\*&---------------------------------------------------------------------\*  
\*& Report  ZBDC\_BAPI\_CREATE\_PO\_01  
\*&  
\*&---------------------------------------------------------------------\*  
\*&  
\*&  
\*&---------------------------------------------------------------------\*  
  
REPORT  zbdc\_bapi\_create\_po\_01.  
INCLUDE zbdc\_bapi\_create\_po\_01\_top.  
  
AT SELECTION-SCREEN ON VALUE-REQUEST FOR p\_path.  
  PERFORM get\_path.  
  
START-OF-SELECTION.  
  PERFORM upload\_data.  
  PERFORM validate\_and\_convert\_data.  
  PERFORM create\_po.  
  
  INCLUDE zbdc\_bapi\_create\_po\_01\_forms.

\*&---------------------------------------------------------------------\*  
\*&  Include           ZBDC\_BAPI\_CREATE\_PO\_01\_TOP  
\*&---------------------------------------------------------------------\*  
\*&---------------------------------------------------------------------\*  
\*&  Include           ZBDC\_CREATE\_PO\_01\_TOP  
\*&---------------------------------------------------------------------\*  
TYPES:  
 BEGIN OF t\_data,  
   lifnr(10)  TYPE c,  
   bsart(4)   TYPE c,  
   bedat(10)  TYPE c,  
   ekorg(4)   TYPE c,  
   ekgrp(3)   TYPE n,  
   matnr(18)  TYPE c,  
   menge(17)  TYPE c,  
   meins(3)   TYPE c,  
   eeind(10)  TYPE c,  
   werks(4)   TYPE c,  
   lgort(4)   TYPE c,  
 END OF t\_data,  
 t\_data\_tab TYPE STANDARD TABLE OF t\_data,  
  
  BEGIN OF t\_po\_head,  
    po     TYPE i,  
    lifnr TYPE ekko-lifnr,  
    bsart TYPE rm06e-bsart,  
\*    bedat TYPE rm06e-bedat,  
    bedat TYPE c LENGTH 10,  
    ekorg TYPE ekko-ekorg,  
    ekgrp TYPE ekko-ekgrp,  
 END OF t\_po\_head,  
 t\_po\_head\_tab TYPE STANDARD TABLE OF t\_po\_head,  
  
 BEGIN OF t\_po\_item,  
   po     TYPE i,  
   matnr  TYPE ekpo-ematn,  
\*   menge  TYPE ekpo-menge,  
   menge(6)  TYPE c,  
   meins(3)  TYPE c,  
   eeind  TYPE rm06e-eeind,  
   werks  TYPE ekpo-werks,  
\*   lgort  TYPE ekpo-lgort,  
   lgort(4)  TYPE n,  
 END OF t\_po\_item,  
 t\_po\_item\_tab TYPE STANDARD TABLE OF t\_po\_item,  
 BEGIN OF t\_log,  
   index TYPE sy-tabix,  
   msg TYPE string,  
   END OF t\_log,  
   t\_log\_tab TYPE STANDARD TABLE OF t\_log.  
  
  
DATA: it\_data       TYPE t\_data\_tab,  
      it\_po\_head    TYPE t\_po\_head\_tab,  
      it\_po\_item    TYPE t\_po\_item\_tab,  
      it\_log        TYPE t\_log\_tab,  
      it\_bdcdata    TYPE STANDARD TABLE OF bdcdata,  
      it\_msgcoll    TYPE STANDARD TABLE OF bdcmsgcoll,  
      wa\_data       TYPE t\_data,  
      wa\_po\_head    TYPE t\_po\_head,  
      wa\_po\_item    TYPE t\_po\_item.  
DATA: g\_index TYPE sy-tabix.  
  
PARAMETERS: p\_path TYPE ibipparms-path.

\*&---------------------------------------------------------------------\*  
\*&  Include           ZBDC\_BAPI\_CREATE\_PO\_01\_FORMS  
\*&---------------------------------------------------------------------\*  
\*&---------------------------------------------------------------------\*  
\*&      Form  GET\_PATH  
\*&---------------------------------------------------------------------\*  
\*       text  
\*----------------------------------------------------------------------\*  
\*  -->  p1        text  
\*  <--  p2        text  
\*----------------------------------------------------------------------\*  
FORM get\_path .  
  CALL FUNCTION 'F4\_FILENAME'  
    IMPORTING  
      file\_name = p\_path.  
ENDFORM.                    " GET\_PATH  
\*&---------------------------------------------------------------------\*  
\*&      Form  UPLOAD\_DATA  
\*&---------------------------------------------------------------------\*  
\*       text  
\*----------------------------------------------------------------------\*  
\*  -->  p1        text  
\*  <--  p2        text  
\*----------------------------------------------------------------------\*  
FORM upload\_data .  
  DATA:  
        l\_file TYPE rlgrap-filename,  
        l\_data TYPE truxs\_t\_text\_data.  
  MOVE p\_path TO l\_file.  
  CALL FUNCTION 'TEXT\_CONVERT\_XLS\_TO\_SAP'  
    EXPORTING  
      i\_field\_seperator    = 'X'  
      i\_line\_header        = 'X'  
      i\_tab\_raw\_data       = l\_data  
      i\_filename           = l\_file  
    TABLES  
      i\_tab\_converted\_data = it\_data  
    EXCEPTIONS  
      conversion\_failed    = 1  
      OTHERS               = 2.  
  IF sy-subrc <> 0.  
\* Implement suitable error handling here  
  ENDIF.  
ENDFORM.                    " UPLOAD\_DATA  
\*&---------------------------------------------------------------------\*  
\*&      Form  VALIDATE\_AND\_CONVERT\_DATA  
\*&---------------------------------------------------------------------\*  
\*       text  
\*----------------------------------------------------------------------\*  
\*  -->  p1        text  
\*  <--  p2        text  
\*----------------------------------------------------------------------\*  
FORM validate\_and\_convert\_data .  
  DATA: l\_index     TYPE i.  
  LOOP AT it\_data INTO wa\_data.  
    IF wa\_data-lifnr IS NOT INITIAL.  
      "Collect  Header Details  
      l\_index = l\_index + 1.  
      wa\_po\_head-po = l\_index.  
      "Document Type  
      wa\_po\_head-bsart = wa\_data-bsart.  
      "Vendor  
      CALL FUNCTION 'CONVERSION\_EXIT\_ALPHA\_OUTPUT'  
        EXPORTING  
          input  = wa\_data-lifnr  
        IMPORTING  
          output = wa\_po\_head-lifnr.  
      "Document Date  
      wa\_po\_head-bedat = wa\_data-bedat.  
      wa\_po\_head-ekorg = wa\_data-ekorg.  
      wa\_po\_head-ekgrp = wa\_data-ekgrp.  
      APPEND wa\_po\_head TO it\_po\_head.  
    ENDIF.  
    "Collect Item Details  
    wa\_po\_item-po     = l\_index.  
    "Material Number  
    CALL FUNCTION 'CONVERSION\_EXIT\_MATN1\_OUTPUT'  
      EXPORTING  
        input  = wa\_data-matnr  
      IMPORTING  
        output = wa\_po\_item-matnr.  
    "Quantity  
    wa\_po\_item-menge  = wa\_data-menge.  
    "Unit of Measurement  
    CALL FUNCTION 'CONVERSION\_EXIT\_CUNIT\_OUTPUT'  
      EXPORTING  
        input          = wa\_data-meins  
        language       = sy-langu  
      IMPORTING  
\*       LONG\_TEXT      =  
        output         = wa\_po\_item-meins  
\*       SHORT\_TEXT     =  
      EXCEPTIONS  
        unit\_not\_found = 1  
        OTHERS         = 2.  
    IF sy-subrc <> 0.  
\* Implement suitable error handling here  
    ENDIF.  
    "Item Delivery Date  
    wa\_po\_item-eeind = wa\_data-eeind.  
    "Plant  
    wa\_po\_item-werks = wa\_data-werks.  
    "Storage Location  
    wa\_po\_item-lgort = wa\_data-lgort.  
    APPEND wa\_po\_item TO it\_po\_item.  
  ENDLOOP.  
  SORT:  
   it\_po\_head BY po,  
   it\_po\_item BY po.  
ENDFORM.                    " VALIDATE\_AND\_CONVERT\_DATA  
\*&---------------------------------------------------------------------\*  
\*&      Form  CREATE\_PO  
\*&---------------------------------------------------------------------\*  
\*       text  
\*----------------------------------------------------------------------\*  
\*  -->  p1        text  
\*  <--  p2        text  
\*----------------------------------------------------------------------\*  
FORM create\_po .  
  DATA:  
        l\_po        TYPE bapimepoheader-po\_number,  
        lwa\_pohead  TYPE bapimepoheader,  
        lwa\_poheadx TYPE bapimepoheaderx,  
        lwa\_return   TYPE bapiret2,  
        lwa\_poitem   TYPE bapimepoitem,  
        lwa\_poitemx  TYPE bapimepoitemx,  
        lwa\_posch    TYPE bapimeposchedule,  
        lwa\_poschx   TYPE bapimeposchedulx.  
  
  DATA:  
        lt\_return   TYPE TABLE OF bapiret2,  
        lt\_poitem   TYPE TABLE OF bapimepoitem,  
        lt\_poitemx  TYPE TABLE OF bapimepoitemx,  
        lt\_posch    TYPE TABLE OF bapimeposchedule,  
        lt\_poschx   TYPE TABLE OF bapimeposchedulx.  
  LOOP AT it\_po\_head INTO wa\_po\_head.  
    lwa\_pohead-doc\_type   = wa\_po\_head-bsart.  
\*     lwa\_pohead-vendor     = wa\_po\_head-lifnr.  
    CALL FUNCTION 'CONVERSION\_EXIT\_ALPHA\_INPUT'  
      EXPORTING  
        input  = wa\_po\_head-lifnr  
      IMPORTING  
        output = lwa\_pohead-vendor.  
    lwa\_pohead-purch\_org  = wa\_po\_head-ekorg.  
    lwa\_pohead-pur\_group  = wa\_po\_head-ekgrp.  
  
    lwa\_poheadx-doc\_type   = 'X'.  
    lwa\_poheadx-vendor     = 'X'.  
    lwa\_poheadx-purch\_org  = 'X'.  
    lwa\_poheadx-pur\_group  = 'X'.  
    LOOP AT it\_po\_item INTO wa\_po\_item  
      WHERE po = wa\_po\_head-po.  
      lwa\_poitem-po\_item   = lwa\_poitem-po\_item + 10.  
      lwa\_poitem-material = wa\_po\_item-matnr.  
      lwa\_poitem-plant    = wa\_po\_item-werks.  
      lwa\_poitem-stge\_loc = wa\_po\_item-lgort.  
      lwa\_poitem-quantity = wa\_po\_item-menge.  
      APPEND lwa\_poitem TO lt\_poitem.  
      lwa\_poitemx-po\_item   = lwa\_poitemx-po\_item + 10.  
      lwa\_poitemx-material = 'X'.  
      lwa\_poitemx-plant    = 'X'.  
      lwa\_poitemx-stge\_loc = 'X'.  
      lwa\_poitemx-quantity = 'X'.  
      APPEND lwa\_poitemx TO lt\_poitemx.  
      lwa\_posch-po\_item       = lwa\_posch-po\_item + 10.  
      lwa\_posch-quantity      = wa\_po\_item-menge.  
      lwa\_posch-delivery\_date = wa\_po\_item-eeind.  
      APPEND lwa\_posch TO lt\_posch.  
      lwa\_poschx-po\_item       = lwa\_poschx-po\_item + 10.  
      lwa\_poschx-quantity      = 'X'.  
      lwa\_poschx-delivery\_date = 'X'.  
      APPEND lwa\_poschx TO lt\_poschx.  
    ENDLOOP.  
    CALL FUNCTION 'BAPI\_PO\_CREATE1'  
      EXPORTING  
        poheader         = lwa\_pohead  
        poheaderx        = lwa\_poheadx  
      IMPORTING  
        exppurchaseorder = l\_po  
      TABLES  
        return           = lt\_return  
        poitem           = lt\_poitem  
        poitemx          = lt\_poitemx  
        poschedule       = lt\_posch  
        poschedulex      = lt\_poschx.  
    WRITE / l\_po.  
    CLEAR:  
    lwa\_pohead,  
    lwa\_poheadx,  
    lwa\_poitem,  
    lwa\_poitemx,  
    lwa\_posch,  
    lwa\_poschx,  
    lt\_return,  
    lt\_poitem,  
    lt\_poitemx,  
    lt\_posch,  
    lt\_poschx.  
  ENDLOOP.  
ENDFORM.                    " CREATE\_PO