Hierarchy Comparisons

Applies to:

This sample code applies to all the versions starting from 4.6

Summary

This paper will explain about the hierarchy comparisons, this will show you the parent node to parent node comparison and Child node to child node comparison. This is specially compare at node level, if the level of the node name is wrong or the name of the same level node is not the same then it will give the error in the output as well as it will send the mail to the mail id which we provide in the selection screen.

Author(s): Sudheer Junnuthula

Company: Intelligroup Asia Pvt ltd

Created on: 27 December 2006

Author Bio

	Sudheer junnuthula is working in Intelligroup Asia Pvt ltd, I have 4.5 years of SAP Experience. I have strong knowledge in SAP Scripts, BDC's and Reports, and presently I am a Programmer analyst in Intelligroup.

Table of Contents

The Code

```
report  zeta_simple_hierarchy_comp line-size 250 no standard page heading
        message-id zccc.

type-pools: gsetc.

selection-screen skip 2.

*---------------------------------------------------------------------*
*             SELECTION-SCREEN PARAMETERS
*---------------------------------------------------------------------*

parameters: g_group1 type grpname obligatory default 'Z_GGP00000',
            g_group2 type grpname obligatory default 'S_GGP00000',
            g_setid1 type setid no-display,
            g_setid2 type setid no-display,
            g_class  type setclass
                     default gsetc_all_co_setclasses
                     no-display.
selection-screen skip.
parameters: g_mail  type so_recname obligatory.

types: begin of gty_values,
         from1  type setvalue,
         to1    type setvalue,
         setid1 type setid,
         from2  type setvalue,
         to2    type setvalue,
         setid2 type setid,
       end of gty_values.
types: begin of gty_sets,
         setid     type setid,
         shortname type setnamenew,
         descript  type settext,
       end of gty_sets.

types: gty_values_t type standard table of gty_values,
       gty_sets_t type sorted table of gty_sets
                  with unique key setid.


data: it_first_node like sethier  occurs 0 with header line,
      it_second_node like sethier  occurs 0 with header line.


data: begin of it_first_output occurs 0,
       setid(24),
       des(40) ,
       path(200) ,
      end of it_first_output.

data: begin of it_second_output occurs 0,
       setid(24),
       des(40) ,
       path(200) ,
      end of it_second_output.
data: begin of it_first_diff occurs 0,
       setid(24),
       des(40) ,
```



```
       path(200) ,
      end of it_first_diff.

data: begin of it_first_header occurs 0,
       setid(24),
       des(40) ,
       path(200) ,
      end of it_first_header.

data: begin of it_second_header occurs 0,
       setid(24),
       des(40) ,
       path(200) ,
      end of it_second_header.

data: begin of it_mail occurs 0,
       setid(24),
       to(24),
       des(75) ,
       path(200) ,
      end of it_mail.
*
data: hier1 like sethier occurs 0 with header line,
      hier2 like sethier occurs 0 with header line.
*
data: begin of value1 occurs 0,
       from(24),
       to(24) ,
      end of value1.

data: begin of value2 occurs 0,
       from(24),
       to(24) ,
      end of value2.

data: begin of diff_val1 occurs 0,
       setid(24),
       from(24),
       to(24) ,
       text type settext,
      end of diff_val1.

data: begin of diff_val2 occurs 0,
       setid(24),
       from(24),
       to(24) ,
       text type settext,
      end of diff_val2.

data: g_setid(24),
      g_store(24),
      g_path(200),
      flag.

*---------------------------------------------------------------------*
*             AT SELECTION-SCREEN: Get set ID's from external names
*---------------------------------------------------------------------*
at selection-screen.

* Get Setid from group names
  if not g_group1 is initial.
    call function 'G_SET_GET_ID_FROM_NAME'
      exporting
        shortname = g_group1
        setclass  = g_class
```



```
      old_setid = g_setid1
    importing
      new_setid = g_setid1.
  endif.

  if not g_group2 is initial.
    call function 'G_SET_GET_ID_FROM_NAME'
      exporting
        shortname = g_group2
        setclass  = g_class
        old_setid = g_setid2
    importing
      new_setid = g_setid2.
  endif.


*---------------------------------------------------------------------*
*             AT SELECTION-SCREEN: Value requests for groups
*---------------------------------------------------------------------*
at selection-screen on value-request for g_group1.
  perform value_request_group using    g_class
                               changing g_group1 g_setid1.

at selection-screen on value-request for g_group2.
  perform value_request_group using    g_class
                               changing g_group2 g_setid2.



*---------------------------------------------------------------------*
*             START-OF-SELECTION
*---------------------------------------------------------------------*
start-of-selection.

  perform main using g_setid1 g_setid2.


*---------------------------------------------------------------------*
*       FORM value_request_group                                      *
*---------------------------------------------------------------------*
*       Value request for groups                                      *
*---------------------------------------------------------------------*
*  -->  p_class        Set class                                      *
*  <->  p_group        Group name                                     *
*  <--  p_setid        Set Id of group                                *
*---------------------------------------------------------------------*
form value_request_group using    p_class type setclass
                          changing p_group type grpname
                                   p_setid type setid.

  data: l_table type tabname,
        l_field type fieldname.

* Get table and field from set class
  call function 'G_SETCLASS_GET_TABLE_FIELD'
    exporting
      setclass  = p_class
    importing
      fieldname = l_field
      tabname   = l_table.

* Call Value request function
  call function 'K_GROUP_SELECT'
    exporting
      field_name = l_field
```



```
      set        = p_group
      table      = l_table
    importing
      set_name   = p_group
      setid      = p_setid
    exceptions
      others     = 2.
  if sy-subrc <> 0.
    message id sy-msgid type 'S' number sy-msgno
            with sy-msgv1 sy-msgv2 sy-msgv3 sy-msgv4.
  endif.

endform.                    "value_request_group

*&---------------------------------------------------------------------*
*&      Form  main
*&---------------------------------------------------------------------*
*       Get the Nodes and correspondng GL Accounts
*----------------------------------------------------------------------*
*      -->P_G_SETID1  ID of 1st set                                    *
*      -->P_G_SETID2  ID of 2nd set                                    *
*----------------------------------------------------------------------*
form main  using    p_setid1
                    p_setid2.

  data: lt_found_values type gty_values_t,
        lt_found_sets   type gty_sets_t,
        lt_diff1_values type gty_values_t,
        lt_diff1_sets   type gty_sets_t,
        lt_diff2_values type gty_values_t,
        lt_diff2_sets   type gty_sets_t,
        l_err           type flag.

* Check authority to read sets
  call function 'G_SET_OR_GROUP_AUTHORITY'
    exporting
      i_setid = p_setid1
      i_actvt = '03'.
  call function 'G_SET_OR_GROUP_AUTHORITY'
    exporting
      i_setid = p_setid2
      i_actvt = '03'.


* Compare set hierarchies
  perform compare_sets using    p_setid1
                                p_setid2
                       changing l_err
                                lt_found_values
                                lt_found_sets
                                lt_diff1_values
                                lt_diff1_sets
                                lt_diff2_values
                                lt_diff2_sets.



endform.                    " main



*----------------------------------------------------------------------*
*       FORM compare_sets                                              *
*----------------------------------------------------------------------*
```



```
*       Read 2 sets from DB, check for duplicate values or sets       *
*---------------------------------------------------------------------*
*  -->  p_setid1              ID of set one                           *
*  -->  p_Setid2              ID of set two                           *
*  <--  p_err                 'X': Error occured                      *
*  <--  pt_found_values       Common value intervals                  *
*  <--  pt_found_sets         Common sets                             *
*  <--  pt_diff1_values       Values of 1st set missing in 2nd        *
*  <--  pt_diff1_sets         Sub-sets of 1st set missing in 2nd      *
*  <--  pt_diff2_values       Values of 2nd set missing in 1st        *
*  <--  pt_diff2_sets         Sub-sets of 2nd set missing in 1st      *
*---------------------------------------------------------------------*
form compare_sets using    p_setid1        type setid
                           p_setid2        type setid
                  changing p_err           type flag
                           pt_found_values type gty_values_t
                           pt_found_sets   type gty_sets_t
                           pt_diff1_values type gty_values_t
                           pt_diff1_sets   type gty_sets_t
                           pt_diff2_values type gty_values_t
                           pt_diff2_sets   type gty_sets_t.


  data: lt_hier1      type standard table of sethier,
        lt_hier2      type standard table of sethier,
        lt_val1       type standard table of setvalues,
        lt_val2       type standard table of setvalues,
        lt_hier_ptr1  type standard table of sethierptr,
        lt_hier_ptr2  type standard table of sethierptr,
        lt_val_ptr1   type standard table of setvalptr,
        lt_val_ptr2   type standard table of setvalptr,
        lt_hier_sort1 type standard table of sethiersrt,
        lt_hier_sort2 type standard table of sethiersrt,
        lt_val_sort1  type standard table of setvalsrt,
        lt_val_sort2  type standard table of setvalsrt.

  data: l_set          type sethierptr,
        l_parent       type sethierptr,
        l_value        type setvalptr,
        l_value_sort   type setvalsrt,
        l_found_value  type gty_values,
        l_found_set    type gty_sets,
        l_int          type i,
        l_check        type sethierptr.

  data: l_first        type sethierptr,
        l_second       type sethierptr,
        l_text(100),
        l_text1(100),
        l_first_tot type i,
        l_second_tot type i,
        first_node type setnamenew,
        second_node type setnamenew,
        z(4) value '''Z%''',
        s(4) value '''S%'''.



* Initialize list of found values and sets
  refresh: pt_found_values, pt_found_sets.
  clear: p_err.

* Read set hierarchies
  call function 'G_SET_TREE_IMPORT'
    exporting
```



```
      no_descriptions = ' '
      no_rw_info      = 'X'
      setid           = p_setid1
    tables
      set_hierarchy   = lt_hier1
      set_values      = lt_val1.

  call function 'G_SET_TREE_IMPORT'
    exporting
      no_descriptions = ' '
      no_rw_info      = 'X'
      setid           = p_setid2
    tables
      set_hierarchy   = lt_hier2
      set_values      = lt_val2.

hier1[] = lt_hier1.
hier2[] = lt_hier2.

delete hier1 where type <> 'B'.
delete hier2 where type <> 'B'.

sort hier1.
sort hier2.

* Add Pointers to set hierarchies
  call function 'G_SET_TREE_ADD_POINTERS'
    tables
      set_hierarchy = lt_hier1
      set_values    = lt_val1
      set_hier_ptr  = lt_hier_ptr1
      set_val_ptr   = lt_val_ptr1
      set_hier_sort = lt_hier_sort1
      set_val_sort  = lt_val_sort1.

  call function 'G_SET_TREE_ADD_POINTERS'
    tables
      set_hierarchy = lt_hier2
      set_values    = lt_val2
      set_hier_ptr  = lt_hier_ptr2
      set_val_ptr   = lt_val_ptr2
      set_hier_sort = lt_hier_sort2
      set_val_sort  = lt_val_sort2.

data:
       l_tabix type sy-tabix,
       ind     type sy-tabix.
*
* Search Nodes in 1st Hierarchy missing in 2nd Hierarchy
  loop at lt_hier1 into l_first.

    l_tabix = sy-tabix.

    read table lt_hier2 into l_second with key
               shortname+2(8) = l_first-shortname+2(8).

    if sy-subrc <> 0.
      move-corresponding l_first to it_first_node.
      append it_first_node.
      clear it_first_node.
** New change to display the Node with different parent node
    else.
       select count(*) from setnode into l_first_tot
                    where setname = l_first-shortname.
       select count(*) from setnode into l_second_tot
```



```
                where setname = l_second-shortname.
     if l_first_tot <> l_second_tot.
        move: l_first-shortname to it_first_diff-setid,
              l_first-descript  to it_first_diff-des.
      do.
        if flag is initial.
         g_store = l_first-shortname.
        else.
         g_store = g_setid.
        clear g_setid.
       endif.  " IF FLAG IS INITIAL.

     select single setname into g_setid from setnode
            where subsetname = g_store.

      if flag is initial.
        concatenate g_setid ' -> ' g_store into g_path.
      else.
       concatenate g_setid ' -> ' g_path into g_path.
      endif.  " IF FLAG IS INITIAL.

        flag = 1.
       if g_setid = 'Z_GGP00000' or g_setid = 'S_GGP00000'      .
         clear flag.
         move g_path to it_first_diff-path.
              append it_first_diff.
              clear  it_first_diff.
         exit.
       endif.  " IF G_SETID = 'Z_GGP00000' OR G_SETID = 'S_GGP00000'
enddo.


endif.   " IF L_FIRST_TOT <> L_SECOND_TOT.
 if l_first-shortname+0(1) <> l_second-shortname+0(1).
  if l_first-shortname+0(1) = 'Z'.
   select single setname from setnode into first_node
              where subsetname like 'Z%' and
                    subsetname = l_first-shortname .
   select single setname from setnode into second_node
              where subsetname like 'S%' and
                    subsetname = l_second-shortname .

  elseif l_first-shortname+0(1) = 'S'.
   select single setname from setnode into second_node
              where subsetname like 'S%' and
                    subsetname = l_first-shortname .
   select single setname from setnode into second_node
              where subsetname like 'Z%' and
                    subsetname = l_first-shortname .

  endif.   " IF L_FIRST-SHORTNAME+0(1) = 'Z'.

  if not first_node  is initial and
     not second_node is initial.

     if first_node+2(8) <> second_node+2(8).
         move: l_first-shortname to it_first_header-setid,
               l_first-descript  to it_first_header-des.
         do.
            if flag is initial.
              g_store = l_first-shortname.
            else.
              g_store = g_setid.
              clear g_setid.
           endif.  " IF FLAG IS INITIAL.
```



```
              select single setname into g_setid from setnode
                                  where subsetname = g_store.

             if flag is initial.
               concatenate g_setid ' -> ' g_store into g_path.
             else.
               concatenate g_setid ' -> ' g_path into g_path.
             endif.  " IF FLAG IS INITIAL.

              flag = 1.
            if g_setid = 'Z_GGP00000' or g_setid = 'S_GGP00000'      .
               clear flag.
               move g_path to it_first_header-path.
               append it_first_header.
               clear  it_first_header.
             exit.
            endif.
         enddo.

       endif.   " IF FIRST_NODE+2(8) <> SECOND_NODE+2(8).
     endif.   " IF NOT FIRST_NODE  IS INITIAL AND
    endif.    " IF L_FIRST-SHORTNAME+0(1) <> L_SECOND-SHORTNAME+0(1).
  endif. " IF sy-subrc <> 0.

 endloop.    " LOOP AT LT_HIER INTO L_FIRST
*
* Search Nodes in 2nd Hierarchy missing in 1st Hierarchy
  loop at lt_hier2 into l_second.
    ind = sy-tabix.

    read table lt_hier1 into l_first with key
              shortname+2(8) = l_second-shortname+2(8).
    if sy-subrc <> 0.
      move-corresponding l_second to it_second_node.
      append it_second_node.
      clear  it_second_node.
    endif.   " IF SY-SUBRC <> 0.

  endloop.  " LOOP AT LT_HIER2 INTO L_SET.

* Search for values in 1st hierarchy missing in 2nd hierarchy
  loop at lt_val_ptr1 into l_value.
    loop at lt_val2 transporting no fields
         where from <= l_value-from
           and to   >= l_value-to.
      exit.
    endloop.  " LOOP AT lt_val2 TRANSPORTING NO FIELDS
    check sy-subrc <> 0.
*   Found a missing value
    clear l_found_value.
    l_found_value-from1 = l_value-from.
    l_found_value-to1 = l_value-to.
    read table lt_hier_ptr1 into l_set index l_value-pup.
    if sy-subrc <> 0.
      message x016(gs) with 'SET_VALUES_CORRUPT' p_setid1.
    endif.
    l_found_value-setid1 = l_set-setid.
    append l_found_value to pt_diff1_values.
  endloop.   " LOOP AT lt_val_ptr1 INTO l_value.

*
*
* Search for values in 2nd hierarchy missing in 1st hierarchy
  loop at lt_val_ptr2 into l_value.
```



```
    loop at lt_val1 transporting no fields
         where from <= l_value-from
           and to   >= l_value-to.
      exit.
    endloop.   " LOOP AT lt_val1 TRANSPORTING NO FIELDS
    check sy-subrc <> 0.
*   Found a missing value
    clear l_found_value.
    l_found_value-from2 = l_value-from.
    l_found_value-to2 = l_value-to.
    read table lt_hier_ptr2 into l_set index l_value-pup.
    if sy-subrc <> 0.
      message x016(gs) with 'SET_VALUES_CORRUPT' p_setid2.
    endif.
    l_found_value-setid2 = l_set-setid.
    append l_found_value to pt_diff2_values.
  endloop.  " LOOP AT lt_val_ptr2 INTO l_value.

* Sort found values
  sort pt_found_values by from1.
  sort pt_diff1_values by from1.
  sort pt_diff2_values by from2.
*
*
*
*** Code for the Diff in GL Accounts in both hietarchies
loop at hier1.
 read table hier2 with key
              shortname+2(8) = hier1-shortname+2(8)
              type           = hier1-type .
 if sy-subrc = 0.
   select valfrom
          valto
          into table value1
          from setleaf
          where setname = hier1-shortname.
   select valfrom
          valto
          into table value2
          from setleaf
          where setname = hier2-shortname.

   loop at value1.
    read table value2 with key
               from = value1-from
               to   = value1-to.
    if sy-subrc <> 0.
       diff_val1-setid = hier1-shortname.
       diff_val1-from  = value1-from.
       diff_val1-to    = value1-to.
      append diff_val1.
      clear diff_val1.
    endif. " if sy-subrc <> 0.

   endloop.    " loop at value1.

  clear: value1,value2.
  refresh: value1,value2.

 endif.   " if sy-subrc = 0.

endloop.  " loop at hier1.

loop at hier2.
 read table hier1 with key
```



```
                shortname+2(8) = hier2-shortname+2(8)
                type           = hier2-type .
 if sy-subrc = 0.
   select valfrom
          valto

          into  table value1
          from setleaf
          where setname = hier2-shortname.
   select valfrom
          valto

          into  table value2
          from setleaf
          where setname = hier1-shortname.

   loop at value1.
    read table value2 with key
                from = value1-from
                to   = value1-to.
    if sy-subrc <> 0.
       diff_val2-setid = hier2-shortname.
       diff_val2-from  = value1-from.
       diff_val2-to    = value1-to.
      append diff_val2.
      clear diff_val2.
    endif. " if sy-subrc <> 0.

   endloop.    " loop at value2.

  clear: value1,value2.
  refresh: value1,value2.

 endif.   " if sy-subrc = 0.

endloop.  " loop at hier2.
*** End Code for the Diff in GL Accounts in both hietarchies

  l_text = text-004.
  write: 60(15) l_text color col_total.
  it_mail-des = l_text.
  append it_mail.
  clear it_mail.
  clear l_text.

  l_text = text-001.
  replace '&1' with g_group1 into l_text.
  replace '&2' with g_group2 into l_text.
  condense l_text.
  write: /41(52) l_text color col_total.
  it_mail-des = l_text.
  append it_mail.
  clear it_mail.
  write:/38 text-012 color col_total, sy-datum color col_total.
  concatenate text-012 sy-datum into l_text1.
  it_mail-des = l_text1.
  append it_mail.
  clear:it_mail,l_text1.
  write: 56 text-013 color col_total, sy-timlo color col_total.
  concatenate text-013 sy-timlo into l_text1.
  it_mail-des = l_text1.
  append it_mail.
  clear:it_mail,l_text1.
  write: 71 text-014 color col_total, sy-sysid color col_total.
  concatenate text-014 sy-sysid into l_text1.
```



```
  it_mail-des = l_text1.
  append it_mail.
  clear:it_mail,l_text1.
  write: 86 text-015 color col_total, sy-mandt color col_total.
  concatenate: text-015 sy-mandt into l_text1.
  it_mail-des = l_text1.
  append it_mail.
  clear:it_mail,l_text1.


write:/ sy-uline.
  if not it_first_node[] is initial.

   loop at it_first_node.

   move: it_first_node-shortname to it_first_output-setid,
         it_first_node-descript  to it_first_output-des.

    do.
     if flag is initial.
      g_store = it_first_node-shortname.
     else.
      g_store = g_setid.
      clear g_setid.
     endif.

     select single setname into g_setid from setnode
            where subsetname = g_store.

     if flag is initial.
      concatenate g_setid ' -> ' g_store into g_path.
     else.
      concatenate g_setid ' -> ' g_path into g_path.
     endif.

     flag = 1.
    if g_setid = 'Z_GGP00000' or g_setid = 'S_GGP00000'      .
      clear flag.
      move g_path to it_first_output-path.
      append it_first_output.
      clear it_first_output.
      exit.
    endif. " IF G_SETID = 'Z_GGP00000' OR G_SETID = 'S_GGP00000'
   enddo.
 endloop.   " LOOP AT IT_FIRST_NODE.


 endif.  "  IF NOT it_first_node[] IS INITIAL.
 skip.

  if not it_second_node[] is initial.

    clear: flag,g_store,g_setid,g_path.

   loop at it_second_node.

   move: it_second_node-shortname to it_second_output-setid,
         it_second_node-descript  to it_second_output-des.

    do.
     if flag is initial.
      g_store = it_second_node-shortname.
     else.
      g_store = g_setid.
      clear g_setid.
```



```
     endif.

     select single setname into g_setid from setnode
            where subsetname = g_store.

     if flag is initial.
      concatenate g_setid ' -> ' g_store into g_path.
     else.
      concatenate g_setid ' -> ' g_path into g_path.
     endif.

     flag = 1.
    if g_setid = 'Z_GGP00000' or g_setid = 'S_GGP00000'      .
      clear flag.
      move g_path to it_second_output-path.
      append it_second_output.
      clear it_second_output.
      exit.
    endif.
   enddo.
 endloop.   " LOOP AT IT_SECOND_NODE.

  endif.   " IF NOT it_second_node[] IS INITIAL.

*** Output
perform list_display using pt_diff1_values
                           pt_diff2_values.


endform.                    " compare_sets


*&---------------------------------------------------------------------*
*&      Form  LIST_DISPLAY
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
*  -->  p1        text
*  <--  p2        text
*----------------------------------------------------------------------*
form list_display using it_diff1_values type gty_values_t
                        it_diff2_values type gty_values_t.

data: l_text(100).
data: miss1_val type gty_values,
      miss2_val type gty_values .

** Blank line
    it_mail-setid = '        '    .
    append it_mail.
    clear it_mail.

    l_text = text-003.
    replace '&1' with g_group1 into l_text.
    replace '&2' with g_group2 into l_text.
    condense l_text.
    write:/(44) l_text color col_total.
    move: l_text to it_mail-des.
    append it_mail.
    clear it_mail.

if not it_first_output[] is initial.
  loop at it_first_output.
    write:/ it_first_output.
    move: it_first_output-setid to it_mail-setid,
```



```
          it_first_output-des to it_mail-des,
          it_first_output-path to it_mail-path.
          append it_mail.
          clear  it_mail.
  endloop.
else.
   l_text = text-020.
   write:/ l_text.
   move: l_text to it_mail-des.
   append it_mail.
   clear it_mail.

endif.   " IF NOT IT_FIRST_OUTPUT[] IS INITIAL.

** Blank line
    it_mail-setid = '        '    .
    append it_mail.
    clear it_mail.

    skip.
    l_text = text-003.
    replace '&2' with g_group1 into l_text.
    replace '&1' with g_group2 into l_text.
    condense l_text.
    write:/(44) l_text color col_total.
    move: l_text to it_mail-des.
    append it_mail.
    clear it_mail.

if not it_second_output[] is initial.
  loop at it_second_output.
    write:/ it_second_output.
    move: it_second_output-setid to it_mail-setid,
          it_second_output-des to it_mail-des,
          it_second_output-path to it_mail-path.
          append it_mail.
          clear  it_mail.
  endloop.
else.
   l_text = text-020.
   write:/ l_text.
   move: l_text to it_mail-des.
   append it_mail.
   clear it_mail.
endif.   " IF NOT IT_SECOND_OUTPUT[] IS INITIAL.


** Blank line
    it_mail-setid = '        '    .
    append it_mail.
    clear it_mail.

    skip.
    l_text = text-025.
    replace '&1' with g_group1 into l_text.
    replace '&2' with g_group2 into l_text.
    condense l_text.
    write:/(76) l_text color col_total.
    move: l_text to it_mail-des.
    append it_mail.
    clear it_mail.

if not it_first_header[] is initial and
   not it_first_diff[] is initial.
```



```
 loop at it_first_header.
    write:/ it_first_header.
    move: it_first_header-setid to it_mail-setid,
          it_first_header-des to it_mail-des,
          it_first_header-path to it_mail-path.
          append it_mail.
          clear  it_mail.
 endloop.    "  LOOP AT IT_FIRST_HEADER.

 loop at it_first_diff.
    write:/ it_first_diff.
    move: it_first_diff-setid to it_mail-setid,
          it_first_diff-des to it_mail-des,
          it_first_diff-path to it_mail-path.
          append it_mail.
          clear  it_mail.
 endloop.    "  LOOP AT IT_FIRST_DIFF.

else.    " IF NOT IT_FIRST_HEADER[] IS INITIAL AND

   l_text = text-020.
   write:/ l_text.
   move: l_text to it_mail-des.
   append it_mail.
   clear it_mail.

endif.    " IF NOT IT_FIRST_HEADER[] IS INITIAL AND

** Blank line
    it_mail-setid = '        '    .
    append it_mail.
    clear it_mail.

    skip.
    l_text = text-002.
    replace '&1' with g_group1 into l_text.
    replace '&2' with g_group2 into l_text.
    condense l_text.
    write:/(50) l_text color col_total.
    move: l_text to it_mail-des.
    append it_mail.
    clear it_mail.

data: l_con(20).

if not it_diff1_values[] is initial or
   not diff_val1[] is initial.

 loop at it_diff1_values into miss1_val.
    write:/ miss1_val-from1(11).
    move: miss1_val-from1 to it_mail-setid.
     if miss1_val-from1 <> miss1_val-to1.
      write: '-'.
      write: miss1_val-to1.
      concatenate '-'  miss1_val-to1 into l_con.
      move: l_con to it_mail-to.
      clear l_con.
     endif.  " IF MISS1_VAL-FROM1 <> MISS1_VAL-TO1.
***** Code for the Path
    do.
     if flag is initial.
      g_store = miss1_val-setid1+8(10).
     else.
      g_store = g_setid.
      clear g_setid.
```



```
     endif.   " IF FLAG IS INITIAL

      select single setname into g_setid from setnode
            where subsetname = g_store.

     if flag is initial.
      concatenate g_setid ' -> ' g_store into g_path.
     else.
      concatenate g_setid ' -> ' g_path into g_path.
     endif.   " IF FLAG IS INITIAL

     flag = 1.
    if g_setid = 'Z_GGP00000' or g_setid = 'S_GGP00000'      .
      clear flag.
      write: 30 g_path.
      move: g_path to it_mail-path.
      exit.
     endif.

     enddo.
      append it_mail.
      clear  it_mail.
    endloop.     " LOOP AT IT_DIFF1_VALUES INTO MISS1_VAL.
***** End Code for the Path

** Code for the missing GL accounts
loop at diff_val1.

  read table it_diff1_values into miss1_val
       with key from1 = diff_val1-from.
    if sy-subrc <> 0.
     write:/ diff_val1-from(11).
     move: diff_val1-from to it_mail-setid.
      if diff_val1-from <> diff_val1-to.
       write: '-'.
       write: diff_val1-to.
       concatenate '-'  miss1_val-to1 into l_con.
       move: l_con to it_mail-to.
       clear l_con.
     endif.  " IF MISS1_VAL-FROM1 <> MISS1_VAL-TO1.
    endif.   " IF SY-SUBRC <> 0.


****** Code for the Path
    do.
     if flag is initial.
      g_store = diff_val1-setid.
     else.
      g_store = g_setid.
      clear g_setid.
     endif.

      select single setname into g_setid from setnode
            where subsetname = g_store.

     if flag is initial.
      concatenate g_setid ' -> ' g_store into g_path.
     else.
      concatenate g_setid ' -> ' g_path into g_path.
     endif.

     flag = 1.
    if g_setid = 'Z_GGP00000' or g_setid = 'S_GGP00000'      .
      clear flag.
      write: 70 g_path.
```



```
      move: g_path to it_mail-path.
      exit.
     endif.

     enddo.
      append it_mail.
      clear  it_mail.
    endloop.   " LOOP AT DIFF_VAL1.
** End Code for the missing GL accounts



else.

   l_text = text-020.
   write:/ l_text.
   move: l_text to it_mail-des.
   append it_mail.
   clear it_mail.

endif.   " IF NOT IT_DIFF1_VALUES[] IS INITIAL.

** Blank line
    it_mail-setid = '        '    .
    append it_mail.
    clear it_mail.

    skip.
    l_text = text-002.
    replace '&2' with g_group1 into l_text.
    replace '&1' with g_group2 into l_text.
    condense l_text.
    write:/(50) l_text color col_total.
    move: l_text to it_mail-des.
    append it_mail.
    clear it_mail.

if not it_diff2_values[] is initial or
   not diff_val2[] is initial.

 loop at it_diff2_values into miss2_val.
    write:/ miss2_val-from2(11).
    move: miss2_val-from2 to it_mail-setid.
     if miss2_val-from2 <> miss1_val-to2.
      write: '-'.
      write: miss2_val-to2.
      concatenate '-'  miss2_val-to2 into l_con.
      move: l_con to it_mail-to.
      clear l_con.
     endif.  " IF MISS2_VAL-FROM1 <> MISS2_VAL-TO1.
***** Code for the Path
    do.
     if flag is initial.
      g_store = miss2_val-setid2+8(10).
     else.
      g_store = g_setid.
      clear g_setid.
     endif.

      select single setname into g_setid from setnode
            where subsetname = g_store.

     if flag is initial.
      concatenate g_setid ' -> ' g_store into g_path.
     else.
```



```
      concatenate g_setid ' -> ' g_path into g_path.
     endif.

     flag = 1.
    if g_setid = 'Z_GGP00000' or g_setid = 'S_GGP00000'      .
      clear flag.
      write: 30 g_path.
      move: g_path to it_mail-path.
      exit.
     endif.

     enddo.
      append it_mail.
      clear  it_mail.
    endloop.
***** End Code for the Path

***Code for the missing GL accounts
loop at diff_val2.

  read table it_diff2_values into miss2_val
       with key from2 = diff_val2-from.
    if sy-subrc <> 0.
     write:/ diff_val2-from(11).
     move: diff_val2-from to it_mail-setid.
      if diff_val2-from <> diff_val2-to.
       write: '-'.
       write: diff_val2-to.
       concatenate '-'  miss2_val-to2 into l_con.
       move: l_con to it_mail-to.
       clear l_con.
     endif.  " IF MISS2_VAL-FROM2 <> MISS2_VAL-TO2.
    endif.   " IF SY-SUBRC <> 0.

****** Code for the Path
    do.
     if flag is initial.
      g_store = diff_val2-setid.
     else.
      g_store = g_setid.
      clear g_setid.
     endif.

      select single setname into g_setid from setnode
            where subsetname = g_store.

     if flag is initial.
      concatenate g_setid ' -> ' g_store into g_path.
     else.
      concatenate g_setid ' -> ' g_path into g_path.
     endif.

     flag = 1.
    if g_setid = 'Z_GGP00000' or g_setid = 'S_GGP00000'      .
      clear flag.
      write: 30 g_path.
      move: g_path to it_mail-path.
      exit.
     endif.

     enddo.
      append it_mail.
      clear  it_mail.
    endloop.   " LOOP AT DIFF_VAL2.
```



```
** End Code for the missing GL accounts

else.

   l_text = text-020.
   write:/ l_text.
   move: l_text to it_mail-des.
   append it_mail.
   clear it_mail.


endif.    " IF NOT IT_DIFF2_VALUES[] IS INITIAL.



perform send_mail.

** No differences to report
if it_first_output[]   is initial and
   it_second_output[]  is initial and
   it_first_header[]   is initial and
   it_first_diff[]     is initial and
   diff_val1[]         is initial and
   diff_val2[]         is initial and
   it_diff1_values[]   is initial and
   it_diff2_values[]   is initial.
 skip.

else.
 if syst-batch = 'X'.
  message e006 with text-019.
 endif.
endif.

endform.                    " LIST_DISPLAY
*&---------------------------------------------------------------------*
*&      Form  SEND_MAIL
*&---------------------------------------------------------------------*
*       Mail Sending
*----------------------------------------------------------------------*
form send_mail .

* Structure to hold mail subject
data: wa_doc_chng like sodocchgi1.
* To hold info pertaining to SAPoffice: Description of Imported Object
* Components
data: i_objpack like sopcklsti1 occurs 2 with header line.
data: lt_pdf_nuc_data type standard table of solisti1.
* A structure to hold SAPoffice: Single List with Column Length 255
data: i_objtxt like solisti1 occurs 10 with header line.
* Structure to hold SAPoffice: Structure of the API Recipient List
data: i_reclist like somlreci1  occurs 5 with header line.
data : lt_upload type standard table of solisti1 with header line.
data w_lt_pdf_nuc_data like line of lt_pdf_nuc_data.

data: tab_lines type i.      " LIKE SY-TABIX.
data : l_cline type solisti1.

 data : lt_index type sy-tabix.
 data : doc_type(3) type c.
 data : descr like i_objpack-obj_descr.
 data :  temp_data like i_objpack-obj_descr.
 data : temp1 type string.
```



```
  wa_doc_chng-obj_descr  = text-016.

  wa_doc_chng-obj_langu = 'en'.
  wa_doc_chng-obj_expdat = sy-datum + 10 . " expirydate.


 i_objtxt = text-017.
 append  i_objtxt.
 clear i_objtxt.

concatenate text-010
            text-011
            into i_objtxt separated by space.
  append  i_objtxt.
 clear i_objtxt.

concatenate text-008
            g_group1
            text-009
            g_group2 into i_objtxt separated by space.
 append  i_objtxt.
 clear i_objtxt.


 describe table i_objtxt lines tab_lines .
  read table i_objtxt index tab_lines into l_cline.
  wa_doc_chng-doc_size  = ( tab_lines - 1 ) * 255 +
  strlen( l_cline ).
  clear i_objpack-transf_bin.
  i_objpack-head_start = 1.
  i_objpack-head_num   = 0.
  i_objpack-body_start = 1.
  i_objpack-body_num   = tab_lines.
  i_objpack-doc_type   = 'RAW'.    "'RAW'.
  append i_objpack.
  clear i_objpack.


    describe table it_mail lines tab_lines.
    tab_lines = tab_lines + 2.
    describe table lt_pdf_nuc_data lines lt_index.
    lt_index = lt_index + 1.


data       w_str(255) type c.  " To store E-Amil contents

    loop at it_mail.
       concatenate it_mail-setid
                   it_mail-des
                   it_mail-to
                   it_mail-path into
                   w_lt_pdf_nuc_data separated by space.
                   condense w_lt_pdf_nuc_data.
*        move w_lt_pdf_nuc_data to w_str.

      concatenate cl_abap_char_utilities=>cr_lf w_lt_pdf_nuc_data into
                    w_lt_pdf_nuc_data.
*        w_lt_pdf_nuc_data = w_str.
      append w_lt_pdf_nuc_data to lt_pdf_nuc_data.
      clear w_lt_pdf_nuc_data.
    endloop.

*read table it_mail
```



```
    i_objpack-transf_bin = 'X'.
    i_objpack-head_start = 0.
    i_objpack-head_num   = 0.
    i_objpack-body_start = lt_index.
    i_objpack-body_num   = tab_lines.
    i_objpack-doc_type   = 'TXT'. " doc_type.
* I_OBJPACK-OBJ_NAME   = 'Subject'.
    i_objpack-obj_descr  = text-018.
    i_objpack-doc_size   = tab_lines * 255 .
    append i_objpack.
    clear i_objpack.

    i_reclist-receiver = g_mail. "'sudheer.junnatla@ge.com'.

    i_reclist-rec_type = 'U'.
    append i_reclist.


      call function 'SO_NEW_DOCUMENT_ATT_SEND_API1'
                exporting
                     document_data              = wa_doc_chng
                     put_in_outbox              = 'X'
                     commit_work                = 'X'
                tables
                     packing_list               = i_objpack
*                     OBJECT_HEADER              = objhead
                     contents_bin               = lt_pdf_nuc_data
                     contents_txt               = i_objtxt
                     receivers                  = i_reclist
                exceptions
                     too_many_receivers         = 1
                     document_not_sent          = 2
                     operation_no_authorization = 4
                     others                     = 99.
      case sy-subrc.
    when 1.

    when 2.

    when 3.

    when 0.
      submit rsconn01 with mode = 'INT' and return.
  endcase.

endform.                    " SEND_MAIL
```


Disclaimer and Liability Notice

This document may discuss sample coding or other information that does not include SAP official interfaces and therefore is not supported by SAP. Changes made based on this information are not supported and can be overwritten during an upgrade.

SAP will not be held liable for any damages caused by using or misusing the information, code or methods suggested in this document, and anyone using these methods does so at his/her own risk.

SAP offers no guarantees and assumes no responsibility or liability of any type with respect to the content of this technical article or code sample, including any liability resulting from incompatibility between the content within this document and the materials and services offered by SAP. You agree that you will not hold, or seek to hold, SAP responsible or liable with respect to the content of this document.

