\*----------------------------------------------------------------------\*  
\*---------------------------------------------------------------------\*

\*&---------------------------------------------------------------------\*  
\*& Report ZAPR002\_AP\_FAMIS\_INBOUND  
\*&---------------------------------------------------------------------\*  
\*&  
\*&---------------------------------------------------------------------\*  
REPORT zapr002\_ap\_famis\_inbound.  
\*----------------------------------------------------------------------\*  
\*  
  
\*---------------------------------------------------------------------\*  
\* Types  
\*---------------------------------------------------------------------\*  
  
TYPES : BEGIN OF ty\_log\_exprev,  
          trans\_id TYPE zapt\_expend-ztrans\_id,  
\*            file\_nm  TYPE char15,  
          status   TYPE char30,  
        END OF ty\_log\_exprev.  
  
TYPES : BEGIN OF ty\_log\_chstrn,  
          batch\_type   TYPE char10,  
          batch\_dept   TYPE char16,  
          batch\_date   TYPE zapt\_cashtrn-batch\_dt,  
          batch\_number TYPE char13,  
          batch\_seq\_no TYPE char25,  
          status       TYPE char30,  
        END OF ty\_log\_chstrn.  
  
\*---------------------------------------------------------------------\*  
\* Global Data  
\*---------------------------------------------------------------------\*  
TABLES : zapt\_cashtrn, zapt\_expend, zapt\_reven, zapt\_mbp405.  
  
DATA : gv\_string TYPE string.  
  
DATA : gt\_chstrn TYPE TABLE OF zapt\_cashtrn,  
       gs\_chstrn TYPE zapt\_cashtrn.  
  
DATA : gt\_exp TYPE TABLE OF zapt\_expend,  
       gs\_exp TYPE zapt\_expend.  
  
DATA : gt\_rev TYPE TABLE OF zapt\_reven,  
       gs\_rev TYPE zapt\_reven.  
  
DATA : gt\_mbp405 TYPE TABLE OF zapt\_mbp405,  
       gs\_mbp405 TYPE zapt\_mbp405.  
  
DATA : tty\_log\_exprev TYPE TABLE OF ty\_log\_exprev,  
       tty\_log\_chs    TYPE TABLE OF ty\_log\_chstrn.  
  
DATA: attname    TYPE so\_obj\_des, "Attached File Name  
      i\_text     TYPE soli\_tab,   "Body Content  
      i\_subject  TYPE so\_obj\_des, "Subject Content  
      text1(250).                            "Body Content  
  
DATA: pdf             TYPE rcl\_bag\_tline,  
      wx\_document\_bcs TYPE REF TO cx\_document\_bcs,  
      wx\_send\_req\_bcs TYPE REF TO cx\_send\_req\_bcs,  
      wx\_address\_bcs  TYPE REF TO cx\_address\_bcs,  
      bcs\_mess        TYPE string.  
  
DATA: gv\_chs\_ar    TYPE text256,  
      gv\_exprev\_ar TYPE text256,  
      gv\_mbp\_ar    TYPE text256,  
      gv\_stamp     TYPE ertsp.  
  
\*---------------------------------------------------------------------\*  
\* Constants  
\*---------------------------------------------------------------------\*  
DATA : gc\_chsfile  TYPE string VALUE '/usr/sap/trans/cash\_transfer.csv',  
       gc\_revexp   TYPE string VALUE '/usr/sap/trans/rev\_exp.csv',  
       gc\_mbp405   TYPE string VALUE '/usr/sap/trans/mbp405\_daily.csv',  
       gc\_def\_path TYPE string VALUE '/usr/sap/interfaces/',  
       gc\_archive  TYPE string VALUE '/FAMIS/archive/',  
       gc\_match    TYPE string VALUE 'Matched Successfully',  
       gc\_ntmatch  TYPE string VALUE 'Unable To Match'.  
\*---------------------------------------------------------------------\*  
\* Selection screen  
\*---------------------------------------------------------------------\*  
SELECTION-SCREEN BEGIN OF BLOCK b1 WITH FRAME TITLE TEXT-001.  
  PARAMETERS:  
    pc\_chs  AS CHECKBOX DEFAULT 'X',  
    pc\_rvxp AS CHECKBOX DEFAULT 'X',  
    pc\_mbp  AS CHECKBOX DEFAULT 'X'.  
SELECTION-SCREEN END OF BLOCK b1.  
\*---------------------------------------------------------------------\*  
\* Start of Selection  
\*---------------------------------------------------------------------\*  
START-OF-SELECTION.  
  DATA(gv\_chs\_flg) = abap\_false.  
  DATA(gv\_exprev\_flg) = abap\_false.  
  DATA(gv\_mbp\_flg) = abap\_false.  
  
  GET TIME STAMP FIELD gv\_stamp.  
  
  IF pc\_chs EQ abap\_true.  
    PERFORM read\_chstrn.  
  ELSE.  
    MESSAGE 'Dont want to read Cash Transfer file data?' TYPE 'I'.  
    EXIT.  
  ENDIF.  
  
  IF pc\_rvxp EQ abap\_true.  
    PERFORM read\_revexp.  
  ELSE.  
    MESSAGE 'Dont want to read  file data?' TYPE 'I'.  
    EXIT.  
  ENDIF.  
  
  IF pc\_mbp EQ abap\_true.  
    PERFORM read\_mbp405.  
  ELSE.  
    MESSAGE 'Dont want to read MBP405 file data?' TYPE 'I'.  
    EXIT.  
  ENDIF.  
  
  PERFORM update\_log.  
\*&---------------------------------------------------------------------\*  
\*& Form read\_chstrn  
\*&---------------------------------------------------------------------\*  
\*& text  
\*&---------------------------------------------------------------------\*  
\*& -->  p1        text  
\*& <--  p2        text  
\*&---------------------------------------------------------------------\*  
FORM read\_chstrn .  
  DATA : ls\_chstrn TYPE zapt\_cashtrn.  
  DATA : lv\_tr\_am TYPE zapt\_cashtrn-trans\_amt.  
  OPEN DATASET gc\_chsfile FOR INPUT IN TEXT MODE ENCODING DEFAULT.  
  CLEAR gv\_string.  
  IF sy-subrc NE 0.  
    gv\_chs\_flg = abap\_true.  
    WRITE : /1 TEXT-002.  
  ELSE.  
    DO.  
      READ DATASET gc\_chsfile INTO gv\_string.  
      IF sy-subrc NE 0.  
        EXIT.  
      ELSE.  
        gs\_chstrn-mandt = sy-mandt.  
        gs\_chstrn-zrdate = sy-datum.  
        SPLIT gv\_string AT '|' INTO gs\_chstrn-batch\_ty gs\_chstrn-batch\_dept gs\_chstrn-batch\_dt gs\_chstrn-batch\_no gs\_chstrn-batch\_seq gs\_chstrn-batch\_dup\_ind gs\_chstrn-fm gs\_chstrn-proc\_date gs\_chstrn-trans\_cd  
        gs\_chstrn-fund gs\_chstrn-zfund\_desc gs\_chstrn-fund\_yr gs\_chstrn-acct gs\_chstrn-zdept gs\_chstrn-zdept\_desc gs\_chstrn-allot\_cat gs\_chstrn-srce\_obj gs\_chstrn-zmajor\_object gs\_chstrn-cost\_center gs\_chstrn-proj\_no gs\_chstrn-proj\_phase  
        gs\_chstrn-actv DATA(lv\_tr) gs\_chstrn-modifier gs\_chstrn-reverse gs\_chstrn-funds\_ctl\_ovrd gs\_chstrn-dept\_batch\_refno gs\_chstrn-cvn gs\_chstrn-gl\_acct gs\_chstrn-gl\_acct gs\_chstrn-vendor\_no gs\_chstrn-vendor\_sfx  
        gs\_chstrn-vendor\_name gs\_chstrn-zvendor\_inv\_no gs\_chstrn-invoice\_date gs\_chstrn-doc\_no gs\_chstrn-doc\_sfx gs\_chstrn-doc\_date gs\_chstrn-ref\_doc\_no gs\_chstrn-ref\_doc\_sfx gs\_chstrn-opt\_dept\_data gs\_chstrn-opt\_remitt\_data  
        gs\_chstrn-warrant\_no DATA(lv\_net\_doc) gs\_chstrn-zdiv gs\_chstrn-zdiv\_desc gs\_chstrn-mof gs\_chstrn-prog\_id gs\_chstrn-aging\_start\_date gs\_chstrn-payment\_type\_ind gs\_chstrn-function gs\_chstrn-gaap\_fund gs\_chstrn-gaap\_sfund gs\_chstrn-zfund\_detail  
        gs\_chstrn-ap\_type gs\_chstrn-ztrans\_year gs\_chstrn-trans\_ty gs\_chstrn-zfyyyymm gs\_chstrn-zf\_fy\_acct gs\_chstrn-zbatch\_type\_sort DATA(lv\_net\_tr).  
  
        IF gs\_chstrn-batch\_ty = 'B'.  
          CONTINUE.  
        ELSE.  
          gs\_chstrn-trans\_amt = lv\_tr.  
          gs\_chstrn-net\_doc\_amt = lv\_net\_doc.  
          gs\_chstrn-ztrans\_amt\_net = lv\_net\_tr.  
  
          APPEND gs\_chstrn TO gt\_chstrn.  
          CLEAR gs\_chstrn.  
        ENDIF.  
      ENDIF.  
    ENDDO.  
  ENDIF.  
  CLOSE DATASET gc\_chsfile.  
\*  cl\_demo\_output=>display\_data( gt\_chstrn ).  
  
  IF gt\_chstrn[] IS NOT INITIAL.  
    DELETE FROM zapt\_cashtrn CLIENT SPECIFIED WHERE mandt = sy-mandt.  
    INSERT zapt\_cashtrn FROM TABLE gt\_chstrn ACCEPTING DUPLICATE KEYS.  
    COMMIT WORK.  
  ENDIF.  
  IF gt\_chstrn[] IS NOT INITIAL  
  AND gv\_chs\_flg = abap\_false.  
  
    DATA(lv\_file) = 'C'.  
  
    PERFORM archive USING lv\_file.  
  
  ENDIF.  
  
  
ENDFORM.  
\*&---------------------------------------------------------------------\*  
\*& Form read\_revexp  
\*&---------------------------------------------------------------------\*  
\*& text  
\*&---------------------------------------------------------------------\*  
\*& -->  p1        text  
\*& <--  p2        text  
\*&---------------------------------------------------------------------\*  
FORM read\_revexp .  
  
  OPEN DATASET gc\_revexp FOR INPUT IN TEXT MODE ENCODING DEFAULT.  
  CLEAR gv\_string.  
  IF sy-subrc NE 0.  
    gv\_exprev\_flg = abap\_true.  
    WRITE : /1 TEXT-003.  
  ELSE.  
    DO.  
      READ DATASET gc\_revexp INTO gv\_string.  
      IF sy-subrc NE 0.  
        EXIT.  
      ELSE.  
        DATA(lv\_exprev) = gv\_string(1).  
        CASE lv\_exprev.  
          WHEN '1'.  
            gs\_rev-mandt = sy-mandt.  
            SPLIT gv\_string AT '|' INTO gs\_rev-zrevexp gs\_rev-zdept gs\_rev-zdiv gs\_rev-zpgm\_id gs\_rev-zmof gs\_rev-zappn\_type gs\_rev-ztrans\_code gs\_rev-zf\_fy\_acct gs\_rev-zobj gs\_rev-zproj\_no gs\_rev-zact gs\_rev-zamount gs\_rev-zsfx gs\_rev-zvendor\_no  
            gs\_rev-zvendor\_name gs\_rev-zinvoice\_no gs\_rev-zinv\_date gs\_rev-zaging\_date gs\_rev-zcheck\_date gs\_rev-zcheck\_no gs\_rev-zdoc gs\_rev-zcomp\_no gs\_rev-zdept\_no gs\_rev-zdoc\_no gs\_rev-zopt\_data gs\_rev-z1099\_ind gs\_rev-zfy\_fm gs\_rev-zenc\_fm  
            gs\_rev-zproc\_date gs\_rev-ztrans\_id.  
            APPEND gs\_rev TO gt\_rev.  
            CLEAR gs\_rev.  
          WHEN '2'.  
            gs\_exp-mandt = sy-mandt.  
            SPLIT gv\_string AT '|' INTO gs\_exp-zrevexp gs\_exp-zdept gs\_exp-zdiv gs\_exp-zpgm\_id gs\_exp-zmof gs\_exp-zappn\_type gs\_exp-ztrans\_code gs\_exp-zf\_fy\_acct gs\_exp-zobj gs\_exp-zproj\_no gs\_exp-zact gs\_exp-zamount gs\_exp-zsfx gs\_exp-zvendor\_no  
            gs\_exp-zvendor\_name gs\_exp-zinvoice\_no gs\_exp-zinv\_date gs\_exp-zaging\_date gs\_exp-zcheck\_date gs\_exp-zcheck\_no gs\_exp-zdoc gs\_exp-zcomp\_no gs\_exp-zdept\_no gs\_exp-zdoc\_no gs\_exp-zopt\_data gs\_exp-z1099\_ind gs\_exp-zfy\_fm gs\_exp-zenc\_fm  
            gs\_exp-zproc\_date gs\_exp-ztrans\_id.  
            APPEND gs\_exp TO gt\_exp.  
            CLEAR gs\_exp.  
          WHEN OTHERS.  
        ENDCASE.  
      ENDIF.  
    ENDDO.  
  ENDIF.  
  CLOSE DATASET gc\_revexp.  
  IF gt\_rev[] IS NOT INITIAL.  
    DELETE FROM zapt\_reven CLIENT SPECIFIED WHERE mandt = sy-mandt.  
    INSERT zapt\_reven FROM TABLE gt\_rev ACCEPTING DUPLICATE KEYS.  
    COMMIT WORK.  
  ENDIF.  
  IF gt\_exp[] IS NOT INITIAL.  
    DELETE FROM zapt\_cashtrn CLIENT SPECIFIED WHERE mandt = sy-mandt.  
    INSERT zapt\_expend FROM TABLE gt\_exp ACCEPTING DUPLICATE KEYS.  
    COMMIT WORK.  
  ENDIF.  
  
  IF gt\_rev[] IS NOT INITIAL  
    AND gt\_exp[] IS NOT INITIAL  
    AND gv\_exprev\_flg = abap\_false.  
  
    DATA(lv\_file) = 'E'.  
  
    PERFORM archive USING lv\_file.  
  
  ENDIF.  
  
ENDFORM.  
\*&---------------------------------------------------------------------\*  
\*& Form read\_mbp405  
\*&---------------------------------------------------------------------\*  
\*& text  
\*&---------------------------------------------------------------------\*  
\*& -->  p1        text  
\*& <--  p2        text  
\*&---------------------------------------------------------------------\*  
FORM read\_mbp405 .  
  
  OPEN DATASET gc\_mbp405 FOR INPUT IN TEXT MODE ENCODING DEFAULT.  
  CLEAR gv\_string.  
  IF sy-subrc NE 0.  
    gv\_mbp\_flg = abap\_true.  
    WRITE : /1 TEXT-004.  
  ELSE.  
    DO.  
      READ DATASET gc\_mbp405 INTO gv\_string.  
      IF sy-subrc NE 0.  
        EXIT.  
      ELSE.  
        SPLIT gv\_string AT '|' INTO gs\_mbp405-zbatch\_type gs\_mbp405-zbatch\_dept gs\_mbp405-zbatch\_date gs\_mbp405-zbatch\_number gs\_mbp405-zbatch\_seq\_no gs\_mbp405-zbatch\_dup\_ind gs\_mbp405-zfiscal\_month gs\_mbp405-zproc\_date\_raw gs\_mbp405-ztrans\_code  
        gs\_mbp405-zfund gs\_mbp405-zfund\_desc gs\_mbp405-zfy gs\_mbp405-zapprn\_acct gs\_mbp405-zdept gs\_mbp405-zdept\_desc gs\_mbp405-zallot\_cat gs\_mbp405-zsrce\_objt gs\_mbp405-zmajor\_object gs\_mbp405-zcost\_center gs\_mbp405-zproj\_no gs\_mbp405-zproj\_ph  
        gs\_mbp405-zactivity gs\_mbp405-ztrans\_amt gs\_mbp405-zm\_mod gs\_mbp405-zr\_reverse gs\_mbp405-zf\_funds\_ctl gs\_mbp405-zdept\_btch\_ref\_no gs\_mbp405-zcompt\_vouch\_no gs\_mbp405-zgl\_acct gs\_mbp405-zsl\_acct gs\_mbp405-zvendor\_no gs\_mbp405-zvendor\_no\_sfx  
        gs\_mbp405-zvendor\_name gs\_mbp405-zvendor\_inv\_no gs\_mbp405-zinvoice\_date gs\_mbp405-zcurr\_doc\_no gs\_mbp405-zcurr\_doc\_no\_sfx gs\_mbp405-zdoc\_date\_raw gs\_mbp405-zref\_doc\_no gs\_mbp405-zref\_doc\_no\_sfx gs\_mbp405-zopt\_dept\_data gs\_mbp405-zopt\_remitt\_data  
        gs\_mbp405-zwarrant\_no gs\_mbp405-znet\_doc\_amt gs\_mbp405-zdiv gs\_mbp405-zdiv\_desc gs\_mbp405-zmof gs\_mbp405-zprogram\_id gs\_mbp405-zasd\_raw gs\_mbp405-zpti gs\_mbp405-zfunction gs\_mbp405-zgaap\_fund gs\_mbp405-zgaap\_subfund gs\_mbp405-zfund\_detail  
        gs\_mbp405-zappn\_type gs\_mbp405-ztrans\_year gs\_mbp405-ztrans\_type gs\_mbp405-zfyyyymm gs\_mbp405-zf\_fy\_acct gs\_mbp405-zbatch\_type\_sort gs\_mbp405-zid.  
  
        APPEND gs\_mbp405 TO gt\_mbp405.  
        CLEAR gs\_mbp405.  
      ENDIF.  
    ENDDO.  
  ENDIF.  
  CLOSE DATASET gc\_mbp405.  
  DELETE gt\_mbp405 INDEX 1.  
\*  cl\_demo\_output=>display\_data( gt\_mbp405 ).  
  IF gt\_mbp405[] IS NOT INITIAL.  
    DELETE FROM zapt\_mbp405 CLIENT SPECIFIED WHERE mandt = sy-mandt.  
    INSERT zapt\_mbp405 FROM TABLE gt\_mbp405 ACCEPTING DUPLICATE KEYS.  
    COMMIT WORK.  
  ENDIF.  
  IF gt\_mbp405[] IS NOT INITIAL  
  AND gv\_mbp\_flg = abap\_false.  
  
    DATA(lv\_file) = 'M'.  
  
    PERFORM archive USING lv\_file.  
  
  ENDIF.  
  
  
ENDFORM.  
\*&---------------------------------------------------------------------\*  
\*& Form update\_log  
\*&---------------------------------------------------------------------\*  
\*& text  
\*&---------------------------------------------------------------------\*  
\*& -->  p1        text  
\*& <--  p2        text  
\*&---------------------------------------------------------------------\*  
FORM update\_log .  
  
  DATA : lt\_log\_exp TYPE TABLE OF ty\_log\_exprev,  
         ls\_log\_exp TYPE ty\_log\_exprev.  
  
  DATA : lt\_log\_rev TYPE TABLE OF ty\_log\_exprev,  
         ls\_log\_rev TYPE ty\_log\_exprev.  
  
  DATA : lt\_chstrn TYPE TABLE OF zapt\_cashtrn,  
         ls\_chstrn TYPE zapt\_cashtrn.  
  
  DATA : lt\_rev TYPE TABLE OF zapt\_reven,  
         ls\_rev TYPE zapt\_reven.  
  
  DATA : lt\_exp TYPE TABLE OF zapt\_expend,  
         ls\_exp TYPE zapt\_expend.  
  
  DATA : lt\_log\_chs TYPE TABLE OF ty\_log\_chstrn,  
         ls\_log\_chs TYPE ty\_log\_chstrn.  
  
  DATA : ls\_famis\_jv TYPE zfid\_famis\_jv.  
  
  FREE : lt\_rev[],  
         lt\_exp[],  
         lt\_chstrn,  
         lt\_log\_rev[],  
         lt\_log\_exp[],  
         lt\_log\_chs[].  
  
  SELECT  
    belnr,  
    dept\_no,  
    sfx,  
    rec\_date,  
    transcode,  
    source\_obj,  
    proj\_number,  
    proj\_phase,  
    dept\_act,  
    amount  
  FROM zglt\_state\_jv  
  INTO TABLE @DATA(lt\_jvfrm).  
  
  IF gt\_rev[] IS NOT INITIAL.  
\*    IF sy-subrc EQ 0.  
    lt\_rev[] = gt\_rev[].  
  
    LOOP AT lt\_rev INTO ls\_rev.  
      CLEAR ls\_log\_rev.  
      ls\_log\_rev-trans\_id = ls\_rev-ztrans\_id.  
      IF line\_exists( lt\_jvfrm[ dept\_no = ls\_rev-zdept\_no  
                                transcode = ls\_rev-ztrans\_code  
                                source\_obj = ls\_rev-zobj  
                                proj\_number = ls\_rev-zproj\_no  
                                dept\_act = ls\_rev-zact  
                                amount = ls\_rev-zamount ] ).  
  
        ls\_log\_rev-status = gc\_match.  
      ELSE.  
        ls\_log\_rev-status = gc\_ntmatch.  
      ENDIF.  
      APPEND ls\_log\_rev TO lt\_log\_rev.  
    ENDLOOP.  
\*    ENDIF.  
  ENDIF.  
  
  IF gt\_exp[] IS NOT INITIAL.  
    lt\_exp[] = gt\_exp[].  
    DATA(lv\_flg) = abap\_false.  
    SELECT  
      ztransid,  
      zhtranscode,  
      zhyrapprd,  
      zlobj,  
      zlproj,  
      zlph,  
      zlact  
      FROM zapt\_swv\_custom  
      INTO TABLE @DATA(lt\_swv).  
  
    LOOP AT lt\_exp INTO ls\_exp.  
      CLEAR ls\_log\_exp.  
      ls\_log\_exp-trans\_id = ls\_exp-ztrans\_id.  
      IF line\_exists( lt\_swv[ ztransid = ls\_exp-ztrans\_id  
                              zhtranscode = ls\_exp-ztrans\_code  
                              zhyrapprd = ls\_exp-zf\_fy\_acct  
                              zlobj = ls\_exp-zobj  
                              zlproj = ls\_exp-zproj\_no  
                              zlact = ls\_exp-zact ] ).  
  
        ls\_log\_exp-status = gc\_match.  
        lv\_flg = abap\_true.  
  
      ELSE.  
        ls\_log\_exp-status = gc\_ntmatch.  
      ENDIF.  
      APPEND ls\_log\_exp TO lt\_log\_exp.  
    ENDLOOP.  
  
    IF lv\_flg = abap\_false.  
      FREE lt\_log\_exp[].  
  
      LOOP AT lt\_exp INTO ls\_exp.  
        CLEAR ls\_log\_exp.  
        ls\_log\_exp-trans\_id = ls\_exp-ztrans\_id.  
        IF line\_exists( lt\_jvfrm[ dept\_no = ls\_exp-zdept\_no  
                                  transcode = ls\_exp-ztrans\_code  
                                  source\_obj = ls\_exp-zobj  
                                  proj\_number = ls\_exp-zproj\_no  
                                  dept\_act = ls\_exp-zact  
                                  amount = ls\_exp-zamount ] ).  
  
          ls\_log\_exp-status = gc\_match.  
        ELSE.  
          ls\_log\_exp-status = gc\_ntmatch.  
        ENDIF.  
        APPEND ls\_log\_exp TO lt\_log\_exp.  
      ENDLOOP.  
  
    ENDIF.  
  ENDIF.  
  
  IF gt\_chstrn[] IS NOT INITIAL.  
    lt\_chstrn[] = gt\_chstrn[].  
    LOOP AT lt\_chstrn INTO ls\_chstrn.  
      CLEAR ls\_log\_chs.  
      ls\_log\_chs-batch\_type = ls\_chstrn-batch\_ty.  
      ls\_log\_chs-batch\_dept = ls\_chstrn-batch\_dept.  
      ls\_log\_chs-batch\_date = ls\_chstrn-batch\_dt.  
      ls\_log\_chs-batch\_number = ls\_chstrn-batch\_no.  
      ls\_log\_chs-batch\_seq\_no = ls\_chstrn-batch\_seq.  
      IF line\_exists( lt\_jvfrm[ transcode = ls\_chstrn-trans\_cd  
                                source\_obj = ls\_chstrn-srce\_obj  
                                proj\_number = ls\_chstrn-proj\_no ] ).  
  
        ls\_log\_chs-status = gc\_match.  
        MOVE-CORRESPONDING ls\_chstrn TO ls\_famis\_jv.  
\*        ls\_famis\_jv-mandt = ls\_chstrn-mandt.  
        MODIFY zfid\_famis\_jv FROM ls\_famis\_jv.  
      ELSE.  
        ls\_log\_chs-status = gc\_ntmatch.  
      ENDIF.  
      APPEND ls\_log\_chs TO lt\_log\_chs.  
    ENDLOOP.  
  ENDIF.  
  
  PERFORM send\_mail USING lt\_log\_rev lt\_log\_exp lt\_log\_chs.  
  
ENDFORM.  
\*&---------------------------------------------------------------------\*  
\*& Form send\_mail  
\*&---------------------------------------------------------------------\*  
\*& text  
\*&---------------------------------------------------------------------\*  
\*&      --> LT\_LOG\_REV  
\*&      --> LT\_LOG\_EXP  
\*&      --> LT\_LOG\_CHS  
\*&---------------------------------------------------------------------\*  
FORM send\_mail   USING   ut\_log\_rev LIKE tty\_log\_exprev  
                         ut\_log\_exp LIKE tty\_log\_exprev  
                         ut\_log\_chs LIKE tty\_log\_chs.  
  
  DATA: send\_request   TYPE REF TO cl\_bcs.  
  DATA: binary\_content TYPE solix\_tab.  
  DATA: lo\_document       TYPE REF TO cl\_document\_bcs.  
  DATA: sender         TYPE REF TO cl\_sapuser\_bcs.  
  DATA: recipient      TYPE REF TO if\_recipient\_bcs.  
  DATA: bcs\_exception  TYPE REF TO cx\_bcs.  
  DATA: sent\_to\_all    TYPE os\_boolean.  
  DATA: xl\_content TYPE xstring,  
        l\_content  TYPE string.  
  DATA: ip\_subject TYPE string,  
        s          TYPE string.  
  DATA: ls\_log\_rev   TYPE ty\_log\_exprev,  
        ls\_log\_exp   TYPE ty\_log\_exprev,  
        ls\_log\_chstr TYPE ty\_log\_chstrn.  
  CLEAR i\_text.  
  CLEAR i\_subject.  
  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Create Mail Body\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
  " Subject of mail  
  CONCATENATE 'Cash transfer, Revenue, Expenditure Files Summary For ' sy-datum INTO i\_subject RESPECTING BLANKS.  
  text1 = '<html>'.  
  APPEND text1 TO i\_text.  
  
  text1 = '<body>'.  
  APPEND text1 TO i\_text.  
  
  text1 = '<p>'.  
  APPEND text1 TO i\_text.  
  text1 = 'Dear Sir/Madam,'.  
  APPEND text1 TO i\_text.  
  text1 = '</p>'.  
  APPEND text1 TO i\_text.  
  
  text1 = '<p>'.  
  APPEND text1 TO i\_text.  
  text1 = 'FAMIS Data Update Inbound Interface run is executed and here is the summary of the results.'.  
  APPEND text1 TO i\_text.  
  text1 = '</p>'.  
  APPEND text1 TO i\_text.  
  
  text1 = '<p></p>'.  
  APPEND text1 TO i\_text.  
  
  text1 = '<p><u>FILE SUMMARY:</u></p>'.  
  APPEND text1 TO i\_text.  
  text1 = '<p></p>'.  
  APPEND text1 TO i\_text.  
  
  IF gv\_chs\_flg = abap\_true.  
  
    text1 = '<p><b>'.  
    APPEND text1 TO i\_text.  
    text1 = TEXT-002.  
    APPEND text1 TO i\_text.  
    text1 = '</b></p>'.  
    APPEND text1 TO i\_text.  
  
  ELSE.  
  
    text1 = '<p><b>'.  
    APPEND text1 TO i\_text.  
    text1 = TEXT-005.  
    APPEND text1 TO i\_text.  
    text1 = '</b></p>'.  
    APPEND text1 TO i\_text.  
  
  ENDIF.  
  
  IF gv\_exprev\_flg = abap\_true.  
  
    text1 = '<p><b>'.  
    APPEND text1 TO i\_text.  
    text1 = TEXT-003.  
    APPEND text1 TO i\_text.  
    text1 = '</b></p>'.  
    APPEND text1 TO i\_text.  
  
  ELSE.  
  
    text1 = '<p><b>'.  
    APPEND text1 TO i\_text.  
    text1 = TEXT-006.  
    APPEND text1 TO i\_text.  
    text1 = '</b></p>'.  
    APPEND text1 TO i\_text.  
  
  ENDIF.  
  
  IF gv\_mbp\_flg = abap\_true.  
  
    text1 = '<p><b>'.  
    APPEND text1 TO i\_text.  
    text1 = TEXT-004.  
    APPEND text1 TO i\_text.  
    text1 = '</b></p>'.  
    APPEND text1 TO i\_text.  
  
  ELSE.  
  
    text1 = '<p><b>'.  
    APPEND text1 TO i\_text.  
    text1 = TEXT-007.  
    APPEND text1 TO i\_text.  
    text1 = '</b></p>'.  
    APPEND text1 TO i\_text.  
  
  ENDIF.  
  
  text1 = '<p><u>LOG SUMMARY:</u></p>'.  
  APPEND text1 TO i\_text.  
  text1 = '<p></p>'.  
  APPEND text1 TO i\_text.  
  
  IF ut\_log\_rev[] IS INITIAL.  
  
    text1 = '<p><b>'.  
    APPEND text1 TO i\_text.  
    text1 = TEXT-008.  
    APPEND text1 TO i\_text.  
    text1 = '</b></p>'.  
    APPEND text1 TO i\_text.  
  
  ELSE.  
  
    text1 = '<p><b>'.  
    APPEND text1 TO i\_text.  
    text1 = TEXT-009.  
    APPEND text1 TO i\_text.  
    text1 = '</b></p>'.  
    APPEND text1 TO i\_text.  
  
  ENDIF.  
  
  IF ut\_log\_exp[] IS INITIAL.  
  
    text1 = '<p><b>'.  
    APPEND text1 TO i\_text.  
    text1 = TEXT-010.  
    APPEND text1 TO i\_text.  
    text1 = '</b></p>'.  
    APPEND text1 TO i\_text.  
  
  ELSE.  
    text1 = '<p><b>'.  
    APPEND text1 TO i\_text.  
    text1 = TEXT-011.  
    APPEND text1 TO i\_text.  
    text1 = '</b></p>'.  
    APPEND text1 TO i\_text.  
  
  ENDIF.  
  
  IF ut\_log\_chs[] IS INITIAL.  
  
    text1 = '<p><b>'.  
    APPEND text1 TO i\_text.  
    text1 = TEXT-012.  
    APPEND text1 TO i\_text.  
    text1 = '</b></p>'.  
    APPEND text1 TO i\_text.  
  
  ELSE.  
  
    text1 = '<p><b>'.  
    APPEND text1 TO i\_text.  
    text1 = TEXT-013.  
    APPEND text1 TO i\_text.  
    text1 = '</b></p>'.  
    APPEND text1 TO i\_text.  
  
  ENDIF.  
  
  text1 = '<p></p>'.  
  APPEND text1 TO i\_text.  
  text1 = '<p><u>'.  
  APPEND text1 TO i\_text.  
  text1 = 'Also find Log Files Attached to this mail.'.  
  APPEND text1 TO i\_text.  
  text1 = '</u></p>'.  
  APPEND text1 TO i\_text.  
  
  text1 = '<p></p>'.  
  APPEND text1 TO i\_text.  
  text1 = '<p>'.  
  APPEND text1 TO i\_text.  
  text1 = 'Thank You,'.  
  APPEND text1 TO i\_text.  
  text1 = '</p>'.  
  APPEND text1 TO i\_text.  
  
  text1 = '</body>'.  
  APPEND text1 TO i\_text.  
  text1 = '</html>'.  
  APPEND text1 TO i\_text.  
  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Create Mail Body\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Create Mail document\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
  TRY.  
      send\_request = cl\_bcs=>create\_persistent( ).  
  
      lo\_document = cl\_document\_bcs=>create\_document(  
                      i\_type    = 'HTM'  
                      i\_text    = i\_text  
                      i\_subject = i\_subject ).  
    CATCH cx\_send\_req\_bcs INTO wx\_send\_req\_bcs.  
      bcs\_mess = wx\_send\_req\_bcs->get\_text( ).  
  ENDTRY.  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Create Mail document\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
  
  "Revenue log file attachment  
  IF ut\_log\_rev[] IS NOT INITIAL.  
    CLEAR : l\_content,  
            s,  
            xl\_content,  
            binary\_content.  
  
    LOOP AT ut\_log\_rev INTO ls\_log\_rev.  
      AT FIRST.  
        CONCATENATE 'Transaction ID' 'Status' cl\_abap\_char\_utilities=>cr\_lf INTO s SEPARATED BY cl\_abap\_char\_utilities=>horizontal\_tab.  
        CONCATENATE l\_content s INTO l\_content.  
      ENDAT.  
      CONCATENATE ls\_log\_rev-trans\_id ls\_log\_rev-status cl\_abap\_char\_utilities=>cr\_lf INTO s SEPARATED BY cl\_abap\_char\_utilities=>horizontal\_tab.  
      CONCATENATE l\_content s INTO l\_content.  
      CLEAR ls\_log\_rev.  
    ENDLOOP.  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*generate excel FILE attachment\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* xls  
    CALL FUNCTION 'HR\_KR\_STRING\_TO\_XSTRING'  
      EXPORTING  
        codepage\_to      = '1100'  
        unicode\_string   = l\_content  
        out\_len          = 0  
      IMPORTING  
        xstring\_stream   = xl\_content  
      EXCEPTIONS  
        invalid\_codepage = 1  
        invalid\_string   = 2  
        OTHERS           = 3.  
  
    IF sy-subrc NE 0.  
      CALL FUNCTION 'SCMS\_STRING\_TO\_XSTRING'  
        EXPORTING  
          text   = l\_content  
        IMPORTING  
          buffer = xl\_content  
        EXCEPTIONS  
          failed = 1  
          OTHERS = 2.  
    ENDIF.  
    CALL FUNCTION 'SCMS\_XSTRING\_TO\_BINARY'  
      EXPORTING  
        buffer     = xl\_content  
      TABLES  
        binary\_tab = binary\_content.  
    CLEAR attname.  
    CONCATENATE 'REVENUE\_LOG' sy-datum INTO attname SEPARATED BY '\_'.  
    TRY.  
        CALL METHOD lo\_document->add\_attachment  
          EXPORTING  
            i\_attachment\_type    = 'XLS'  
            i\_attachment\_subject = attname  
            i\_att\_content\_hex    = binary\_content.  
        CALL METHOD send\_request->set\_document( lo\_document ).  
      CATCH cx\_document\_bcs INTO wx\_document\_bcs.  
        bcs\_mess = wx\_document\_bcs->get\_text( ).  
    ENDTRY.  
  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*GENERATE EXCEL FILE ATTACHMENT\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* XLS  
  ENDIF.  
  
  "Expenditure log file attachment  
  IF ut\_log\_exp[] IS NOT INITIAL.  
    CLEAR : l\_content,  
           s,  
           xl\_content.  
    LOOP AT ut\_log\_exp INTO ls\_log\_exp.  
      AT FIRST.  
        CONCATENATE 'Transaction ID' 'Status' cl\_abap\_char\_utilities=>cr\_lf INTO s SEPARATED BY cl\_abap\_char\_utilities=>horizontal\_tab.  
        CONCATENATE l\_content s INTO l\_content.  
      ENDAT.  
  
      CONCATENATE ls\_log\_exp-trans\_id ls\_log\_exp-status cl\_abap\_char\_utilities=>cr\_lf INTO s SEPARATED BY cl\_abap\_char\_utilities=>horizontal\_tab.  
      CONCATENATE l\_content s INTO l\_content.  
      CLEAR ls\_log\_exp.  
    ENDLOOP.  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*generate excel FILE attachment\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* xls  
    CALL FUNCTION 'HR\_KR\_STRING\_TO\_XSTRING'  
      EXPORTING  
        codepage\_to      = '1100'  
        unicode\_string   = l\_content  
        out\_len          = 0  
      IMPORTING  
        xstring\_stream   = xl\_content  
      EXCEPTIONS  
        invalid\_codepage = 1  
        invalid\_string   = 2  
        OTHERS           = 3.  
  
    IF sy-subrc NE 0.  
      CALL FUNCTION 'SCMS\_STRING\_TO\_XSTRING'  
        EXPORTING  
          text   = l\_content  
        IMPORTING  
          buffer = xl\_content  
        EXCEPTIONS  
          failed = 1  
          OTHERS = 2.  
    ENDIF.  
    REFRESH: binary\_content.  
    CALL FUNCTION 'SCMS\_XSTRING\_TO\_BINARY'  
      EXPORTING  
        buffer     = xl\_content  
      TABLES  
        binary\_tab = binary\_content.  
    CLEAR attname.  
    CONCATENATE 'EXPENDITURE\_LOG' sy-datum INTO attname SEPARATED BY '\_'.  
    TRY.  
        CALL METHOD lo\_document->add\_attachment  
          EXPORTING  
            i\_attachment\_type    = 'XLS'  
            i\_attachment\_subject = attname  
            i\_att\_content\_hex    = binary\_content.  
        CALL METHOD send\_request->set\_document( lo\_document ).  
      CATCH cx\_document\_bcs INTO wx\_document\_bcs.  
        bcs\_mess = wx\_document\_bcs->get\_text( ).  
    ENDTRY.  
  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*GENERATE EXCEL FILE ATTACHMENT\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* XLS  
  ENDIF.  
  
  " Cash Transfer log file attachment  
  IF ut\_log\_chs[] IS NOT INITIAL.  
    CLEAR : l\_content,  
            s,  
            xl\_content.  
    LOOP AT ut\_log\_chs INTO ls\_log\_chstr.  
      AT FIRST.  
        CONCATENATE 'Batch\_Type' 'Batch\_Department' 'Batch\_Date' 'Batch\_Number' 'Batch\_Sequence\_Number' 'Status' cl\_abap\_char\_utilities=>cr\_lf INTO s SEPARATED BY cl\_abap\_char\_utilities=>horizontal\_tab.  
        CONCATENATE l\_content s INTO l\_content.  
      ENDAT.  
  
      CONCATENATE ls\_log\_chstr-batch\_type ls\_log\_chstr-batch\_dept ls\_log\_chstr-batch\_date ls\_log\_chstr-batch\_number ls\_log\_chstr-batch\_seq\_no ls\_log\_chstr-status cl\_abap\_char\_utilities=>cr\_lf INTO s SEPARATED BY cl\_abap\_char\_utilities=>horizontal\_tab.  
      CONCATENATE l\_content s INTO l\_content.  
    ENDLOOP.  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*generate excel FILE attachment\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* xls  
    CALL FUNCTION 'HR\_KR\_STRING\_TO\_XSTRING'  
      EXPORTING  
        codepage\_to      = '1100'  
        unicode\_string   = l\_content  
        out\_len          = 0  
      IMPORTING  
        xstring\_stream   = xl\_content  
      EXCEPTIONS  
        invalid\_codepage = 1  
        invalid\_string   = 2  
        OTHERS           = 3.  
  
    IF sy-subrc NE 0.  
      CALL FUNCTION 'SCMS\_STRING\_TO\_XSTRING'  
        EXPORTING  
          text   = l\_content  
        IMPORTING  
          buffer = xl\_content  
        EXCEPTIONS  
          failed = 1  
          OTHERS = 2.  
    ENDIF.  
    REFRESH: binary\_content.  
    CALL FUNCTION 'SCMS\_XSTRING\_TO\_BINARY'  
      EXPORTING  
        buffer     = xl\_content  
      TABLES  
        binary\_tab = binary\_content.  
    CLEAR attname.  
    CONCATENATE 'CASH\_TRANSFER\_LOG' sy-datum INTO attname SEPARATED BY '\_'.  
    TRY.  
        CALL METHOD lo\_document->add\_attachment  
          EXPORTING  
            i\_attachment\_type    = 'XLS'  
            i\_attachment\_subject = attname  
            i\_att\_content\_hex    = binary\_content.  
        CALL METHOD send\_request->set\_document( lo\_document ).  
      CATCH cx\_document\_bcs INTO wx\_document\_bcs.  
        bcs\_mess = wx\_document\_bcs->get\_text( ).  
    ENDTRY.  
  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*GENERATE EXCEL FILE ATTACHMENT\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* XLS  
  ENDIF.  
  
\*\*\*\*\*\*\*\*\*\*ASSIGN SENDER MAIL ADDRESS\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
  DATA: lo\_sender TYPE REF TO if\_sender\_bcs VALUE IS INITIAL.  
  TRY.  
      lo\_sender = cl\_cam\_address\_bcs=>create\_internet\_address( i\_address\_string = 'abc@gmail.com'  
       i\_address\_name = 'TEST' ).  
  
      send\_request->set\_sender(  
      EXPORTING  
      i\_sender = lo\_sender ).  
    CATCH cx\_address\_bcs INTO wx\_address\_bcs.  
      bcs\_mess = wx\_address\_bcs->get\_text( ).  
  ENDTRY.  
  
  TRY.  
      recipient = cl\_cam\_address\_bcs=>create\_internet\_address( 'abc@gmail.com' ).  
      CALL METHOD send\_request->add\_recipient  
        EXPORTING  
          i\_recipient = recipient  
          i\_express   = 'X'.  
    CATCH cx\_address\_bcs INTO wx\_address\_bcs.  
      bcs\_mess = wx\_address\_bcs->get\_text( ).  
  ENDTRY.  
\*\*\*\*\*\*\*\*\*\*ASSIGN SENDER MAIL ADDRESS\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
  
  
\*\*\*\*\*\*\*\*\*\*Send Mail\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
  
  TRY.  
      CALL METHOD send\_request->send(  
        EXPORTING  
          i\_with\_error\_screen = 'X'  
        RECEIVING  
          result              = sent\_to\_all ).  
      COMMIT WORK.  
      IF sy-subrc = 0.  
        WRITE : 'Mail sent successfully!!'.  
      ELSE.  
      ENDIF.  
    CATCH cx\_send\_req\_bcs INTO wx\_send\_req\_bcs.  
      bcs\_mess = wx\_send\_req\_bcs->get\_text( ).  
  ENDTRY.  
\*\*\*\*\*\*\*\*\*\*Send Mail\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
  
ENDFORM.  
\*&---------------------------------------------------------------------\*  
\*& Form archive  
\*&---------------------------------------------------------------------\*  
\*& text  
\*&---------------------------------------------------------------------\*  
\*&      --> LV\_FILE  
\*&---------------------------------------------------------------------\*  
FORM archive  USING uv\_file.  
  
  DATA : lv\_target TYPE saepathful,  
         lv\_source TYPE saepathful.  
  DATA : lv\_file  TYPE string.  
  
  CLEAR: lv\_target, lv\_source, lv\_file, gv\_exprev\_ar, gv\_chs\_ar, gv\_mbp\_ar.  
  
  IF uv\_file = 'E'.  
  
    lv\_source = gc\_revexp.  
    lv\_target = gc\_def\_path && sy-mandt && gc\_archive.  
    lv\_file = gv\_stamp.  
    CONDENSE lv\_file NO-GAPS.  
    CONCATENATE 'REV\_EXP\_ARCH\_' lv\_file '.CSV' INTO lv\_file.  
    lv\_target = lv\_target && lv\_file.  
  
    gv\_exprev\_ar = lv\_target.  
  
    CALL FUNCTION 'ZFM\_ARCHIVE\_SERVER\_TO\_SERVER'  
      EXPORTING  
        sourcepath        = lv\_source  
        targetpath        = lv\_target  
        remove\_sourcefile = 'X'  
\*     IMPORTING  
\*       LENGTH            =  
      EXCEPTIONS  
        error             = 1  
        OTHERS            = 2.  
    IF sy-subrc = 0.  
      WRITE : /1 TEXT-014, 16 gv\_exprev\_ar.  
    ENDIF.  
  
  ELSEIF uv\_file = 'C'.  
  
    lv\_source = gc\_chsfile.  
    lv\_target = gc\_def\_path && sy-mandt && gc\_archive.  
    lv\_file = gv\_stamp.  
    CONDENSE lv\_file NO-GAPS.  
    CONCATENATE 'CASH\_TRANSFER\_ARCH\_' lv\_file '.CSV' INTO lv\_file.  
    lv\_target = lv\_target && lv\_file.  
  
    gv\_chs\_ar = lv\_target.  
  
    CALL FUNCTION 'ZFM\_ARCHIVE\_SERVER\_TO\_SERVER'  
      EXPORTING  
        sourcepath        = lv\_source  
        targetpath        = lv\_target  
        remove\_sourcefile = 'X'  
\*     IMPORTING  
\*       LENGTH            =  
      EXCEPTIONS  
        error             = 1  
        OTHERS            = 2.  
    IF sy-subrc = 0.  
      WRITE : /1 TEXT-014, 16 gv\_chs\_ar.  
    ENDIF.  
  
  ELSEIF uv\_file = 'M'.  
  
    lv\_source = gc\_mbp405.  
    lv\_target = gc\_def\_path && sy-mandt && gc\_archive.  
    lv\_file = gv\_stamp.  
    CONDENSE lv\_file NO-GAPS.  
    CONCATENATE 'MBP405\_ARCH\_' lv\_file '.CSV' INTO lv\_file.  
    lv\_target = lv\_target && lv\_file.  
  
    gv\_mbp\_ar = lv\_target.  
  
    CALL FUNCTION 'ZFM\_ARCHIVE\_SERVER\_TO\_SERVER'  
      EXPORTING  
        sourcepath        = lv\_source  
        targetpath        = lv\_target  
        remove\_sourcefile = 'X'  
\*     IMPORTING  
\*       LENGTH            =  
      EXCEPTIONS  
        error             = 1  
        OTHERS            = 2.  
    IF sy-subrc = 0.  
      WRITE : /1 TEXT-014, 16 gv\_mbp\_ar.  
    ENDIF.  
  
  ENDIF.  
  
ENDFORM.