# ABAP Code

## Global Declaration

TYPE-POOLS: icon, kkblo.  
  
SELECTION-SCREEN BEGIN OF BLOCK b1 WITH FRAME.  
 SELECT-OPTIONS:  
 s\_matnr FOR mara-matnr,  
 s\_werks FOR ekpo-werks OBLIGATORY,  
 s\_lifnr FOR ekko-lifnr,  
 s\_mtart FOR mara-mtart.  
SELECTION-SCREEN END OF BLOCK b1.  
  
CONSTANTS:  
 gc\_dynnr\_0100 TYPE sy-dynnr VALUE '0100',  
 gc\_ucomm\_close TYPE syucomm VALUE 'ZCLOSE',  
 gc\_btn\_text TYPE string VALUE 'Close Selected Items',  
 gc\_btn\_quick TYPE string VALUE 'Set EKPO-ELIKZ = X for selected items'.  
  
TYPES: BEGIN OF ty\_out,  
 sel TYPE abap\_bool, "Selection indicator in ALV  
 ebeln TYPE ekpo-ebeln,  
 ebelp TYPE ekpo-ebelp,  
 elikz TYPE ekpo-elikz, "Delivery Completed Indicator  
 matnr TYPE ekpo-matnr,  
 txz01 TYPE ekpo-txz01,  
 mtart TYPE mara-mtart,  
 werks TYPE ekpo-werks,  
 lifnr TYPE ekko-lifnr,  
 name1 TYPE lfa1-name1,  
 menge TYPE ekpo-menge,  
 meins TYPE ekpo-meins,  
 aedat TYPE ekpo-aedat,  
 END OF ty\_out.  
  
TYPES:  
 tt\_out TYPE STANDARD TABLE OF ty\_out WITH NON-UNIQUE KEY ebeln ebelp,  
 tt\_ekko TYPE HASHED TABLE OF ekko WITH UNIQUE KEY ebeln,  
 tt\_ekpo TYPE STANDARD TABLE OF ekpo WITH NON-UNIQUE KEY ebeln ebelp,  
 tt\_eket TYPE STANDARD TABLE OF eket WITH NON-UNIQUE KEY ebeln ebelp evers etenr,  
 tt\_mara TYPE HASHED TABLE OF mara WITH UNIQUE KEY matnr,  
 tt\_lfa1 TYPE HASHED TABLE OF lfa1 WITH UNIQUE KEY lifnr.  
  
TYPES: BEGIN OF ty\_ekpo\_upd,  
 ebeln TYPE ekpo-ebeln,  
 ebelp TYPE ekpo-ebelp,  
 elikz TYPE ekpo-elikz,  
 END OF ty\_ekpo\_upd.  
TYPES tt\_ekpo\_upd TYPE STANDARD TABLE OF ty\_ekpo\_upd WITH UNIQUE KEY ebeln ebelp.  
  
DATA:  
 g\_repid TYPE sy-repid VALUE sy-repid,  
 g\_okcode TYPE sy-ucomm,  
 gs\_variant TYPE disvariant,  
 gv\_save TYPE c VALUE 'A'.  
  
" ALV references and settings  
DATA:  
 go\_container TYPE REF TO cl\_gui\_custom\_container,  
 go\_grid TYPE REF TO cl\_gui\_alv\_grid,  
 gt\_fcat TYPE lvc\_t\_fcat,  
 gs\_layout TYPE lvc\_s\_layo,  
 gt\_sort TYPE lvc\_t\_sort,  
 gt\_filt TYPE lvc\_t\_filt,  
 gt\_exclude TYPE ui\_functions.  
  
" Main data buffers  
DATA:  
 gt\_out TYPE tt\_out,  
 gs\_out TYPE ty\_out,  
 gt\_ekko TYPE tt\_ekko,  
 gt\_ekpo TYPE tt\_ekpo,  
 gt\_eket TYPE tt\_eket,  
 gt\_mara TYPE tt\_mara,  
 gt\_lfa1 TYPE tt\_lfa1,  
 gt\_ekpo\_upd TYPE tt\_ekpo\_upd.  
  
" Field-symbols  
FIELD-SYMBOLS:  
 <fs\_out> TYPE ty\_out.  
  
" Event handler for ALV  
CLASS lcl\_event\_handler DEFINITION PUBLIC FINAL CREATE PUBLIC.  
 PUBLIC SECTION.  
 METHODS on\_toolbar  
 FOR EVENT toolbar OF cl\_gui\_alv\_grid  
 IMPORTING e\_object e\_interactive.  
 METHODS on\_user\_command  
 FOR EVENT user\_command OF cl\_gui\_alv\_grid  
 IMPORTING e\_ucomm.  
ENDCLASS.

## Selection Screen

SELECTION-SCREEN BEGIN OF BLOCK b1 WITH FRAME.  
 SELECT-OPTIONS:  
 s\_matnr FOR mara-matnr,  
 s\_werks FOR ekpo-werks OBLIGATORY,  
 s\_lifnr FOR ekko-lifnr,  
 s\_mtart FOR mara-mtart.  
SELECTION-SCREEN END OF BLOCK b1.

## Processing Logic

CLASS lcl\_event\_handler IMPLEMENTATION.  
 METHOD on\_toolbar.  
 DATA ls\_button TYPE stb\_button.  
 CLEAR ls\_button.  
 ls\_button-function = gc\_ucomm\_close.  
 ls\_button-icon = icon\_complete.  
 ls\_button-quickinfo = gc\_btn\_quick.  
 ls\_button-text = gc\_btn\_text.  
 ls\_button-butn\_type = 0.  
 APPEND ls\_button TO e\_object->mt\_toolbar.  
 ENDMETHOD.  
  
 METHOD on\_user\_command.  
 CASE e\_ucomm.  
 WHEN gc\_ucomm\_close.  
 PERFORM close\_selected.  
 ENDCASE.  
 ENDMETHOD.  
ENDCLASS.  
  
DATA go\_handler TYPE REF TO lcl\_event\_handler.  
  
START-OF-SELECTION.  
 PERFORM get\_data.  
 IF gt\_out IS INITIAL.  
 MESSAGE 'No open PO items found for the given selection.' TYPE 'S'.  
 ENDIF.  
 CALL SCREEN gc\_dynnr\_0100.  
  
MODULE status\_0100 OUTPUT.  
 IF go\_container IS INITIAL.  
 CREATE OBJECT go\_container  
 EXPORTING  
 container\_name = 'CC\_ALV'.  
 CREATE OBJECT go\_grid  
 EXPORTING  
 i\_parent = go\_container.  
  
 CREATE OBJECT go\_handler.  
 SET HANDLER go\_handler->on\_toolbar FOR go\_grid.  
 SET HANDLER go\_handler->on\_user\_command FOR go\_grid.  
  
 PERFORM build\_fieldcatalog.  
 PERFORM build\_layout.  
  
 gs\_variant-report = g\_repid.  
  
 CALL METHOD go\_grid->set\_table\_for\_first\_display  
 EXPORTING  
 is\_variant = gs\_variant  
 i\_save = gv\_save  
 is\_layout = gs\_layout  
 CHANGING  
 it\_outtab = gt\_out  
 it\_fieldcatalog = gt\_fcat  
 it\_sort = gt\_sort  
 it\_filter = gt\_filt.  
  
 ELSE.  
 CALL METHOD go\_grid->refresh\_table\_display  
 EXCEPTIONS  
 finished = 1  
 OTHERS = 2.  
 ENDIF.  
ENDMODULE.  
  
MODULE user\_command\_0100 INPUT.  
 g\_okcode = sy-ucomm.  
 CLEAR sy-ucomm.  
 CASE g\_okcode.  
 WHEN 'BACK' OR 'CANC' OR 'EXIT'.  
 LEAVE TO SCREEN 0.  
 ENDCASE.  
ENDMODULE.  
  
FORM get\_data.  
 DATA: lt\_ebeln TYPE SORTED TABLE OF ekko-ebeln WITH UNIQUE KEY table\_line,  
 lt\_lifnr TYPE SORTED TABLE OF lfa1-lifnr WITH UNIQUE KEY table\_line.  
  
 DATA: lt\_mara\_matnr TYPE SORTED TABLE OF mara-matnr WITH UNIQUE KEY table\_line.  
  
 CLEAR: gt\_out, gt\_ekko, gt\_ekpo, gt\_eket, gt\_mara, gt\_lfa1.  
  
 " Materials by MTART and/or MATNR  
 IF s\_mtart[] IS NOT INITIAL AND s\_matnr[] IS NOT INITIAL.  
 SELECT matnr mtart  
 FROM mara  
 INTO TABLE gt\_mara  
 WHERE mtart IN s\_mtart  
 AND matnr IN s\_matnr.  
 ELSEIF s\_mtart[] IS NOT INITIAL.  
 SELECT matnr mtart  
 FROM mara  
 INTO TABLE gt\_mara  
 WHERE mtart IN s\_mtart.  
 ELSEIF s\_matnr[] IS NOT INITIAL.  
 " Optional: prefetch MTART for given MATNRs  
 SELECT matnr mtart  
 FROM mara  
 INTO TABLE gt\_mara  
 WHERE matnr IN s\_matnr.  
 ENDIF.  
  
 " EKPO selection (open items only)  
 IF gt\_mara IS NOT INITIAL.  
 SELECT ebeln ebelp elikz matnr txz01 werks menge meins aedat  
 FROM ekpo  
 INTO TABLE gt\_ekpo  
 FOR ALL ENTRIES IN gt\_mara  
 WHERE werks IN s\_werks  
 AND elikz = space  
 AND matnr = gt\_mara-matnr.  
 ELSEIF s\_matnr[] IS NOT INITIAL.  
 SELECT ebeln ebelp elikz matnr txz01 werks menge meins aedat  
 FROM ekpo  
 INTO TABLE gt\_ekpo  
 WHERE werks IN s\_werks  
 AND elikz = space  
 AND matnr IN s\_matnr.  
 ELSE.  
 SELECT ebeln ebelp elikz matnr txz01 werks menge meins aedat  
 FROM ekpo  
 INTO TABLE gt\_ekpo  
 WHERE werks IN s\_werks  
 AND elikz = space.  
 ENDIF.  
  
 IF gt\_ekpo IS INITIAL.  
 RETURN.  
 ENDIF.  
  
 " Collect EBELN for EKKO  
 DATA ls\_ekpo LIKE LINE OF gt\_ekpo.  
 LOOP AT gt\_ekpo INTO ls\_ekpo.  
 INSERT ls\_ekpo-ebeln INTO TABLE lt\_ebeln.  
 ENDLOOP.  
  
 " EKKO with optional vendor restriction  
 IF s\_lifnr[] IS NOT INITIAL.  
 SELECT ebeln lifnr  
 FROM ekko  
 INTO TABLE gt\_ekko  
 FOR ALL ENTRIES IN lt\_ebeln  
 WHERE ebeln = lt\_ebeln-table\_line  
 AND lifnr IN s\_lifnr.  
 ELSE.  
 SELECT ebeln lifnr  
 FROM ekko  
 INTO TABLE gt\_ekko  
 FOR ALL ENTRIES IN lt\_ebeln  
 WHERE ebeln = lt\_ebeln-table\_line.  
 ENDIF.  
  
 IF gt\_ekko IS INITIAL.  
 CLEAR gt\_ekpo.  
 RETURN.  
 ENDIF.  
  
 " Filter EKPO by EKKO (if vendor-restricted)  
 DATA: lt\_ekpo\_f TYPE tt\_ekpo,  
 ls\_ekko TYPE ekko.  
 LOOP AT gt\_ekpo INTO ls\_ekpo.  
 READ TABLE gt\_ekko INTO ls\_ekko WITH KEY ebeln = ls\_ekpo-ebeln.  
 IF sy-subrc = 0.  
 APPEND ls\_ekpo TO lt\_ekpo\_f.  
 ENDIF.  
 ENDLOOP.  
 gt\_ekpo = lt\_ekpo\_f.  
 IF gt\_ekpo IS INITIAL.  
 RETURN.  
 ENDIF.  
  
 " Vendor master  
 CLEAR lt\_lifnr.  
 LOOP AT gt\_ekko INTO ls\_ekko.  
 INSERT ls\_ekko-lifnr INTO TABLE lt\_lifnr.  
 ENDLOOP.  
 IF lt\_lifnr IS NOT INITIAL.  
 SELECT lifnr name1  
 FROM lfa1  
 INTO TABLE gt\_lfa1  
 FOR ALL ENTRIES IN lt\_lifnr  
 WHERE lifnr = lt\_lifnr-table\_line.  
 ENDIF.  
  
 " Ensure MARA present for all result materials (to show MTART)  
 DATA: lt\_out\_matnr TYPE SORTED TABLE OF mara-matnr WITH UNIQUE KEY table\_line,  
 ls\_mara TYPE mara.  
 LOOP AT gt\_ekpo INTO ls\_ekpo.  
 IF ls\_ekpo-matnr IS NOT INITIAL.  
 INSERT ls\_ekpo-matnr INTO TABLE lt\_out\_matnr.  
 ENDIF.  
 ENDLOOP.  
 IF lt\_out\_matnr IS NOT INITIAL.  
 SELECT matnr mtart  
 FROM mara  
 INTO TABLE gt\_mara  
 FOR ALL ENTRIES IN lt\_out\_matnr  
 WHERE matnr = lt\_out\_matnr-table\_line.  
 ENDIF.  
  
 " Build output  
 DATA: ls\_out TYPE ty\_out.  
 CLEAR gt\_out.  
 LOOP AT gt\_ekpo INTO ls\_ekpo.  
 CLEAR ls\_out.  
 ls\_out-ebeln = ls\_ekpo-ebeln.  
 ls\_out-ebelp = ls\_ekpo-ebelp.  
 ls\_out-elikz = ls\_ekpo-elikz.  
 ls\_out-matnr = ls\_ekpo-matnr.  
 ls\_out-txz01 = ls\_ekpo-txz01.  
 ls\_out-werks = ls\_ekpo-werks.  
 ls\_out-menge = ls\_ekpo-menge.  
 ls\_out-meins = ls\_ekpo-meins.  
 ls\_out-aedat = ls\_ekpo-aedat.  
  
 READ TABLE gt\_ekko INTO ls\_ekko WITH KEY ebeln = ls\_ekpo-ebeln.  
 IF sy-subrc = 0.  
 ls\_out-lifnr = ls\_ekko-lifnr.  
 DATA(ls\_lfa1) = VALUE lfa1( ).  
 READ TABLE gt\_lfa1 INTO ls\_lfa1 WITH KEY lifnr = ls\_ekko-lifnr.  
 IF sy-subrc = 0.  
 ls\_out-name1 = ls\_lfa1-name1.  
 ENDIF.  
 ENDIF.  
  
 CLEAR ls\_mara.  
 READ TABLE gt\_mara INTO ls\_mara WITH KEY matnr = ls\_ekpo-matnr.  
 IF sy-subrc = 0.  
 ls\_out-mtart = ls\_mara-mtart.  
 ENDIF.  
  
 ls\_out-sel = abap\_false.  
 APPEND ls\_out TO gt\_out.  
 ENDLOOP.  
  
 SORT gt\_out BY ebeln ebelp.  
ENDFORM.  
  
FORM build\_fieldcatalog.  
 DATA ls\_fcat TYPE lvc\_s\_fcat.  
 CLEAR gt\_fcat.  
  
 " Checkbox selection  
 CLEAR ls\_fcat.  
 ls\_fcat-fieldname = 'SEL'.  
 ls\_fcat-col\_text = 'Sel'.  
 ls\_fcat-checkbox = 'X'.  
 ls\_fcat-edit = 'X'.  
 ls\_fcat-outputlen = 3.  
 APPEND ls\_fcat TO gt\_fcat.  
  
 CLEAR ls\_fcat.  
 ls\_fcat-fieldname = 'EBELN'.  
 ls\_fcat-ref\_table = 'EKPO'.  
 ls\_fcat-ref\_field = 'EBELN'.  
 ls\_fcat-key = 'X'.  
 ls\_fcat-col\_pos = 2.  
 APPEND ls\_fcat TO gt\_fcat.  
  
 CLEAR ls\_fcat.  
 ls\_fcat-fieldname = 'EBELP'.  
 ls\_fcat-ref\_table = 'EKPO'.  
 ls\_fcat-ref\_field = 'EBELP'.  
 ls\_fcat-key = 'X'.  
 ls\_fcat-col\_pos = 3.  
 APPEND ls\_fcat TO gt\_fcat.  
  
 CLEAR ls\_fcat.  
 ls\_fcat-fieldname = 'MATNR'.  
 ls\_fcat-ref\_table = 'EKPO'.  
 ls\_fcat-ref\_field = 'MATNR'.  
 ls\_fcat-col\_pos = 4.  
 APPEND ls\_fcat TO gt\_fcat.  
  
 CLEAR ls\_fcat.  
 ls\_fcat-fieldname = 'TXZ01'.  
 ls\_fcat-ref\_table = 'EKPO'.  
 ls\_fcat-ref\_field = 'TXZ01'.  
 ls\_fcat-col\_pos = 5.  
 APPEND ls\_fcat TO gt\_fcat.  
  
 CLEAR ls\_fcat.  
 ls\_fcat-fieldname = 'MTART'.  
 ls\_fcat-ref\_table = 'MARA'.  
 ls\_fcat-ref\_field = 'MTART'.  
 ls\_fcat-col\_pos = 6.  
 APPEND ls\_fcat TO gt\_fcat.  
  
 CLEAR ls\_fcat.  
 ls\_fcat-fieldname = 'WERKS'.  
 ls\_fcat-ref\_table = 'EKPO'.  
 ls\_fcat-ref\_field = 'WERKS'.  
 ls\_fcat-col\_pos = 7.  
 APPEND ls\_fcat TO gt\_fcat.  
  
 CLEAR ls\_fcat.  
 ls\_fcat-fieldname = 'LIFNR'.  
 ls\_fcat-ref\_table = 'EKKO'.  
 ls\_fcat-ref\_field = 'LIFNR'.  
 ls\_fcat-col\_pos = 8.  
 APPEND ls\_fcat TO gt\_fcat.  
  
 CLEAR ls\_fcat.  
 ls\_fcat-fieldname = 'NAME1'.  
 ls\_fcat-ref\_table = 'LFA1'.  
 ls\_fcat-ref\_field = 'NAME1'.  
 ls\_fcat-col\_pos = 9.  
 APPEND ls\_fcat TO gt\_fcat.  
  
 CLEAR ls\_fcat.  
 ls\_fcat-fieldname = 'MENGE'.  
 ls\_fcat-ref\_table = 'EKPO'.  
 ls\_fcat-ref\_field = 'MENGE'.  
 ls\_fcat-do\_sum = 'X'.  
 ls\_fcat-col\_pos = 10.  
 APPEND ls\_fcat TO gt\_fcat.  
  
 CLEAR ls\_fcat.  
 ls\_fcat-fieldname = 'MEINS'.  
 ls\_fcat-ref\_table = 'EKPO'.  
 ls\_fcat-ref\_field = 'MEINS'.  
 ls\_fcat-col\_pos = 11.  
 APPEND ls\_fcat TO gt\_fcat.  
  
 CLEAR ls\_fcat.  
 ls\_fcat-fieldname = 'AEDAT'.  
 ls\_fcat-ref\_table = 'EKPO'.  
 ls\_fcat-ref\_field = 'AEDAT'.  
 ls\_fcat-col\_pos = 12.  
 APPEND ls\_fcat TO gt\_fcat.  
  
 CLEAR ls\_fcat.  
 ls\_fcat-fieldname = 'ELIKZ'.  
 ls\_fcat-ref\_table = 'EKPO'.  
 ls\_fcat-ref\_field = 'ELIKZ'.  
 ls\_fcat-col\_pos = 13.  
 APPEND ls\_fcat TO gt\_fcat.  
  
 " Default sort by EBELN/EBELP  
 CLEAR gt\_sort.  
 DATA ls\_sort TYPE lvc\_s\_sort.  
 CLEAR ls\_sort.  
 ls\_sort-fieldname = 'EBELN'.  
 ls\_sort-up = 'X'.  
 ls\_sort-subtot = 'X'.  
 APPEND ls\_sort TO gt\_sort.  
  
 CLEAR ls\_sort.  
 ls\_sort-fieldname = 'EBELP'.  
 ls\_sort-up = 'X'.  
 APPEND ls\_sort TO gt\_sort.  
ENDFORM.  
  
FORM build\_layout.  
 CLEAR gs\_layout.  
 gs\_layout-zebra = 'X'.  
 gs\_layout-cwidth\_opt = 'X'.  
 gs\_layout-box\_fname = 'SEL'.  
 gs\_layout-sel\_mode = 'A'.  
ENDFORM.  
  
FORM close\_selected.  
 DATA: lv\_updated TYPE i VALUE 0.  
  
 IF go\_grid IS BOUND.  
 CALL METHOD go\_grid->check\_changed\_data.  
 ENDIF.  
  
 CLEAR gt\_ekpo\_upd.  
 LOOP AT gt\_out ASSIGNING <fs\_out> WHERE sel = abap\_true AND elikz IS INITIAL.  
 APPEND VALUE ty\_ekpo\_upd( ebeln = <fs\_out>-ebeln  
 ebelp = <fs\_out>-ebelp  
 elikz = 'X' ) TO gt\_ekpo\_upd.  
 ENDLOOP.  
  
 IF gt\_ekpo\_upd IS INITIAL.  
 MESSAGE 'Select at least one open item to close.' TYPE 'S'.  
 RETURN.  
 ENDIF.  
  
 " Update database  
 MODIFY ekpo FROM TABLE gt\_ekpo\_upd TRANSPORTING elikz.  
 IF sy-subrc = 0.  
 COMMIT WORK AND WAIT.  
 ELSE.  
 ROLLBACK WORK.  
 MESSAGE 'Update failed for selected items.' TYPE 'E'.  
 RETURN.  
 ENDIF.  
  
 " Reflect changes in output  
 DATA ls\_upd TYPE ty\_ekpo\_upd.  
 LOOP AT gt\_ekpo\_upd INTO ls\_upd.  
 READ TABLE gt\_out ASSIGNING <fs\_out> WITH KEY ebeln = ls\_upd-ebeln ebelp = ls\_upd-ebelp.  
 IF sy-subrc = 0.  
 <fs\_out>-elikz = 'X'.  
 <fs\_out>-sel = abap\_false.  
 lv\_updated = lv\_updated + 1.  
 ENDIF.  
 ENDLOOP.  
  
 IF go\_grid IS BOUND.  
 CALL METHOD go\_grid->refresh\_table\_display  
 EXPORTING  
 is\_stable = VALUE lvc\_s\_stbl( row = 'X' col = 'X' )  
 EXCEPTIONS  
 finished = 1  
 OTHERS = 2.  
 ENDIF.  
  
 MESSAGE |Items closed: { lv\_updated }| TYPE 'S'.  
ENDFORM.

## Output Display

MODULE status\_0100 OUTPUT.  
 IF go\_container IS INITIAL.  
 CREATE OBJECT go\_container  
 EXPORTING  
 container\_name = 'CC\_ALV'.  
 CREATE OBJECT go\_grid  
 EXPORTING  
 i\_parent = go\_container.  
  
 CREATE OBJECT go\_handler.  
 SET HANDLER go\_handler->on\_toolbar FOR go\_grid.  
 SET HANDLER go\_handler->on\_user\_command FOR go\_grid.  
  
 PERFORM build\_fieldcatalog.  
 PERFORM build\_layout.  
  
 gs\_variant-report = g\_repid.  
  
 CALL METHOD go\_grid->set\_toolbar\_interactive.  
  
 CALL METHOD go\_grid->set\_table\_for\_first\_display  
 EXPORTING  
 is\_variant = gs\_variant  
 i\_save = gv\_save  
 is\_layout = gs\_layout  
 CHANGING  
 it\_outtab = gt\_out  
 it\_fieldcatalog = gt\_fcat  
 it\_sort = gt\_sort  
 it\_filter = gt\_filt.  
  
 ELSE.  
 CALL METHOD go\_grid->refresh\_table\_display  
 EXCEPTIONS  
 finished = 1  
 OTHERS = 2.  
 ENDIF.  
ENDMODULE.  
  
MODULE user\_command\_0100 INPUT.  
 g\_okcode = sy-ucomm.  
 CLEAR sy-ucomm.  
 CASE g\_okcode.  
 WHEN 'BACK' OR 'CANC' OR 'EXIT'.  
 LEAVE TO SCREEN 0.  
 ENDCASE.  
ENDMODULE.