```
FUNCTION zkhpm_dock_door_occup.
*"----------------------------------------------------------------------
*"*"Local Interface:
*"  IMPORTING
*"     REFERENCE(IV_LGNUM) TYPE  /SCWM/LGNUM DEFAULT '0001'
*"     REFERENCE(IV_VARIANT) TYPE  VARIANT
*"     REFERENCE(IV_MODE) TYPE  /SCWM/DE_MON_FM_MODE DEFAULT '1'
*"  EXPORTING
*"     REFERENCE(EV_RETURNCODE) TYPE  XFELD
*"     REFERENCE(EV_VARIANT) TYPE  VARIANT
*"     REFERENCE(ET_DATA) TYPE  ZKHPM_DOCK_DOOR_OCCUP_TT
*"  CHANGING
*"     REFERENCE(CT_TAB_RANGE) TYPE  RSDS_TRANGE OPTIONAL
*"     REFERENCE(CT_FIELDCAT) TYPE  LVC_T_FCAT OPTIONAL
*"----------------------------------------------------------------------

*****************************************************************************************
  "                            *ASIAN PAINTS PMG KHANDALA*
*****************************************************************************************
  "                               *OBJECT INFORMATION*
*****************************************************************************************
* Module : EWM
* Developer : Adam Siraj
* Requestor : Shubham Shete
* Date Of Creation : 14.11.2023
* Transport Request : KSDK901169, KSDK901165
* Development Object : ZKHPM_DOCK_DOOR_OCCUP
* Description : FM for dock door occupancy status report
*=======================================================================================*
*                               *MODIFICATION HISTORY*
*---------------------------------------------------------------------------------------*
* Mod. No. Date Name Transport Number Change Desc. *
*---------------------------------------------------------------------------------------*
* 1 <25/10/21> <DeveloperÌs Name> <Mod. TR number> <short description> *
* *
*---------------------------------------------------------------------------------------*

  DATA lt_docno     TYPE /scwm/dlv_docno_itemno_tab.
  DATA lt_selection    TYPE  /scwm/dlv_selection_tab.
  DATA wa_include   TYPE /scwm/dlv_query_incl_str_prd.
  DATA lt_mapping   TYPE /scwm/tt_map_selopt2field.
  DATA(lo_prd) = NEW /scwm/cl_dlv_management_prd( ).

  TYPES: BEGIN OF gty_door,
           door TYPE /scwm/de_door,
         END OF gty_door.

  DATA: gt_door      TYPE STANDARD TABLE OF gty_door INITIAL SIZE 1,
        gt_table     TYPE zkhpm_dock_door_occup_tt,
        ls_t300_md   TYPE /scwm/s_t300_md,
        ls_selection TYPE /scwm/dlv_selection_str.

  CLEAR: et_data, s_docno[], s_door[], ct_tab_range.

  IF iv_mode = 1.
    CALL SELECTION-SCREEN 0100 STARTING AT 10 10 ENDING AT 150 150.
    IF sy-subrc <> 0.
      ev_returncode = abap_true.
      RETURN.
    ENDIF.
  ENDIF.
  IF  iv_variant IS NOT INITIAL.
*   Use the selection criteria from a pre-defined variant without
*   presenting a selection screen
    CALL FUNCTION 'RS_SUPPORT_SELECTIONS'
      EXPORTING
        report               = sy-repid
        variant              = iv_variant
      EXCEPTIONS
        variant_not_existent = 1
        variant_obsolete     = 2
        OTHERS               = 3.
    IF sy-subrc <> 0.
      MESSAGE ID sy-msgid TYPE sy-msgty NUMBER sy-msgno
              WITH sy-msgv1 sy-msgv2 sy-msgv3 sy-msgv4.
    ENDIF.
  ENDIF.

  " Move the range table data to select options when running in refresh mode
  IF ct_tab_range IS NOT INITIAL.
*   the table it_tab_range contains the selection criteria, which
*   have been passed to the function module
*   these selection criteria should be visible in the selection screen
    CALL FUNCTION '/SCWM/RANGETAB2SELOPT'
      EXPORTING
        iv_repid     = sy-repid
        iv_lgnum     = iv_lgnum
        it_mapping   = lt_mapping
      CHANGING
        ct_tab_range = ct_tab_range.
  ENDIF.

  " Move the selection option details to the Range Table ( Changing parameters ) when running in fo
  TRY.
      CALL FUNCTION '/SCWM/SELOPT2RANGETAB'
        EXPORTING
          iv_repid     = sy-repid
          it_mapping   = lt_mapping
        IMPORTING
          et_tab_range = ct_tab_range.
    CATCH /scwm/cx_mon_noexec.
  ENDTRY.

  CALL FUNCTION '/SCWM/T300_MD_READ_SINGLE'
    EXPORTING
      iv_lgnum   = p_lgnum
*     IV_SCUGUID =
    IMPORTING
      es_t300_md = ls_t300_md
    EXCEPTIONS
      not_found  = 1
      OTHERS     = 2.

  IF sy-subrc EQ 0.
    CLEAR ls_selection.
    ls_selection-fieldname = /scdl/if_dl_logfname_c=>sc_locationid_wh_h.
    ls_selection-sign      = wmegc_sign_inclusive.
    ls_selection-option    = wmegc_option_eq.
    ls_selection-low       = ls_t300_md-scuguid.
    APPEND ls_selection TO lt_selection.
  ENDIF.
  wa_include = VALUE #(
   head_status   = abap_true
   item_status   = abap_true
   head_refdoc   = abap_true
   item_refdoc = abap_true ).

  IF s_door IS NOT INITIAL.

    LOOP AT s_door ASSIGNING FIELD-SYMBOL(<ss_door>).
      CLEAR ls_selection.
      ls_selection-fieldname = '/SCWM/DOOR_I'.
      ls_selection-sign      = <ss_door>-sign.
      ls_selection-option    = <ss_door>-option.
      ls_selection-low       = <ss_door>-low.
      ls_selection-high = <ss_door>-high.
      APPEND ls_selection TO lt_selection.
    ENDLOOP.

    TRY.
        lo_prd->query(
          EXPORTING
            iv_whno          = p_lgnum
            it_selection     = lt_selection
*           iv_doccat        = /scdl/if_dl_doc_c=>sc_doccat_inb_prd
*           is_read_options = ls_read_options
            is_include_data = wa_include
          IMPORTING
            et_headers       = DATA(lt_headers)
            et_items         = DATA(lt_items) ).
      CATCH /scdl/cx_delivery.                          "#EC NO_HANDLER
    ENDTRY.

    LOOP AT lt_items INTO DATA(lwa_items).

      DATA(lwa_headers) = VALUE #( lt_headers[ docid = lwa_items-docid ] OPTIONAL ). " Alternative
      IF lwa_headers IS INITIAL.
        CONTINUE.
      ENDIF.
      DATA(lv_gr_status) = VALUE #( lwa_headers-status[ status_type = /scdl/if_dl_status_c=>sc_t_go
      DATA(lv_pw_status) = VALUE #( lwa_headers-status[ status_type = /scdl/if_dl_status_c=>sc_t_pu
      DATA(lv_pick_status) = VALUE #( lwa_headers-status[ status_type = /scdl/if_dl_status_c=>sc_t_
      DATA(lv_gi_status) = VALUE #( lwa_headers-status[ status_type = /scdl/if_dl_status_c=>sc_t_go

      IF lv_gr_status IS NOT INITIAL.
        SELECT SINGLE /scwm/door FROM /scdl/db_proci_i INTO @lv_door WHERE docid EQ @lwa_items-doci
      ELSE.
        SELECT SINGLE /scwm/door FROM /scdl/db_proci_o INTO @lv_door WHERE docid EQ @lwa_items-doci
      ENDIF.

      APPEND VALUE #(
      door = lv_door
      docno       = lwa_headers-docno
      refdocno      = VALUE #( lwa_headers-refdoc[ refdoccat = 'ERP' ]-refdocno OPTIONAL )
      status_gr   = SWITCH #( lv_gr_status
                      WHEN '0' THEN TEXT-002
                      WHEN '1' THEN TEXT-003
                      WHEN '2' THEN TEXT-004
                      WHEN '9' THEN TEXT-005 )

      status_putaway = SWITCH #( lv_pw_status
                              WHEN '0' THEN TEXT-002
                              WHEN '1' THEN TEXT-003
                              WHEN '2' THEN TEXT-004
                              WHEN '9' THEN TEXT-005 )

        status_picking   = SWITCH #( lv_pick_status
                              WHEN '0' THEN TEXT-002
                              WHEN '1' THEN TEXT-003
                              WHEN '2' THEN TEXT-004
                              WHEN '9' THEN TEXT-005 )

        status_gi = SWITCH #( lv_gi_status
                              WHEN '0' THEN TEXT-002
                              WHEN '1' THEN TEXT-003
                              WHEN '2' THEN TEXT-004
                              WHEN '9' THEN TEXT-005 )
           ) TO et_data.
      ENDLOOP.

      DELETE et_data WHERE status_gi = TEXT-005. "gc_completed.
      DELETE et_data WHERE status_gr = TEXT-005. "gc_completed.

    ENDIF.

    IF s_docno IS NOT INITIAL.

      LOOP AT s_docno ASSIGNING FIELD-SYMBOL(<ss_docno>).
        CLEAR ls_selection.
        ls_selection-fieldname = /scdl/if_dl_logfname_c=>sc_docno_h.
        ls_selection-sign      = <ss_docno>-sign.
        ls_selection-option    = <ss_docno>-option.
        ls_selection-low       = <ss_docno>-low.
        ls_selection-high = <ss_docno>-high.
        APPEND ls_selection TO lt_selection.
      ENDLOOP.

      TRY.
          lo_prd->query(
            EXPORTING
              iv_whno         = p_lgnum
              it_selection    = lt_selection
*             iv_doccat       = /scdl/if_dl_doc_c=>sc_doccat_inb_prd
*             is_read_options = ls_read_options
              is_include_data = wa_include
            IMPORTING
              et_headers      = lt_headers
              et_items        = lt_items ).
        CATCH /scdl/cx_delivery.                          "#EC NO_HANDLER
      ENDTRY.

      LOOP AT lt_items INTO lwa_items.

        lwa_headers = VALUE #( lt_headers[ docid = lwa_items-docid ] OPTIONAL ). " Alternative READ T
        IF lwa_headers IS INITIAL.
          CONTINUE.
        ENDIF.
        lv_gr_status = VALUE #( lwa_headers-status[ status_type = /scdl/if_dl_status_c=>sc_t_goods_re
        lv_pw_status = VALUE #( lwa_headers-status[ status_type = /scdl/if_dl_status_c=>sc_t_putaway
        lv_pick_status = VALUE #( lwa_headers-status[ status_type = /scdl/if_dl_status_c=>sc_t_pickin
        lv_gi_status = VALUE #( lwa_headers-status[ status_type = /scdl/if_dl_status_c=>sc_t_goods_is
        IF lv_gr_status IS NOT INITIAL.
          SELECT SINGLE /scwm/door FROM /scdl/db_proci_i INTO @lv_door WHERE docid EQ @lwa_items-doci
        ELSE.
          SELECT SINGLE /scwm/door FROM /scdl/db_proci_o INTO @lv_door WHERE docid EQ @lwa_items-doci
        ENDIF.

        APPEND VALUE #(
        door = lv_door
        docno       = lwa_headers-docno
        refdocno      = VALUE #( lwa_headers-refdoc[ refdoccat = 'ERP' ]-refdocno OPTIONAL )
        status_gr   = SWITCH #( lv_gr_status
                              WHEN '0' THEN TEXT-002
                              WHEN '1' THEN TEXT-003
                              WHEN '2' THEN TEXT-004
                              WHEN '9' THEN TEXT-005 )

        status_putaway = SWITCH #( lv_pw_status
                              WHEN '0' THEN TEXT-002
                              WHEN '1' THEN TEXT-003
                              WHEN '2' THEN TEXT-004
                              WHEN '9' THEN TEXT-005 )

        status_picking   = SWITCH #( lv_pick_status
                              WHEN '0' THEN TEXT-002
                              WHEN '1' THEN TEXT-003
                              WHEN '2' THEN TEXT-004
                              WHEN '9' THEN TEXT-005 )

        status_gi = SWITCH #( lv_gi_status
                              WHEN '0' THEN TEXT-002
                              WHEN '1' THEN TEXT-003
                              WHEN '2' THEN TEXT-004
                              WHEN '9' THEN TEXT-005 ) ) TO et_data.
      ENDLOOP.

      DELETE et_data WHERE status_gi = TEXT-005. "gc_completed.
      DELETE et_data WHERE status_gr = TEXT-005. "gc_completed.

    ENDIF.

    IF s_door IS INITIAL AND s_docno IS INITIAL.

      SELECT door FROM /scwm/tdoor
      INTO TABLE gt_door
      WHERE lgnum = p_lgnum
      AND   door IN s_door
      AND   load_dir IN ('B','O','I').

      LOOP AT gt_door ASSIGNING FIELD-SYMBOL(<ss1_door>).
        CLEAR ls_selection.
        ls_selection-fieldname = '/SCWM/DOOR_I'.
        ls_selection-sign      = wmegc_sign_inclusive.
        ls_selection-option    = wmegc_option_eq.
        ls_selection-low       = <ss1_door>-door.
  *     ls_selection-high = <ss1_door>-high.
        APPEND ls_selection TO lt_selection.
      ENDLOOP.

      TRY.
          lo_prd->query(
            EXPORTING
```

```
            iv_whno          = p_lgnum
            it_selection     = lt_selection
*           iv_doccat        = /scdl/if_dl_doc_c=>sc_doccat_inb_prd
*           is_read_options = ls_read_options
            is_include_data = wa_include
          IMPORTING
            et_headers       = lt_headers
            et_items         = lt_items ).
      CATCH /scdl/cx_delivery.                          "#EC NO_HANDLER
    ENDTRY.

    LOOP AT lt_items INTO lwa_items.

      lwa_headers = VALUE #( lt_headers[ docid = lwa_items-docid ] OPTIONAL ). " Alternative READ T
      IF lwa_headers IS INITIAL.
        CONTINUE.
      ENDIF.
      lv_gr_status = VALUE #( lwa_headers-status[ status_type = /scdl/if_dl_status_c=>sc_t_goods_re
      lv_pw_status = VALUE #( lwa_headers-status[ status_type = /scdl/if_dl_status_c=>sc_t_putaway
      lv_pick_status = VALUE #( lwa_headers-status[ status_type = /scdl/if_dl_status_c=>sc_t_pickin
      lv_gi_status = VALUE #( lwa_headers-status[ status_type = /scdl/if_dl_status_c=>sc_t_goods_is

      IF lv_gr_status IS NOT INITIAL.
        SELECT SINGLE /scwm/door FROM /scdl/db_proci_i INTO @lv_door WHERE docid EQ @lwa_items-doci
      ELSE.
        SELECT SINGLE /scwm/door FROM /scdl/db_proci_o INTO @lv_door WHERE docid EQ @lwa_items-doci
      ENDIF.

      APPEND VALUE #(
      door = lv_door
      docno        = lwa_headers-docno
      refdocno       = VALUE #( lwa_headers-refdoc[ refdoccat = 'ERP' ]-refdocno OPTIONAL )
      status_gr    = SWITCH #( lv_gr_status
                          WHEN '0' THEN TEXT-002
                          WHEN '1' THEN TEXT-003
                          WHEN '2' THEN TEXT-004
                          WHEN '9' THEN TEXT-005 )

      status_putaway = SWITCH #( lv_pw_status
                          WHEN '0' THEN TEXT-002
                          WHEN '1' THEN TEXT-003
                          WHEN '2' THEN TEXT-004
                          WHEN '9' THEN TEXT-005 )

      status_picking    = SWITCH #( lv_pick_status
                          WHEN '0' THEN TEXT-002
                          WHEN '1' THEN TEXT-003
                          WHEN '2' THEN TEXT-004
                          WHEN '9' THEN TEXT-005 )

      status_gi = SWITCH #( lv_gi_status
                          WHEN '0' THEN TEXT-002
                          WHEN '1' THEN TEXT-003
                          WHEN '2' THEN TEXT-004
                          WHEN '9' THEN TEXT-005 ) ) TO et_data.
    ENDLOOP.

    DELETE et_data WHERE status_gi = TEXT-005. "gc_completed.
    DELETE et_data WHERE status_gr = TEXT-005. "gc_completed.

* Delete DN without door
```

```
    DELETE et_data WHERE door IS INITIAL.

* Delete not selected door
    IF NOT s_door[] IS INITIAL.
      DELETE  et_data WHERE NOT door IN s_door.
    ENDIF.

    IF NOT et_data[] IS INITIAL.

      SORT gt_door BY door.
      SORT et_data BY door.

* Format data for display
      LOOP AT et_data ASSIGNING FIELD-SYMBOL(<ls_data>) ##NEEDED.

        APPEND INITIAL LINE TO gt_table ASSIGNING FIELD-SYMBOL(<ls_table>).
        <ls_table>-door = <ls_data>-door.
        <ls_table>-refdocno = <ls_data>-refdocno.
        <ls_table>-status_gr = <ls_data>-status_gr.
        <ls_table>-status_putaway = <ls_data>-status_putaway.
        <ls_table>-status_picking = <ls_data>-status_picking.
        <ls_table>-status_gi = <ls_data>-status_gi.

        CALL FUNCTION 'CONVERSION_EXIT_ALPHA_OUTPUT'
          EXPORTING
            input  = <ls_data>-docno
          IMPORTING
            output = <ls_table>-docno.

*     APPEND gs_table TO gt_table.

        AT NEW door.
          READ TABLE gt_door INTO DATA(ls_door) WITH KEY door = <ls_table>-door BINARY SEARCH ##NEE
          IF sy-subrc = 0.
            DELETE gt_door INDEX sy-tabix.
          ENDIF.
        ENDAT.

      ENDLOOP.

* add doors to output list which dont have OBD
      IF s_docno[] IS INITIAL.       " if DN is selected then no need to display all door list
        LOOP AT gt_door ASSIGNING FIELD-SYMBOL(<ls_door>).
          APPEND INITIAL LINE TO gt_table ASSIGNING FIELD-SYMBOL(<ls_table_blank_door>).
          <ls_table_blank_door>-door = <ls_door>-door.
        ENDLOOP.
      ENDIF.

* display empty doors on top of list
      SORT gt_table BY door docno.

      LOOP AT gt_table ASSIGNING FIELD-SYMBOL(<fs_table>).
        IF <fs_table>-docno IS NOT INITIAL.
          IF <fs_table>-refdocno IS INITIAL.
            DELETE gt_table INDEX sy-tabix.
          ENDIF.
        ENDIF.
      ENDLOOP.

      IF NOT s_door[] IS INITIAL.
        DELETE  gt_table WHERE NOT door IN s_door.
      ENDIF.

    ENDIF.

    CLEAR et_data.
    APPEND LINES OF gt_table TO et_data.

  ENDIF.

  LOOP AT ct_fieldcat ASSIGNING FIELD-SYMBOL(<fs_fca>).
    CASE  <fs_fca>-fieldname.
      WHEN 'DOCNO'.
        <fs_fca>-reptext = TEXT-006.
        <fs_fca>-scrtext_l = TEXT-007.
        <fs_fca>-scrtext_m = TEXT-007.
        <fs_fca>-scrtext_s = TEXT-007.
      WHEN 'REFDOCNO'.
        <fs_fca>-reptext = TEXT-008.
        <fs_fca>-scrtext_l = TEXT-009.
        <fs_fca>-scrtext_m = TEXT-009.
        <fs_fca>-scrtext_s = TEXT-009.
      WHEN 'DOOR'.
        <fs_fca>-reptext = TEXT-010.
        <fs_fca>-scrtext_l = TEXT-011.
        <fs_fca>-scrtext_m = TEXT-011.
        <fs_fca>-scrtext_s = TEXT-011.
      WHEN 'STATUS_GR'.
        <fs_fca>-reptext = TEXT-012.
        <fs_fca>-scrtext_l = TEXT-012.
        <fs_fca>-scrtext_m = TEXT-012.
        <fs_fca>-scrtext_s = TEXT-012.
      WHEN 'STATUS_PUTAWAY'.
        <fs_fca>-reptext = TEXT-013.
        <fs_fca>-scrtext_l = TEXT-013.
        <fs_fca>-scrtext_m = TEXT-013.
        <fs_fca>-scrtext_s = TEXT-013.
      WHEN 'STATUS_PICKING'.
        <fs_fca>-reptext = TEXT-014.
        <fs_fca>-scrtext_l = TEXT-014.
        <fs_fca>-scrtext_m = TEXT-014.
        <fs_fca>-scrtext_s = TEXT-014.
      WHEN 'STATUS_GI'.
        <fs_fca>-reptext = TEXT-015.
        <fs_fca>-scrtext_l = TEXT-015.
        <fs_fca>-scrtext_m = TEXT-015.
        <fs_fca>-scrtext_s = TEXT-015.
    ENDCASE.
  ENDLOOP.

ENDFUNCTION.
```