    LOOP AT  ME->R\_DB\_IO->T\_BPHI ASSIGNING <FS\_BPHI>.  
      CALL FUNCTION 'NUMERIC\_CHECK'  
        EXPORTING  
          STRING\_IN = <FS\_BPHI>-VERSN  
        IMPORTING  
          HTYPE     = LV\_DTYP.  
  
      IF LV\_DTYP = C\_NUMC.  
        RVAL = <FS\_BPHI>-VERSN.  
      ENDIF.  
      CLEAR LV\_DTYP.  
    ENDLOOP.  
  ENDMETHOD.  
  METHOD SORT\_TABLE.  
    SORT ME->R\_DB\_IO->T\_PSTAT BY OBJNR STAT.  
    SORT ME->R\_DB\_IO->T\_TSTAT BY OBJNR STAT.  
    SORT ME->R\_DB\_IO->T\_RPSCO BY OBJNR WRTTP GJAHR VORGA VERSN BELTP WLP00.  
    SORT ME->R\_DB\_IO->T\_TPART BY SPRAS PARVW.  
    SORT ME->R\_DB\_IO->T\_VBAP BY PS\_PSP\_PNR.  
    SORT ME->R\_DB\_IO->T\_VBPA BY VBELN.  
    SORT ME->R\_DB\_IO->T\_VBPA BY VBELN.  
    SORT ME->R\_DB\_IO->T\_TJ02T BY ISTAT SPRAS.  
    SORT ME->R\_DB\_IO->T\_COEP BY OBJNR.  
    SORT ME->R\_DB\_IO->T\_BPGE BY OBJNR WRTTP.  
    SORT ME->R\_DB\_IO->T\_COOI BY OBJNR.  
    SORT ME->R\_DB\_IO->T\_TCJ1T BY PRART.  
    SORT ME->R\_DB\_IO->T\_BPHI BY VERSN DESCENDING.  
  ENDMETHOD.  
  METHOD DB\_CALL.  
    CREATE OBJECT R\_PERSIST\_DB.  
    R\_PERSIST\_DB->FETCH\_DATA( CHANGING RC\_DATA = R\_DB\_IO ).  
  ENDMETHOD.  
  METHOD TBL\_CONT\_MSGS.  
    IF R\_DB\_IO->T\_PRPS IS INITIAL.  
      LCL\_LOG\_UTILITY=>POPULATE\_LOG( EXPORTING  
                                       I\_TYPE   = C\_E  
                                       I\_CL     = C\_MESS\_CL  
                                       I\_NUMBER = 013  
                                     CHANGING  
                                       C\_LOG    = T\_LOG ).  
    ENDIF.  
    IF R\_DB\_IO->T\_PROJ IS INITIAL.  
      LCL\_LOG\_UTILITY=>POPULATE\_LOG( EXPORTING  
                                       I\_TYPE   = C\_E  
                                       I\_CL     = C\_MESS\_CL  
                                       I\_NUMBER = 014  
                                     CHANGING  
                                       C\_LOG    = T\_LOG ).  
    ENDIF.  
  ENDMETHOD.  
  METHOD BUILD\_OUTPUT.  
  
    DATA:  
      LV\_KEEP\_RECORD   TYPE BOOLEAN,  
      LV\_POSID         TYPE PS\_POSID,  
      LV\_STATUS        TYPE J\_STATUS,  
      LV\_ORD\_COST      TYPE BP\_WPL,  
      LS\_LVL\_ONE       TYPE \_OUTPUT,  
      LV\_VERSION       TYPE BP\_VERSION.  
  
    FIELD-SYMBOLS:  
      <FS\_PRPS>    LIKE LINE OF R\_DB\_IO->T\_PRPS,  
      <FS\_RPSCO>     TYPE \_RPSCO,  
      <FS\_OUTPUT>  LIKE LINE OF T\_OUTPUT,  
      <FS\_TCJ1T>   LIKE LINE OF R\_DB\_IO->T\_TCJ1T,  
      <FS\_PROJ>    LIKE LINE OF R\_DB\_IO->T\_PROJ.  
  
    LOOP AT ME->R\_DB\_IO->T\_PROJ ASSIGNING <FS\_PROJ>.  
      CLEAR:  
        LV\_KEEP\_RECORD,  
        LS\_LVL\_ONE.  
  
      LOOP AT ME->R\_DB\_IO->T\_PRPS ASSIGNING <FS\_PRPS>  
           WHERE PSPHI = <FS\_PROJ>-PSPNR.  
        LV\_KEEP\_RECORD = FILTER\_WBS\_ELEMS( I\_STR1 = <FS\_PRPS>  
                                          I\_STR2 = <FS\_PROJ> ).  
        IF LV\_KEEP\_RECORD = C\_YES.  
\*\*\*Keep record...  
\*\*\*Use this as a holding place for level one for summation  
          IF <FS\_PRPS>-STUFE = C\_1.  
            LS\_LVL\_ONE-POSID = <FS\_PRPS>-POSID.  
          ENDIF.  
          APPEND INITIAL LINE TO T\_OUTPUT ASSIGNING <FS\_OUTPUT>.  
          READ TABLE ME->R\_DB\_IO->T\_TCJ1T  
            ASSIGNING <FS\_TCJ1T>  
            WITH KEY PRART = <FS\_PRPS>-PRART  
            BINARY SEARCH.  
          IF SY-SUBRC = 0.  
            <FS\_OUTPUT>-PROJ\_TYPE  = <FS\_TCJ1T>-PRATX.  
          ENDIF.  
          <FS\_OUTPUT>-COMP\_NAME  = ME->R\_DB\_IO->COMP\_NAME.  
          CLEAR LV\_STATUS.  
          LV\_STATUS = GET\_STATTXT( I\_TASK = <FS\_PRPS>-OBJNR ).  
          <FS\_OUTPUT>-TASK\_STAT = LV\_STATUS(1).  
          CLEAR LV\_STATUS.  
          LV\_STATUS = GET\_STATTXT( I\_PROJ = <FS\_PROJ>-OBJNR ).  
          <FS\_OUTPUT>-PROJ\_STAT = LV\_STATUS(1).  
          <FS\_OUTPUT>-TASK\_DATE   = <FS\_PRPS>-ERDAT.  
          <FS\_OUTPUT>-PROJ\_PSPID  = <FS\_PROJ>-PSPID.  
          <FS\_OUTPUT>-PROJ\_POST1  = <FS\_PROJ>-POST1.  
\*\*\*Replace comma with nothing. C\_NO is ''.  
          REPLACE ALL OCCURRENCES OF C\_COMMA  
            IN <FS\_OUTPUT>-PROJ\_POST1  
            WITH C\_NO.  
          <FS\_OUTPUT>-PROJ\_PLSEZ  = <FS\_PROJ>-PLSEZ.  
\*\*\*Sales Price  
          READ TABLE ME->R\_DB\_IO->T\_RPSCO  
            ASSIGNING <FS\_RPSCO>  
            WITH KEY OBJNR = <FS\_PRPS>-OBJNR  
                    WRTTP = C\_01  
                    GJAHR = C\_0000  
                    VORGA = C\_KSTR  
                    VERSN = C\_ZERO  
                    BELTP = C\_2  
            BINARY SEARCH.  
          IF SY-SUBRC = 0.  
            <FS\_OUTPUT>-SLS\_PRICE = <FS\_RPSCO>-WLP00.  
          ENDIF.  
          <FS\_OUTPUT>-PRPS\_USR00  = <FS\_PRPS>-USR00.  
          <FS\_OUTPUT>-PRPS\_POSKI  = <FS\_PRPS>-POSKI.  
          <FS\_OUTPUT>-PRPS\_POST1  = <FS\_PRPS>-POST1.  
  
\*\*\*Original Cost  
          READ TABLE ME->R\_DB\_IO->T\_RPSCO  
            ASSIGNING <FS\_RPSCO>  
            WITH KEY OBJNR = <FS\_PRPS>-OBJNR  
                    WRTTP = C\_01  
                    VORGA = C\_KSTP  
                    VERSN = C\_001  
                    BELTP = C\_1  
            BINARY SEARCH.  
          IF SY-SUBRC = 0.  
            IF <FS\_PRPS>-POSID = LS\_LVL\_ONE-POSID.  
              "THIS SHOULD BE CLEAR FOR LEVEL ONE, BUT JUST INCASE IT IS NOT  
              CLEAR <FS\_RPSCO>-WLP00.  
            ENDIF.  
            <FS\_OUTPUT>-ORIG\_COST = <FS\_RPSCO>-WLP00.  
            LS\_LVL\_ONE-ORIG\_COST  = LS\_LVL\_ONE-ORIG\_COST + <FS\_OUTPUT>-ORIG\_COST.  
          ENDIF.  
  
\*\*\*Current Budget  
          LV\_VERSION = GET\_VERSION( <FS\_PROJ> ).  
          UNPACK LV\_VERSION TO LV\_VERSION.  
          READ TABLE ME->R\_DB\_IO->T\_RPSCO  
            ASSIGNING <FS\_RPSCO>  
            WITH KEY OBJNR = <FS\_PRPS>-OBJNR  
                    WRTTP = C\_01  
                    VORGA = C\_KSTP  
                    VERSN = LV\_VERSION  
                    BELTP = C\_1  
            BINARY SEARCH.  
          IF SY-SUBRC = 0.  
            IF <FS\_PRPS>-POSID = LS\_LVL\_ONE-POSID.  
              "THIS SHOULD BE CLEAR FOR LEVEL ONE, BUT JUST INCASE IT IS NOT  
              CLEAR <FS\_RPSCO>-WLP00.  
            ENDIF.  
            <FS\_OUTPUT>-CURR\_BUDG  = <FS\_RPSCO>-WLP00.  
            LS\_LVL\_ONE-CURR\_BUDG  = LS\_LVL\_ONE-CURR\_BUDG + <FS\_OUTPUT>-CURR\_BUDG.  
          ENDIF.  
  
\*\*\*Projected End Cost  
          READ TABLE ME->R\_DB\_IO->T\_RPSCO  
            ASSIGNING <FS\_RPSCO>  
            WITH KEY OBJNR = <FS\_PRPS>-OBJNR  
                    WRTTP = C\_01  
                    VORGA = C\_KSTP  
                    VERSN = C\_ZERO  
                    BELTP = C\_1  
            BINARY SEARCH.  
          IF SY-SUBRC = 0.  
            IF <FS\_PRPS>-POSID = LS\_LVL\_ONE-POSID.  
              "THIS SHOULD BE CLEAR FOR LEVEL ONE, BUT JUST INCASE IT IS NOT  
              CLEAR <FS\_RPSCO>-WLP00.  
            ENDIF.  
            <FS\_OUTPUT>-END\_COST  = <FS\_RPSCO>-WLP00.  
            LS\_LVL\_ONE-END\_COST  = LS\_LVL\_ONE-END\_COST + <FS\_OUTPUT>-END\_COST.  
          ENDIF.  
          IF <FS\_PRPS>-STUFE = C\_1.  
            "LEVEL ONE  
          ELSE.  
            GET\_ACTUALS( EXPORTING I\_VAR  = <FS\_PRPS>-OBJNR  
                         CHANGING  I\_STR1 = <FS\_OUTPUT> ).  
            LS\_LVL\_ONE-ACTL\_COST  = LS\_LVL\_ONE-ACTL\_COST + <FS\_OUTPUT>-ACTL\_COST.  
  
            GET\_ORDERED( EXPORTING I\_VAR  = <FS\_PRPS>-OBJNR  
                         CHANGING  I\_STR1 = <FS\_OUTPUT> ).  
            LS\_LVL\_ONE-ORDR\_COST    = LS\_LVL\_ONE-ORDR\_COST + <FS\_OUTPUT>-ORDR\_COST.  
          ENDIF.  
  
\*\*\*Ordered Percent  
          CLEAR LV\_ORD\_COST.  
          IF <FS\_OUTPUT>-END\_COST IS NOT INITIAL.  
            LV\_ORD\_COST = ( <FS\_OUTPUT>-ORDR\_COST / <FS\_OUTPUT>-END\_COST ) \* 100.  
            WRITE LV\_ORD\_COST TO <FS\_OUTPUT>-ORDR\_PERC DECIMALS 2 RIGHT-JUSTIFIED.  
            CONCATENATE  <FS\_OUTPUT>-ORDR\_PERC C\_% INTO <FS\_OUTPUT>-ORDR\_PERC.  
          ELSE.  
            WRITE LV\_ORD\_COST TO <FS\_OUTPUT>-ORDR\_PERC DECIMALS 2 RIGHT-JUSTIFIED.  
            CONCATENATE  <FS\_OUTPUT>-ORDR\_PERC C\_% INTO <FS\_OUTPUT>-ORDR\_PERC.  
          ENDIF.  
  
\*\*\*Cur Budget Less Actual Cost  
          <FS\_OUTPUT>-CURR\_ACTL = <FS\_OUTPUT>-ORIG\_COST - <FS\_OUTPUT>-ORDR\_COST.  
  
          <FS\_OUTPUT>-VBUKR = P\_VBUKR.  
          <FS\_OUTPUT>-VKOKR = P\_VKOKR.  
          <FS\_OUTPUT>-PSPNR = <FS\_PROJ>-PSPNR.  
          <FS\_OUTPUT>-POSID = <FS\_PRPS>-POSID.  
          <FS\_OUTPUT>-PRCTR = <FS\_PRPS>-PRCTR.  
          <FS\_OUTPUT>-PSPHI = <FS\_PRPS>-PSPHI.  
          <FS\_OUTPUT>-OBJNR = <FS\_PROJ>-OBJNR.  
          CONV\_ELEMENTS( CHANGING STR1 = <FS\_OUTPUT> ).  
          GET\_CUSTDATA(  EXPORTING  
                               I\_STR1 = <FS\_PRPS>  
                         IMPORTING  
                               E\_PARVW = <FS\_OUTPUT>-PARVW  
                               E\_KUNNR = <FS\_OUTPUT>-KUNNR ).  
  
          CLEAR: LV\_KEEP\_RECORD.  
          IF <FS\_PRPS>-STUFE <> C\_1.  
            LV\_KEEP\_RECORD = CHECK\_VALUES( <FS\_OUTPUT> ).  
            IF LV\_KEEP\_RECORD = C\_NO.  
              LV\_POSID = <FS\_OUTPUT>-POSID.  
              UNASSIGN <FS\_OUTPUT>.  
              READ TABLE T\_OUTPUT ASSIGNING <FS\_OUTPUT>  
                WITH KEY POSID = LV\_POSID.  
              IF SY-SUBRC = 0.  
                DELETE T\_OUTPUT INDEX SY-TABIX.  
                CONTINUE.  
              ENDIF.  
            ENDIF.  
          ENDIF.  
        ELSE.  
          CONTINUE.  
        ENDIF.  
      ENDLOOP.  
  
      CALL FUNCTION 'CONVERSION\_EXIT\_PROJN\_OUTPUT'  
        EXPORTING  
          INPUT  = LS\_LVL\_ONE-POSID  
        IMPORTING  
          OUTPUT = LS\_LVL\_ONE-POSID.  
      SORT T\_OUTPUT BY POSID.  
      READ TABLE T\_OUTPUT  
        ASSIGNING <FS\_OUTPUT>  
        WITH KEY POSID = LS\_LVL\_ONE-POSID.  
      IF SY-SUBRC = 0.  
        <FS\_OUTPUT>-ORIG\_COST = LS\_LVL\_ONE-ORIG\_COST.  
        <FS\_OUTPUT>-CURR\_BUDG = LS\_LVL\_ONE-CURR\_BUDG.  
        <FS\_OUTPUT>-END\_COST  = LS\_LVL\_ONE-END\_COST.  
        <FS\_OUTPUT>-ACTL\_COST = LS\_LVL\_ONE-ACTL\_COST.  
        <FS\_OUTPUT>-ORDR\_COST = LS\_LVL\_ONE-ORDR\_COST.  
  
        CLEAR LV\_ORD\_COST.  
        IF LS\_LVL\_ONE-END\_COST IS NOT INITIAL.  
          LV\_ORD\_COST = ( LS\_LVL\_ONE-ORDR\_COST / LS\_LVL\_ONE-END\_COST ) \* 100.  
          WRITE LV\_ORD\_COST TO <FS\_OUTPUT>-ORDR\_PERC DECIMALS 2 RIGHT-JUSTIFIED.  
          CONCATENATE  <FS\_OUTPUT>-ORDR\_PERC C\_% INTO <FS\_OUTPUT>-ORDR\_PERC.  
        ELSE.  
          WRITE LV\_ORD\_COST TO <FS\_OUTPUT>-ORDR\_PERC DECIMALS 2 RIGHT-JUSTIFIED.  
          CONCATENATE  <FS\_OUTPUT>-ORDR\_PERC C\_% INTO <FS\_OUTPUT>-ORDR\_PERC.  
        ENDIF.  
\*        <FS\_OUTPUT>-ORDR\_PERC = LS\_LVL\_ONE-ORDR\_COST / LS\_LVL\_ONE-END\_COST.  
        <FS\_OUTPUT>-CURR\_ACTL = LS\_LVL\_ONE-ORIG\_COST - LS\_LVL\_ONE-ORDR\_COST.  
      ENDIF.  
    ENDLOOP.  
    DESCRIBE TABLE T\_OUTPUT LINES V\_RECORDS.  
  ENDMETHOD.  
  METHOD CHECK\_VALUES.  
    DATA:  
      S\_STR  TYPE \_OUTPUT.  
    S\_STR = I\_STR.  
    IF S\_STR-ORIG\_COST IS INITIAL AND  
       S\_STR-CURR\_BUDG IS INITIAL AND  
       S\_STR-END\_COST  IS INITIAL AND  
       S\_STR-ACTL\_COST IS INITIAL AND  
       S\_STR-ORDR\_COST IS INITIAL AND  
       S\_STR-SLS\_PRICE IS INITIAL.  
      RVAL = C\_NO.  
    ELSE.  
      RVAL = C\_YES.  
    ENDIF.  
  ENDMETHOD.  
ENDCLASS.  
CLASS CL\_MAINCLS DEFINITION.  
  PUBLIC SECTION.  
    METHODS: CONSTRUCTOR,  
                   START.  
    DATA: R\_CL\_MODEL TYPE REF TO CL\_MODEL,  
          R\_CL\_VIEW  TYPE REF TO CL\_VIEW.  
  
ENDCLASS.  
CLASS CL\_MAINCLS IMPLEMENTATION.  
  METHOD CONSTRUCTOR.  
    CREATE OBJECT: R\_CL\_MODEL,R\_CL\_VIEW.  
  ENDMETHOD.  
  METHOD START.  
    R\_CL\_VIEW->GET\_PARAMS( IMPORTING  
        VBUKR = R\_CL\_MODEL->R\_DB\_IO->VBUKR  
        VKOKR = R\_CL\_MODEL->R\_DB\_IO->VKOKR  
        STAT = R\_CL\_MODEL->R\_DB\_IO->STAT  
        EXPATH = R\_CL\_MODEL->R\_DB\_IO->EXPATH  
        PSPID = R\_CL\_MODEL->R\_DB\_IO->PSPID[]  
        POSID = R\_CL\_MODEL->R\_DB\_IO->POSID[]  
        POSKI = R\_CL\_MODEL->R\_DB\_IO->POSKI[]  
        PRCTR = R\_CL\_MODEL->R\_DB\_IO->PRCTR[]  
        PRART = R\_CL\_MODEL->R\_DB\_IO->PRART[] ).  
    CHECK R\_CL\_MODEL->ACCESS\_DATA( ) <> C\_YES.  
    R\_CL\_MODEL->BUILD\_OUTPUT( ).  
    IF SY-BATCH IS INITIAL.  
      R\_CL\_VIEW->DISPLAY\_GRID( EXPORTING T\_OP = R\_CL\_MODEL->T\_OUTPUT ).  
    ELSE.  
      R\_CL\_VIEW->SAVE\_FILE( IMPORTING T\_LOG = R\_CL\_MODEL->T\_LOG  ).  
    ENDIF.  
  ENDMETHOD.  
ENDCLASS.  
DATA LCL\_MAIN TYPE REF TO CL\_MAINCLS.  
  
INITIALIZATION.  
  CREATE OBJECT LCL\_MAIN.  
  
AT SELECTION-SCREEN.  
  IF SY-BATCH IS NOT INITIAL.  
    LCL\_MAIN->R\_CL\_VIEW->VALIDATE\_FILENAME( ).  
  ENDIF.  
  
AT SELECTION-SCREEN ON VALUE-REQUEST FOR P\_EXPATH.  
  LCL\_MAIN->R\_CL\_VIEW->F4\_HELP\_FILENAME( CHANGING EXPATH = P\_EXPATH ).  
  
START-OF-SELECTION.  
  
  LCL\_MAIN->START( ).  
  
END-OF-SELECTION.  
  LCL\_MAIN->R\_CL\_VIEW->WRITE\_LOG( EXPORTING EXPATH = P\_EXPATH ).