**GROUP 1**

**CaseStudy Topic** : Finding Workorder from Notification Status and Send via Email Based on Order Type

**Developers:** Dhanush Kaduluri – 280403

Rashik Shaik – 280402

Nidhisha Sonaji - 280408

**Tables used:**

1)ZG01\_AFKO

2)ZG01\_AFVC

3)ZG01\_AUFK

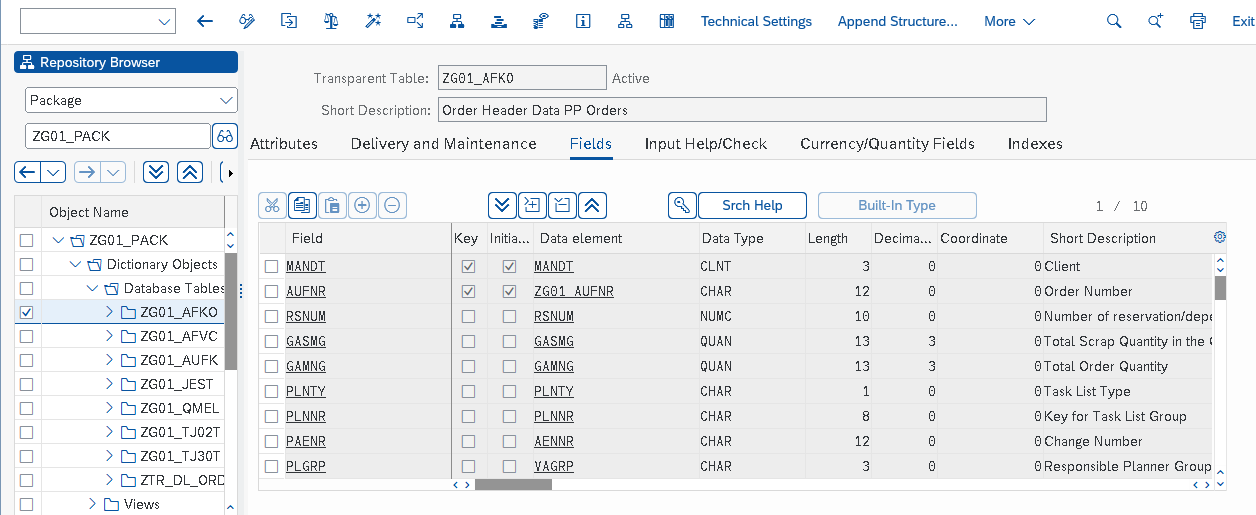
4)ZG01\_JEST

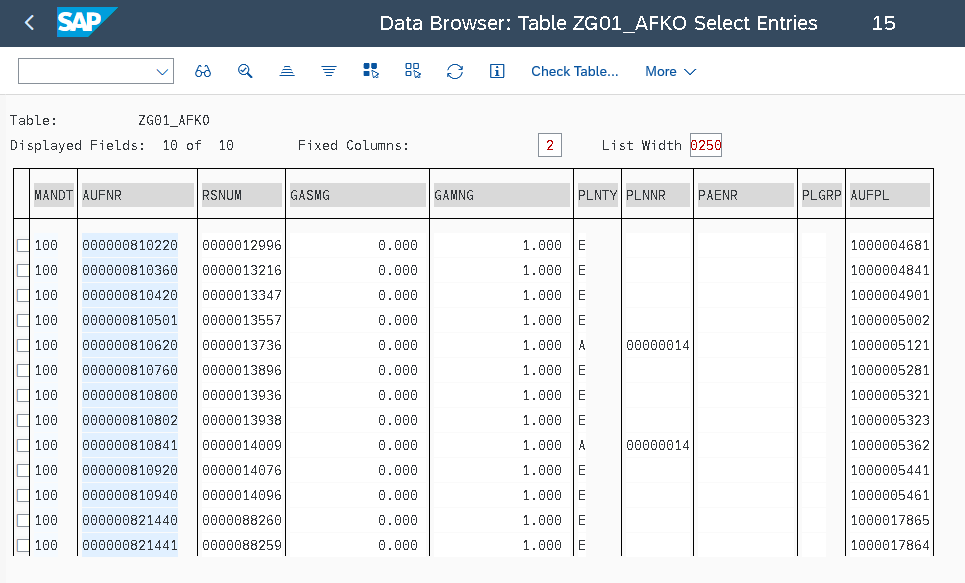
5)ZG01\_QMEL

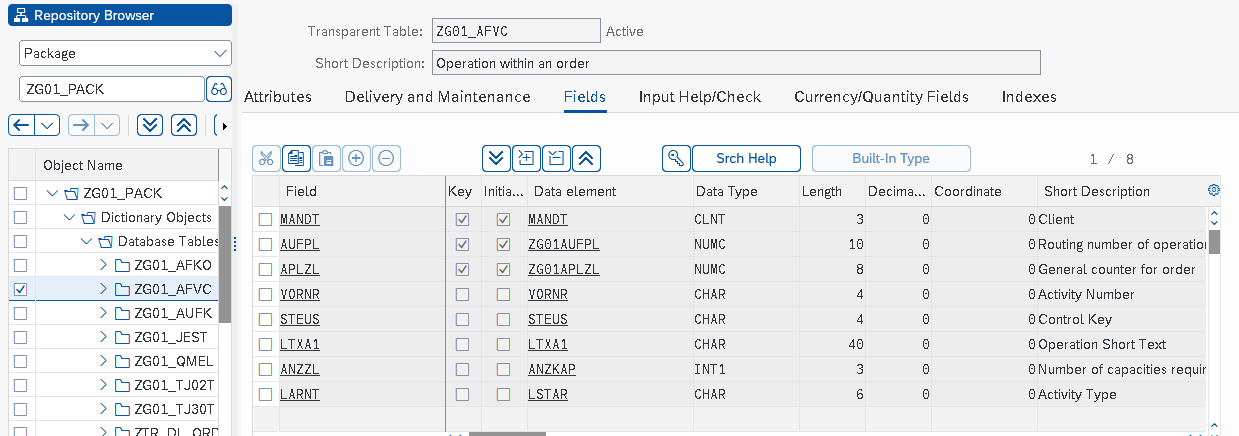
6)ZG01\_TJ02T

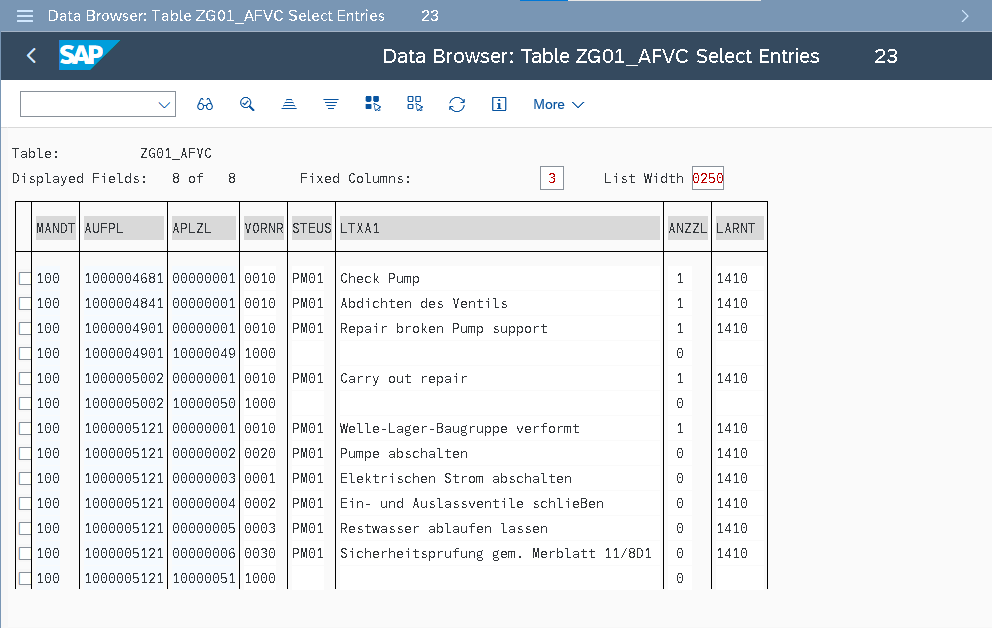
7)ZG01\_TJ30T

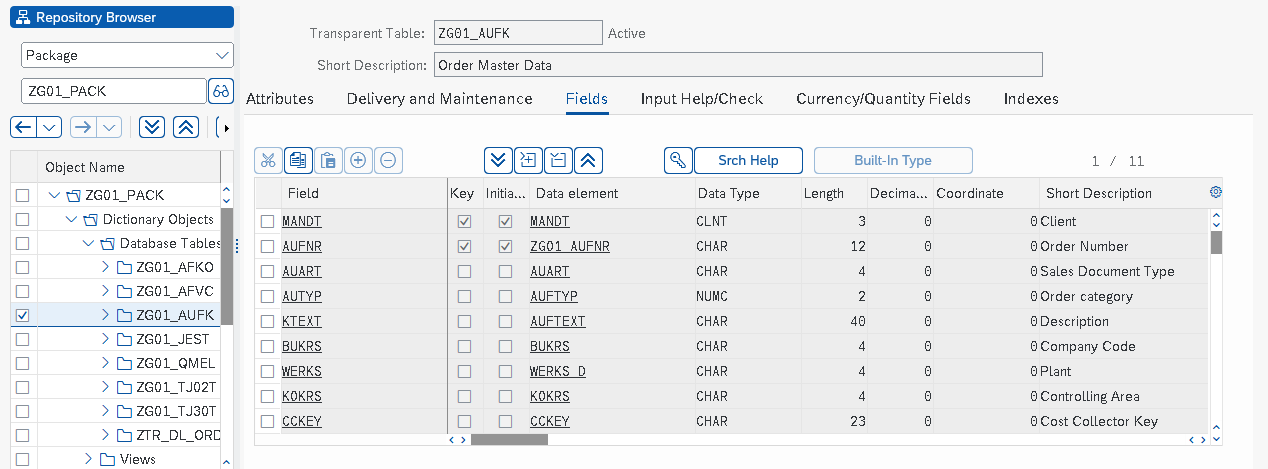
8)ZTR\_DL\_ORD\_TYPE

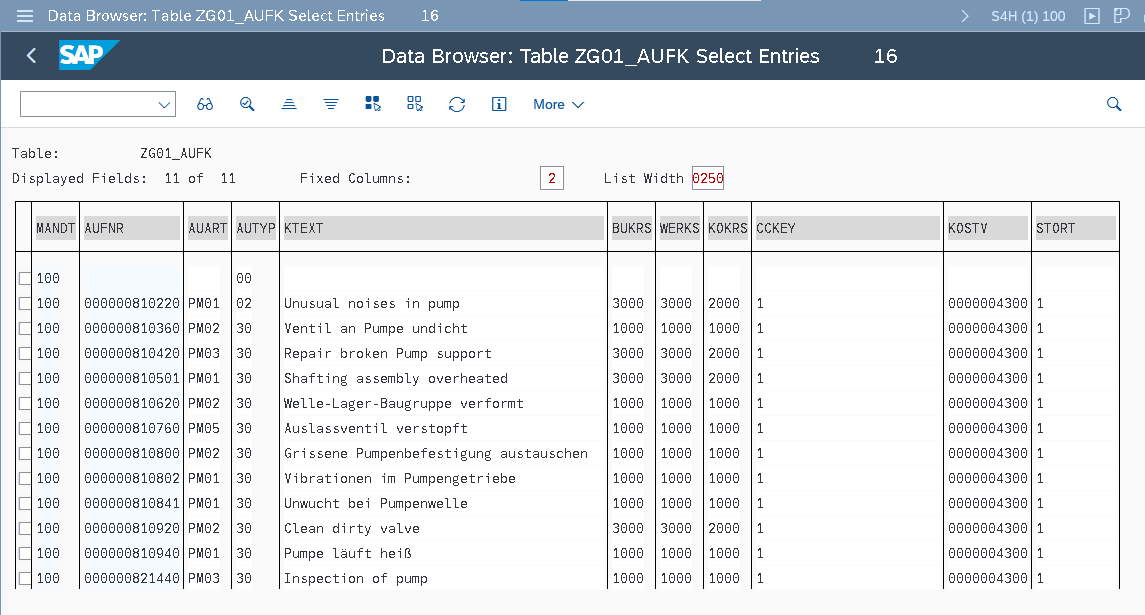


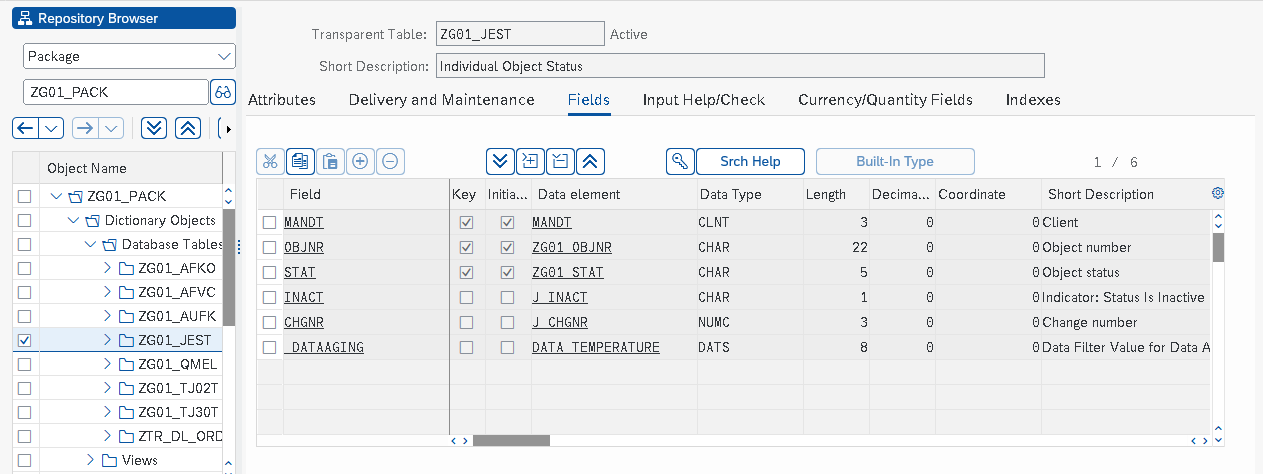


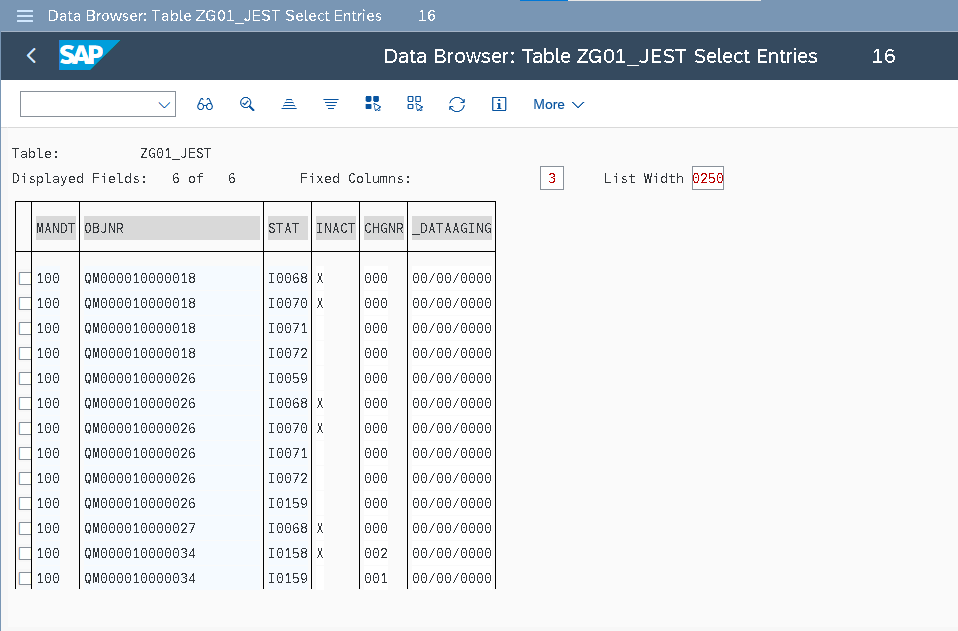


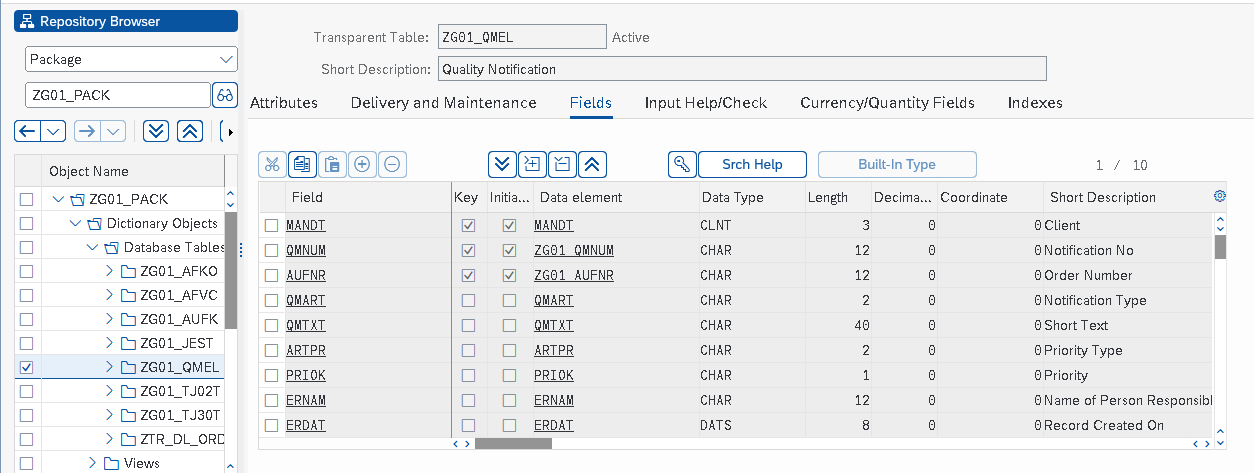


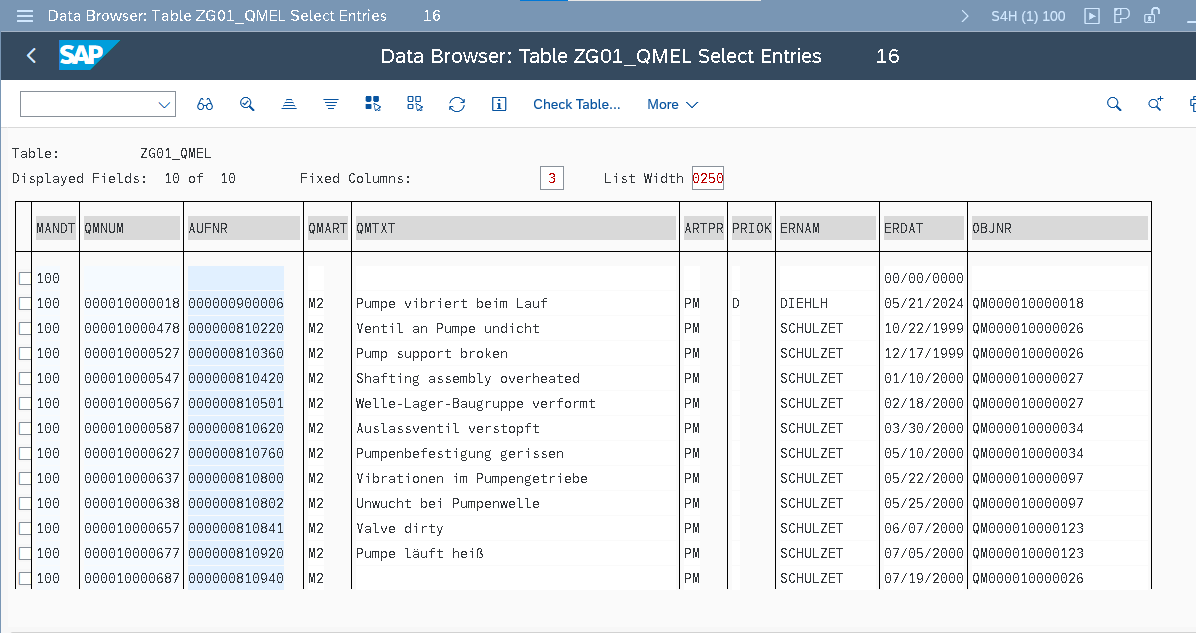


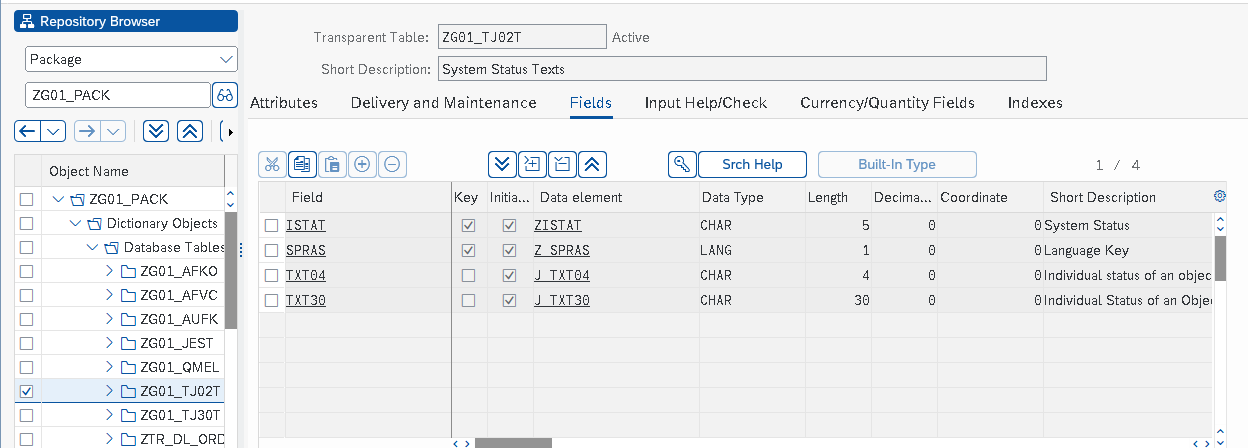


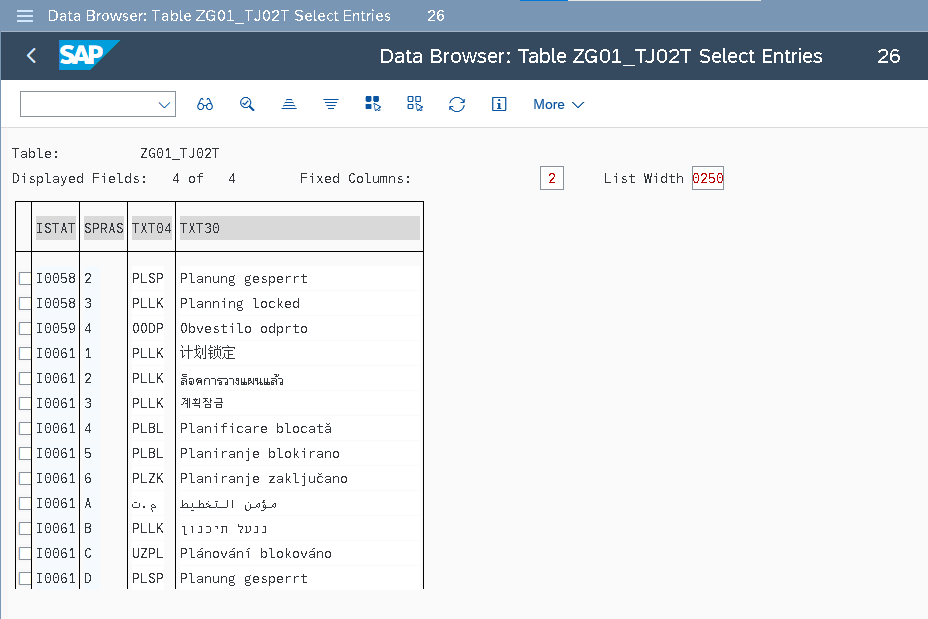


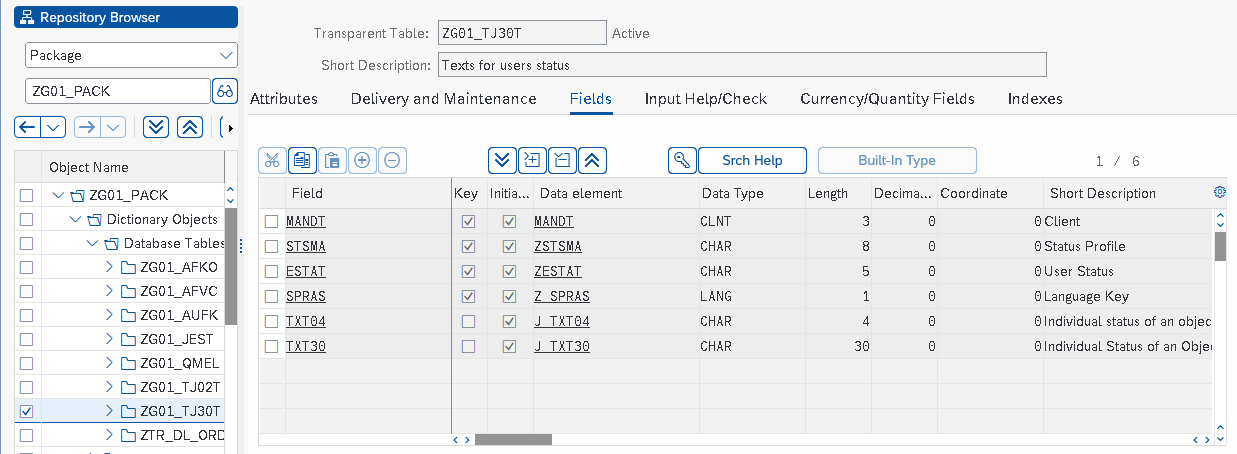


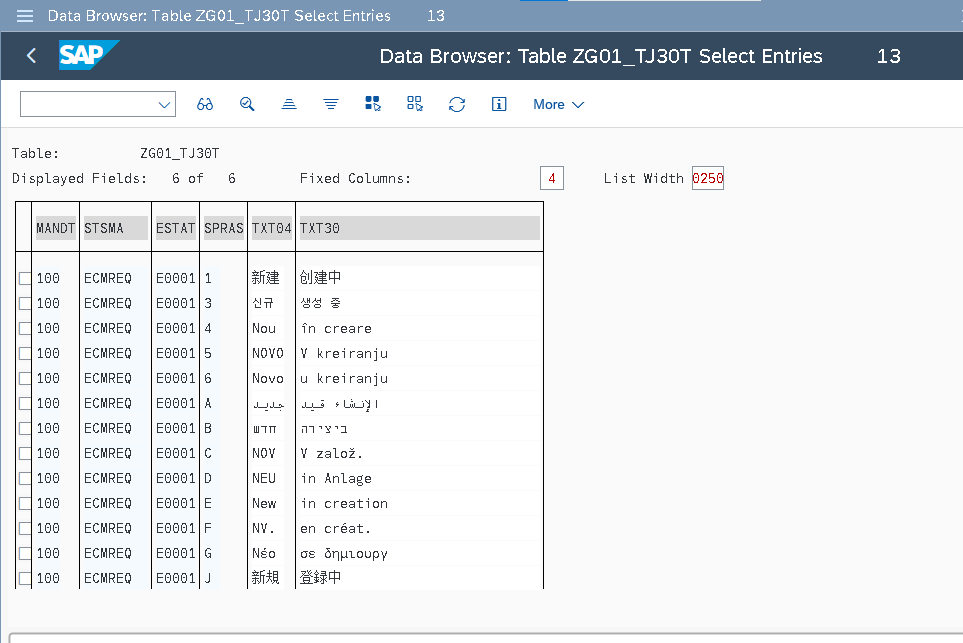


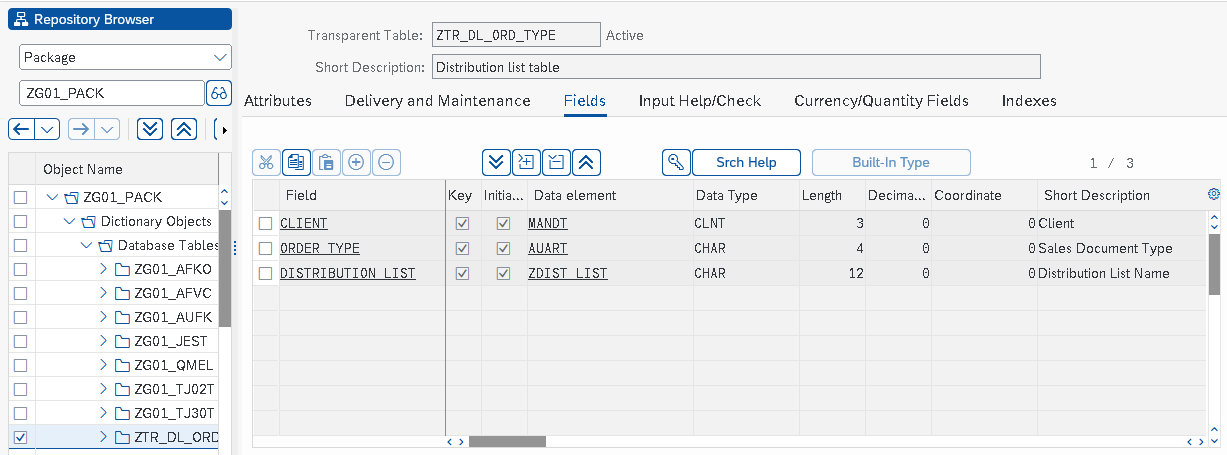


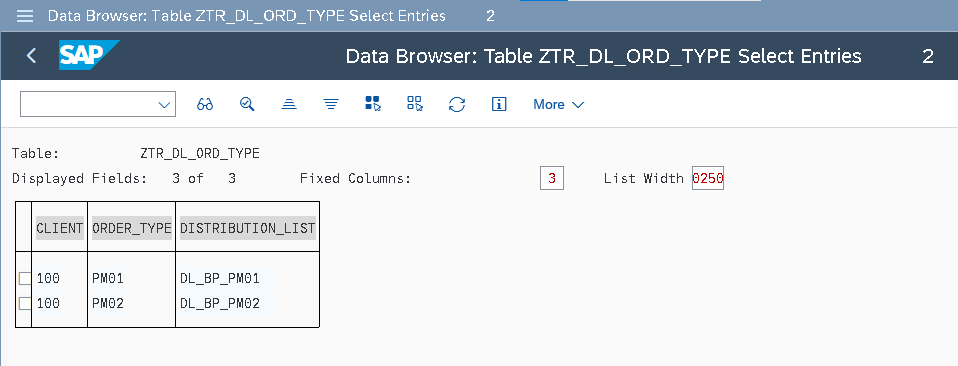






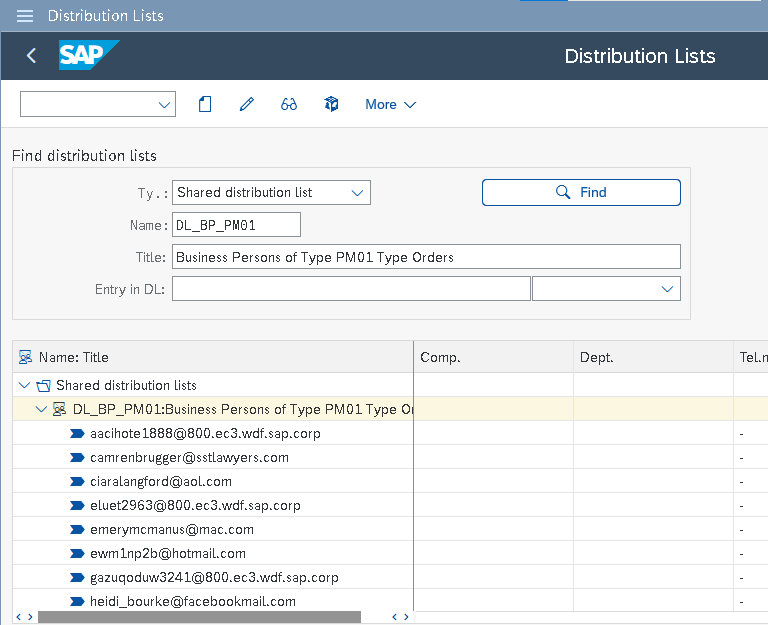




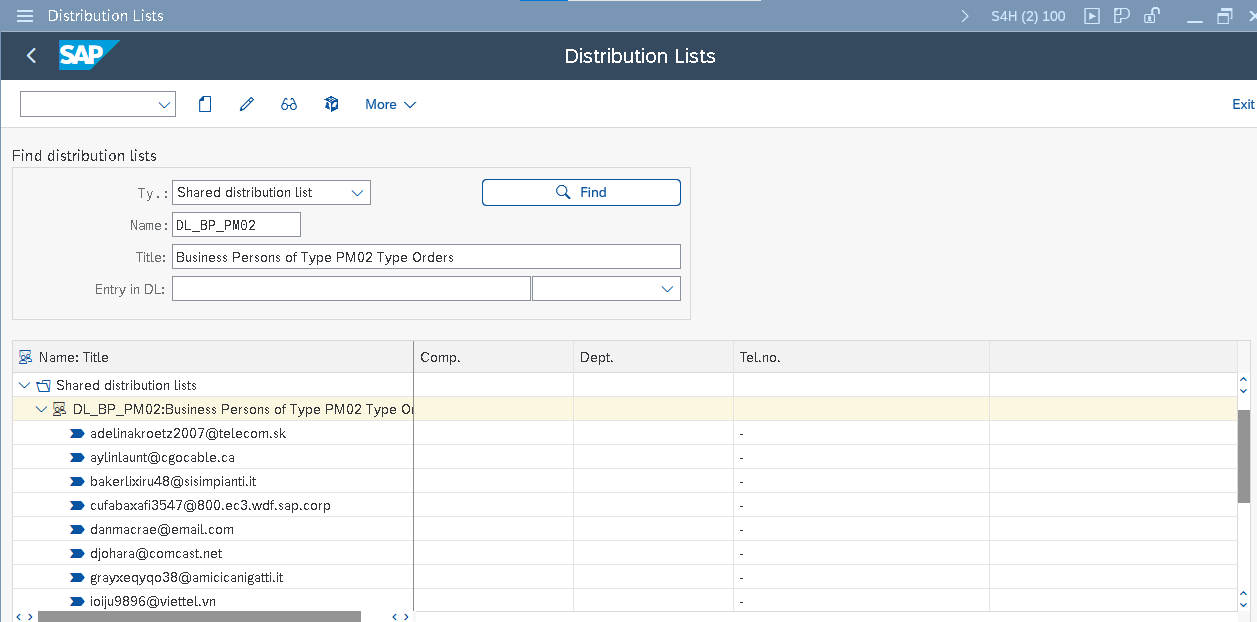


**Distribution Lists**

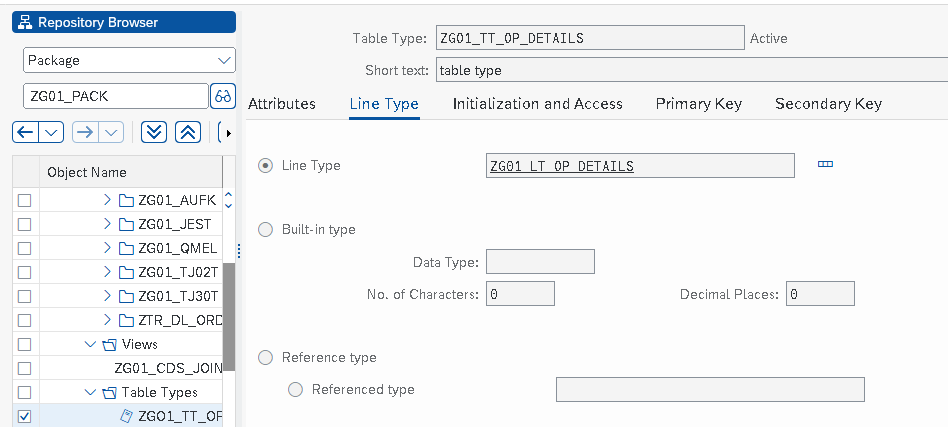
**1)DL\_BP\_PM01**

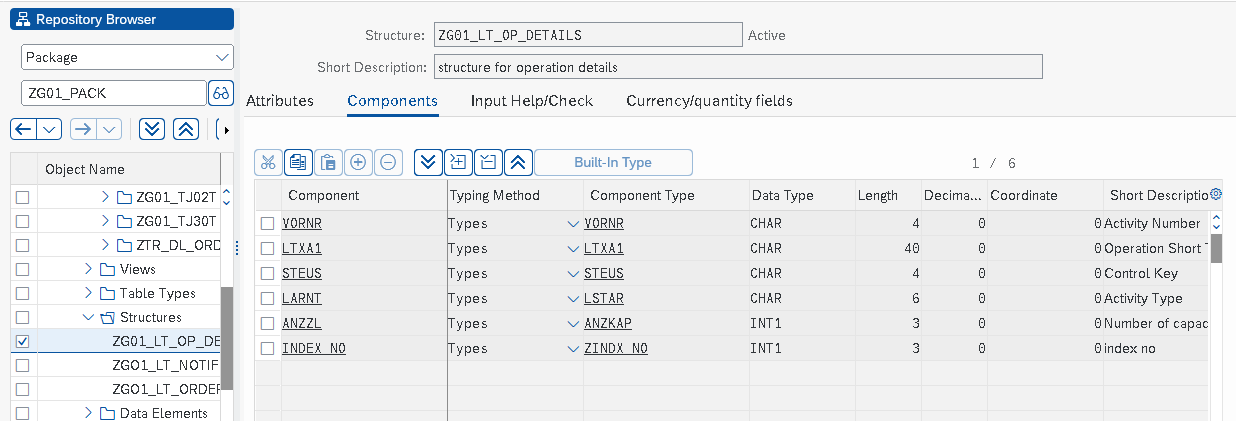


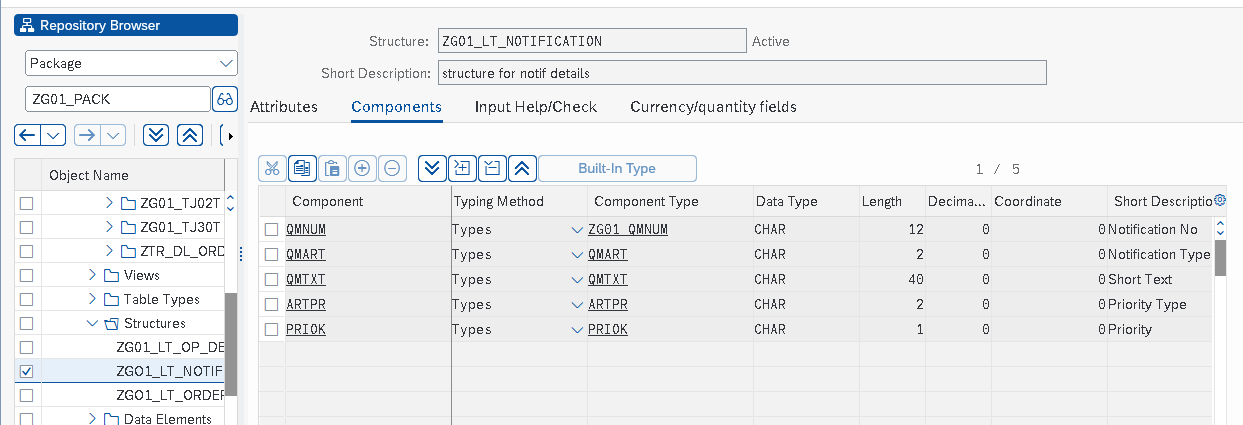
**2)DL\_BP\_PM02**

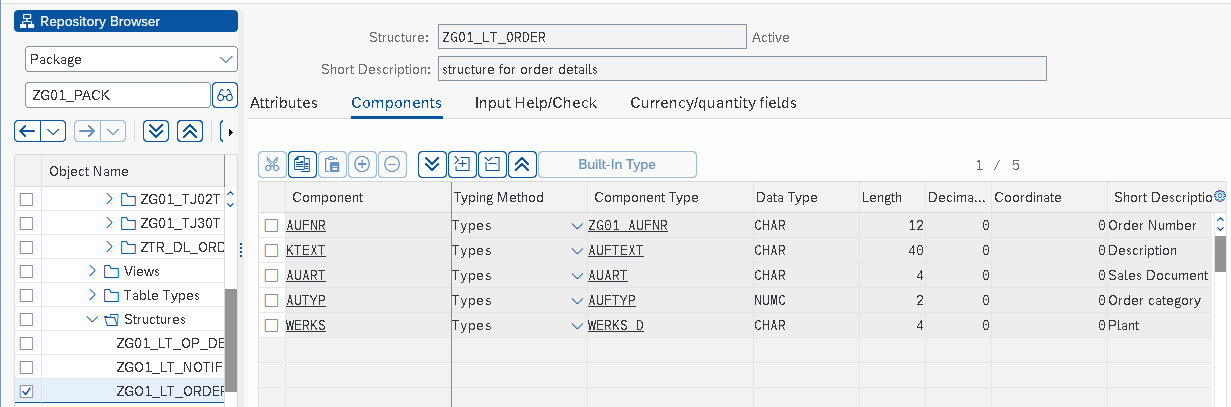


**TABLE TYPES AND LINE TYPES**

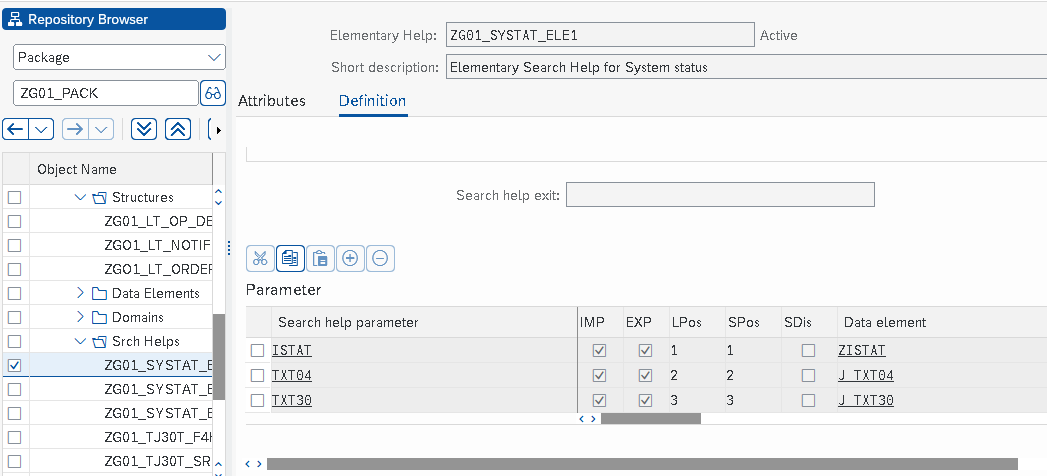


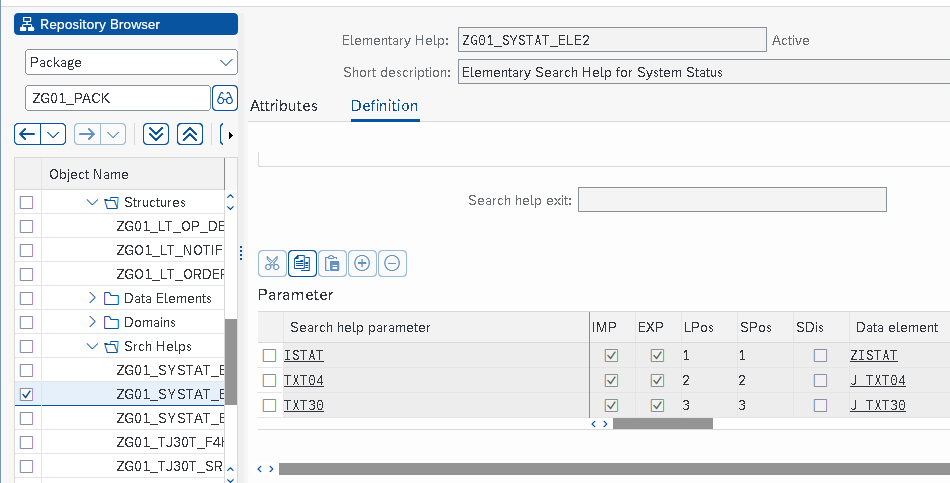


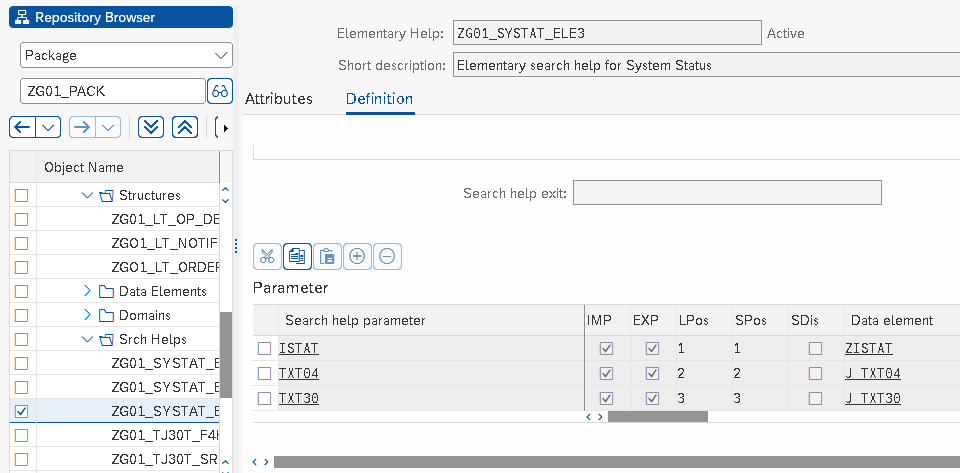


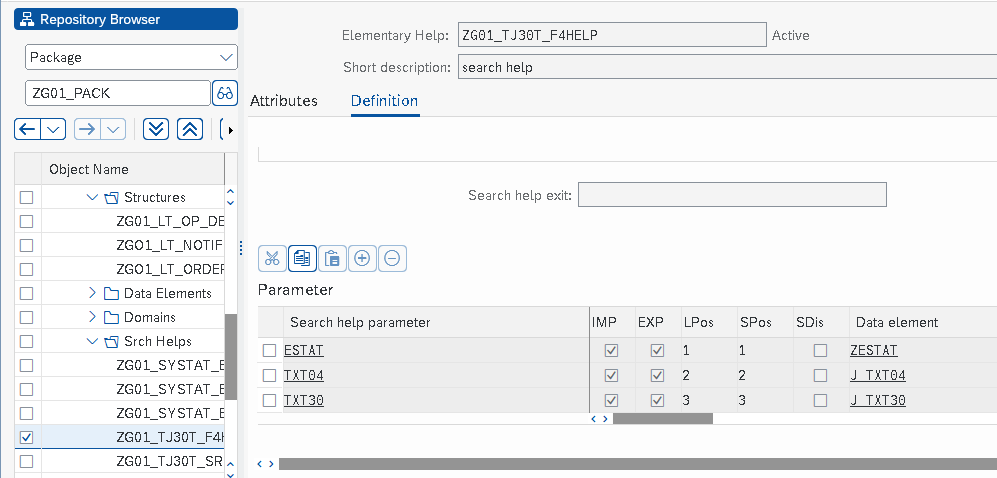


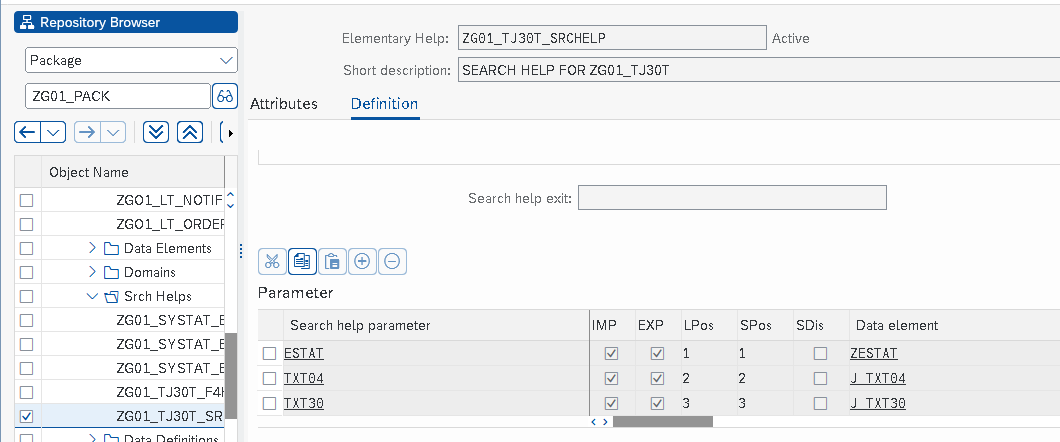
**SEARCH HELPS**



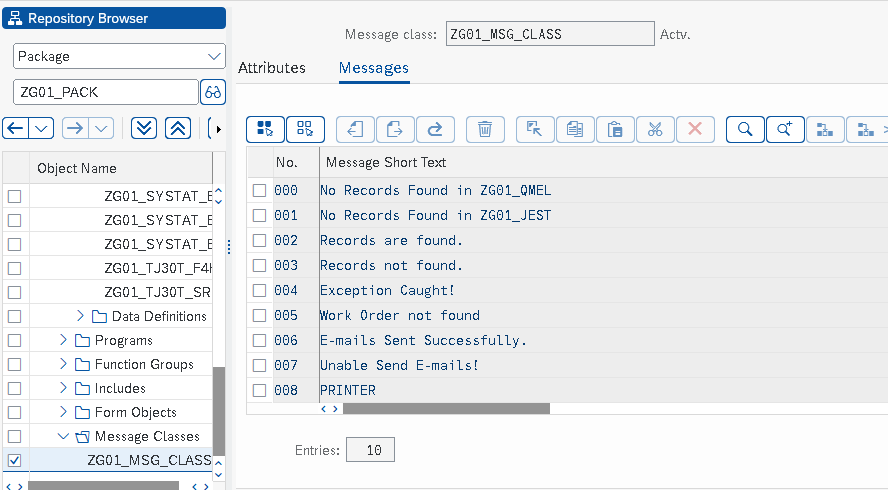








**Message Class**



**REPORT NAME : ZG01\_WORKORDER\_STATUS**

\*&---------------------------------------------------------------------\*  
\*& Report ZG01\_WORKORDER\_STATUS  
\*&---------------------------------------------------------------------\*  
\*&  Developer Names : Dhanush Kaduluri,  
\*&                    Nidhisha Sonaji,  
\*&                    Rashik Shaik.  
\*&  
\*&  Object Description : Finding WorkOrder Notification Status and  
\*&                       Send via Email Based on Order Type.  
\*&  
\*&  
\*&  FS Object Id : GROUP1  
\*&  
\*&  Object Creation : 22.04.2024  
\*&--------------------------------------------------------------------------\*  
REPORT ZG01\_WORKORDER\_STATUS NO STANDARD PAGE HEADING LINE-COUNT 30(3)  
                             LINE-SIZE 300 MESSAGE-ID ZG01\_MSG\_CLASS.  
  
\*&--------------------------------------------------------------------------\*  
\*&              T O P  I N C L U D E  D E C L A R A T I O N S  
\*&--------------------------------------------------------------------------\*  
Include ZWorkorder\_top\_inc.  
  
\*&--------------------------------------------------------------------------\*  
\*&  CLASS FOR DOUBLE CLICK EVENT AND GENERATING AN ADOBE FORM ON DOUBLE CLICK  
\*&--------------------------------------------------------------------------\*  
include zg01\_alv\_class.  
  
\*&--------------------------------------------------------------------------\*  
\*&                      S E L E C T I O N S C R E E N  
\*&--------------------------------------------------------------------------\*  
Include ZUser\_Input\_Status.  
  
  
\*&--------------------------------------------------------------------------\*  
\*&                   M A I N  B U S I N E S S  L O G I C  
\*&--------------------------------------------------------------------------\*  
Include Zworkorder\_Mainlogic\_code.  
  
\*&--------------------------------------------------------------------------\*  
\*&         S E N D I N G    E M A I L   N O T I F I C A T I O N  
\*&--------------------------------------------------------------------------\*  
Include Zworkorder\_send\_notification.  
  
\*&--------------------------------------------------------------------------\*  
\*&                          A D O B E  F O R M  
\*&--------------------------------------------------------------------------\*  
Include Zworkorder\_adobe\_form.  
  
\*&--------------------------------------------------------------------------\*  
\*& BASED ON THE USER SELECTION FETCHING DATA AND SENDING NOTIFICATION STATUS  
\*&--------------------------------------------------------------------------\*  
Include Zworkorder\_perform.  
  
\*&--------------------------------------------------------------------------\*  
\*&             O B J E C T   O R I E N T E D   A L V  
\*&--------------------------------------------------------------------------\*  
Include ZG01\_WORKORDER\_OOALV.

**INCLUDES**

\*&---------------------------------------------------------------------\*  
\*& Include          ZWORKORDER\_TOP\_INC  
\*&---------------------------------------------------------------------\*  
  
TYPE-POOLS: vrm. "Uses Type pool vrm for dropdown at selection-screen  
  
TYPES : BEGIN OF ls\_struct,       " LOCAL STRUCTURE FOR OBJ NUM  
          objnr TYPE ZG01\_objnr,  
        END OF ls\_struct,  
  
        BEGIN OF ls\_qmel,  
          qmnum TYPE zg01\_qmnum, " Notification Number  
          aufnr TYPE zg01\_aufnr, " Order Number  
        END OF ls\_qmel,  
  
        BEGIN OF combo\_struct,  
          "ZG01\_AUFK TABLE  
          aufnr TYPE zg01\_AUFNR, " Order Number  
          ktext TYPE ktext, " Description  
          auart TYPE zg01\_AUART, " Sales Document No  
          autyp TYPE AUfTYP, " Order Type  
          stort TYPE stort,  " Asset Location  
  
          aufpl TYPE co\_aufpl, " Routing number of operations in the order  
  
          vornr TYPE vornr, "Activity Number  
  
          steus TYPE steus, "Control Key  
          ltxa1 TYPE ltxa1, " Operation Short Text  
          larnt TYPE lstar, " Activity Type  
          anzzl TYPE anzzl, " Number of capacities required  
        END OF combo\_struct.  
  
  
DATA : it\_obj\_stat TYPE STANDARD TABLE OF ls\_struct,   "INTERNAL TAB FOR OBJ NUM IN STATUS  
       wa\_obj\_stat TYPE ls\_struct,                     "WORK AREA FOR OBJ NUM IN STATUS  
  
       it\_qmel     TYPE STANDARD TABLE OF ls\_qmel,  
       wa\_qmel     TYPE ls\_qmel,  
  
       it\_combo    TYPE STANDARD TABLE OF combo\_struct,  
       wa\_combo    TYPE combo\_struct,  
  
       it\_pm       TYPE STANDARD TABLE OF combo\_struct,  
       wa\_pm       TYPE combo\_struct,  
  
       it\_pmx      TYPE STANDARD TABLE OF combo\_struct,  
       wa\_pmx      TYPE combo\_struct,  
  
       wa\_fcat2    TYPE lvc\_s\_fcat,   "WORKAREA LIKE OOALV FCAT STRUCTURE  
       lt\_fcat2    TYPE lvc\_t\_fcat,   "INTERNAL TABLE LIKE OOALV FCAT TABLE  
       LS\_layout   TYPE lvc\_s\_layo,   "LAYOUT LIKE OOALV FCAT LAYOUT  
  
       i\_container TYPE REF TO cl\_gui\_custom\_container,  " Container for storing data.  
       i\_alvgrid   TYPE REF TO cl\_gui\_alv\_grid, " Grid display for output.  
  
       name        TYPE vrm\_id,  
       list        TYPE vrm\_values,   "To store Dropdown options  
       value       LIKE LINE OF list, "Status type are assigned to values.  
       lv\_status   TYPE zestat.       "Based on the user selection status type is updated.  
  
  
\*Defines a selection screen block named "b1" with a title  
SELECTION-SCREEN BEGIN OF BLOCK b1 WITH FRAME TITLE TEXT-007.  
\*list box with a default value of '1' and allows users to select a value from a dropdown list.  
  PARAMETERS: ps\_parm(10) AS LISTBOX VISIBLE LENGTH 10 DEFAULT '1'.  
\*parameters are status type used to identify the input fields on the selection-screen.  
  PARAMETERS : stm\_stat TYPE zistat MODIF ID d1, "  
               usr\_stat TYPE zestat MODIF ID d2.  
SELECTION-SCREEN END OF BLOCK b1.

\*&---------------------------------------------------------------------\*  
\*& Include          ZG01\_ALV\_CLASS  
\*&---------------------------------------------------------------------\*  
CLASS lcl\_alv\_inter DEFINITION.  
  Public SECTION.  
\* Method is triggered when the user double-clicks on a row in an ALV grid control.  
\*It takes the index of the double-clicked row (e\_row) as input.  
    METHODS : handle\_double\_click FOR EVENT double\_click OF cl\_gui\_alv\_grid  IMPORTING e\_row.  
ENDCLASS.  
  
CLASS lcl\_alv\_inter IMPLEMENTATION.  
  METHOD  handle\_double\_click.  
\*read a row from an internal table named it\_pmx into a work area named wa\_pmx  
    READ TABLE it\_pmx INTO wa\_pmx INDEX e\_row-index."e\_row-index which represents clicked row index  
    IF wa\_pmx IS INITIAL.  
      MESSAGE I005. " If no data is found (initial), message no data is displayed.  
    ELSE.  
\* Subroutine is called i.e creates a new form and passes the order number(aufnr) from the work area wa\_pmx as input  
      PERFORM zf\_create\_form USING wa\_pmx-aufnr.  
    ENDIF.  
  ENDMETHOD.  
ENDCLASS.

\*&---------------------------------------------------------------------\*  
\*& Include          ZUSER\_INPUT\_STATUS  
\*&---------------------------------------------------------------------\*  
  
at SELECTION-SCREEN OUTPUT. "defines an even that gets triggered after the selection screen  
  
LOOP AT SCREEN. "starts a loop that iterates over all the elements on the selection screen.  
    if PS\_PARM = TEXT-001 and screen-group1 = 'D1'. " PS\_PARM = '1' and screen-group1 = 'D2  
      screen-active = 0. "makes element inactive on the screen, user cannot interact with it.  
    ELSEIF PS\_PARM = TEXT-003 and screen-group1 = 'D2'. " PS\_PARM = '2' and screen-group1 = 'D2  
      screen-active = 0.  
    ENDIF.  
    MODIFY SCREEN."applies the changes made to the screen element's screen-active attribute within the loop.  
ENDLOOP.  
  
INITIALIZATION.  
  
\*DropDown List for Selection of Status Type  
VALUE-KEY = text-001.  "1  
VALUE-TEXT = text-002. "User Status  
APPEND VALUE TO LIST.  "adds the dropdown selection inputs to an internal table named LIST.  
  
VALUE-KEY = text-003.  "2  
VALUE-TEXT = text-004. "System Status  
APPEND VALUE TO LIST.  
  
CALL FUNCTION 'VRM\_SET\_VALUES'  " Function Module  
  EXPORTING  
    ID = text-000  " Parameters of Status type for the user selection.  
    VALUES = LIST.

\*&---------------------------------------------------------------------\*  
\*& Include          ZWORKORDER\_MAINLOGIC\_CODE  
\*&---------------------------------------------------------------------\*  
  
\*&---------------------------------------------------------------------\*  
\*& Form get\_data  
\*&---------------------------------------------------------------------\*  
  
FORM zf\_get\_data .  
\*selects the field objnr from the table zg01\_jest store it in internal table  
  SELECT objnr INTO TABLE it\_obj\_stat  
    FROM zg01\_jest  
\*filters the data based on User selected status type from the dropdown  
    WHERE stat EQ lv\_status AND inact EQ ' '.  
  
  IF it\_obj\_stat IS NOT INITIAL. " checks if the table is not empty.  
  
    SELECT qmnum aufnr INTO TABLE it\_qmel  
    FROM zg01\_qmel  
    FOR ALL ENTRIES IN it\_obj\_stat  
\*filters the data where objnr from zg01\_qmel matches the objnr from the current entry in it\_obj\_stat  
    WHERE objnr EQ it\_obj\_stat-objnr.  
  
    IF it\_qmel IS NOT INITIAL. " checks if the table is not empty.  
      PERFORM zf\_table\_joins. "performs joins on additional tables based on the notification numbers.  
      LOOP AT it\_combo INTO wa\_combo. " loop iterates through each entry in the it\_combo table  
  
\*condition checks if the field auart in (wa\_combo) is not equal to either 'PM01' or 'PM02'  
        IF wa\_combo-auart NE TEXT-037 AND wa\_combo-auart NE TEXT-038. " wa\_combo-auart NE 'PM01' AND wa\_combo-auart NE 'PM02'  
          MOVE wa\_combo TO wa\_pmx. "Move data from wa\_combo to wa\_pmx excluding pm01 and pm02.  
          APPEND wa\_pmx TO it\_pmx.  
          CLEAR wa\_pmx.  
        ENDIF.  
        CLEAR wa\_combo.  
      ENDLOOP.  
    ELSE.  
      MESSAGE e000.  
    ENDIF.  
  ELSE.  
    MESSAGE e001.  
  ENDIF.  
ENDFORM.  
  
  
\*&---------------------------------------------------------------------\*  
\*& Form JOINS  
\*&---------------------------------------------------------------------\*  
  
FORM zf\_table\_joins.  
  
\*\*\* SQL QUERY - INNERJOINS  
\*  
  SELECT         " selects required fields from three tables to perform joins  
      aufk~aufnr  
      aufk~ktext  
      aufk~auart  
      aufk~autyp  
      aufk~stort  
      afko~aufpl  
      afvc~vornr  
      afvc~steus  
      afvc~ltxa1  
      afvc~larnt  
      afvc~anzzl  
  INTO TABLE it\_combo  " Stores into internal table  
  FROM zg01\_aufk AS aufk " table zg01\_aufk aliased as aufk  
\*inner join connects ZG01\_AUFK(aliased as aufk) and ZG01\_AFKO(aliased as afko) based on (aufnr).  
  INNER JOIN zg01\_afko AS afko ON aufk~aufnr = afko~aufnr  
\* inner join connects previously joined table(effectively ZG01\_AUFK and ZG01\_AFKO combined)  
\* with ZG01\_AFVC(aliased as afvc) based on (aufpl)  
  INNER JOIN zg01\_afvc AS afvc ON afko~aufpl = afvc~aufpl  
  FOR ALL ENTRIES IN it\_qmel " iterates through all entries in the internal table it\_qmel  
\*filters selected data based on the order number(aufnr) from table(aufk) matches the order number in the current entry from it\_qmel  
  WHERE aufk~aufnr = it\_qmel-aufnr.  
  IF sy-subrc IS not INITIAL.  
    MESSAGE e003.  
  ENDIF.  
ENDFORM.

\*&---------------------------------------------------------------------\*  
\*& Include          ZWORKORDER\_SEND\_NOTIFICATION  
\*&---------------------------------------------------------------------\*  
  
FORM zf\_send\_email\_notification USING receipents TYPE any.  
  
  DATA: lt\_mailrecipients TYPE STANDARD TABLE OF somlrec90 WITH HEADER LINE,  
        lt\_mailtxt        TYPE STANDARD TABLE OF soli      WITH HEADER LINE,  
        lt\_attachment     TYPE STANDARD TABLE OF solisti1  WITH HEADER LINE,  
        lt\_mailsubject    TYPE sodocchgi1,  
        lt\_packing\_list   TYPE STANDARD TABLE OF sopcklsti1 WITH HEADER LINE,  
        subject           TYPE string,  
        gv\_cnt            TYPE i.  
  
\*Now, lets start formatting the mail.  
  
\*Add Recipients  
  
  lt\_mailrecipients-rec\_type  = 'C'.  "shared distribution list.  
  lt\_mailrecipients-com\_type  = 'INT'.  
  lt\_mailrecipients-receiver  = receipents.  
  APPEND lt\_mailrecipients .  
  CLEAR lt\_mailrecipients .  
  
\*Put in the Mail Contents  
  
  lt\_mailtxt = text-039.  "'Hello,'.  
  APPEND lt\_mailtxt. CLEAR lt\_mailtxt.  
  lt\_mailtxt = ' '.  
  APPEND lt\_mailtxt. CLEAR lt\_mailtxt.  
  lt\_mailtxt = text-040. "'Attached is the list of WorkOrder with status Notification in process.'.  
  APPEND lt\_mailtxt. CLEAR lt\_mailtxt.  
  lt\_mailtxt = text-041. "' Kindly take required action to complete the work within timeline.'.  
  APPEND lt\_mailtxt. CLEAR lt\_mailtxt.  
  lt\_mailtxt = '  '.  
  APPEND lt\_mailtxt. CLEAR lt\_mailtxt.  
  lt\_mailtxt = text-042. "'Best Regards,'.  
  APPEND lt\_mailtxt. CLEAR lt\_mailtxt.  
  lt\_mailtxt = text-043. " 'Group 1'.  
  APPEND lt\_mailtxt. CLEAR lt\_mailtxt.  
  lt\_mailtxt = '  '.  
  APPEND lt\_mailtxt. CLEAR lt\_mailtxt.  
  lt\_mailtxt = text-044. " 'Note: This is system generated email, do not reply to this email.'.  
  APPEND lt\_mailtxt. CLEAR lt\_mailtxt.  
  
  DATA ord\_type TYPE auart.  
  IF receipents EQ text-005. "'DL\_BP\_PM01'.  
    ord\_type = text-037. " 'PM01'.  
    subject = text-045. " 'Action required on WorkOrder of Type - PM01'.  
  ELSE.  
    ord\_type = text-038. " 'PM02'.  
    subject = text-046. " 'Action required on WorkOrder of Type - PM02'.  
  ENDIF.  
  
  LOOP AT it\_combo INTO wa\_combo.  
  
    IF WA\_combo-auart EQ ord\_type.  
      MOVE wa\_combo TO wa\_pm.  
      APPEND wa\_pm TO it\_pm.  
      CLEAR wa\_pm.  
    ENDIF.  
    CLEAR wa\_combo.  
  
  ENDLOOP.  
  
  CLASS cl\_abap\_char\_utilities DEFINITION LOAD.  
  CONCATENATE Text-047 text-048 text-049 text-050 text-051 text-052 text-053 text-054 text-055 text-056  
              INTO lt\_attachment SEPARATED BY  
              cl\_abap\_char\_utilities=>horizontal\_tab.  
  APPEND lt\_attachment. CLEAR lt\_attachment.  
  
  LOOP AT it\_pm INTO wa\_pm.  
    CONCATENATE wa\_pm-aufnr wa\_pm-ktext wa\_pm-auart wa\_pm-autyp wa\_pm-stort wa\_pm-aufpl wa\_pm-vornr wa\_pm-steus wa\_pm-ltxa1 wa\_pm-larnt "wa\_pm-anzzl  
                INTO lt\_attachment SEPARATED BY  
                cl\_abap\_char\_utilities=>horizontal\_tab.  
  
    CONCATENATE cl\_abap\_char\_utilities=>newline lt\_attachment  
                INTO lt\_attachment.  
  
    APPEND lt\_attachment. CLEAR lt\_attachment.  
  ENDLOOP.  
  
\*Pack the mail contents and attachment  
  
  lt\_packing\_list-transf\_bin  = space.  
  lt\_packing\_list-head\_start  = 1.  
  lt\_packing\_list-head\_num    = 0.  
  lt\_packing\_list-body\_start  = 1.  
  lt\_packing\_list-body\_num    = lines( lt\_mailtxt ).  
  lt\_packing\_list-doc\_type    = 'RAW'.  
  APPEND lt\_packing\_list. CLEAR lt\_packing\_list.  
  
  lt\_packing\_list-transf\_bin  = 'X'.  
  lt\_packing\_list-head\_start  = 1.  
  lt\_packing\_list-head\_num    = 1.  
  lt\_packing\_list-body\_start  = 1.  
  lt\_packing\_list-body\_num    = lines( lt\_attachment ).  
  lt\_packing\_list-doc\_type    = 'XLS'. " You can give RAW incase if you want just a txt file.  
  lt\_packing\_list-obj\_name    = 'data.xls'.  
  lt\_packing\_list-obj\_descr   = 'data.xls'.  
  lt\_packing\_list-doc\_size    = lt\_packing\_list-body\_num \* 255.  
  APPEND lt\_packing\_list. CLEAR lt\_packing\_list.  
  
  lt\_mailsubject-obj\_name     = 'MAILATTCH'.  
  lt\_mailsubject-obj\_langu    = sy-langu.  
  lt\_mailsubject-obj\_descr    = subject.  
  lt\_mailsubject-sensitivty   = 'F'.  
  gv\_cnt = lines( lt\_attachment ).  
  lt\_mailsubject-doc\_size     = ( gv\_cnt - 1 ) \* 255 + strlen(  
  lt\_attachment ).  
  
\*Finally, send the mail out.  
\*That’s it. You are all done. Just call the function module to send the mail out.  
  
  CALL FUNCTION 'SO\_NEW\_DOCUMENT\_ATT\_SEND\_API1'  
    EXPORTING  
      document\_data              = lt\_mailsubject  
    TABLES  
      packing\_list               = lt\_packing\_list  
      contents\_bin               = lt\_attachment  
      contents\_txt               = lt\_mailtxt  
      receivers                  = lt\_mailrecipients  
    EXCEPTIONS  
      too\_many\_receivers         = 1  
      document\_not\_sent          = 2  
      document\_type\_not\_exist    = 3  
      operation\_no\_authorization = 4  
      parameter\_error            = 5  
      x\_error                    = 6  
      enqueue\_error              = 7  
      OTHERS                     = 8.  
  IF sy-subrc EQ 0.  
    COMMIT WORK.  
    SUBMIT rsconn01 WITH mode = 'INT' AND RETURN.  
    MESSAGE s006.  
  ENDIF.  
  clear it\_pm.  
ENDFORM. "send\_notification

**Adobe form : ZGO1\_ADOBE\_FORM\_X**

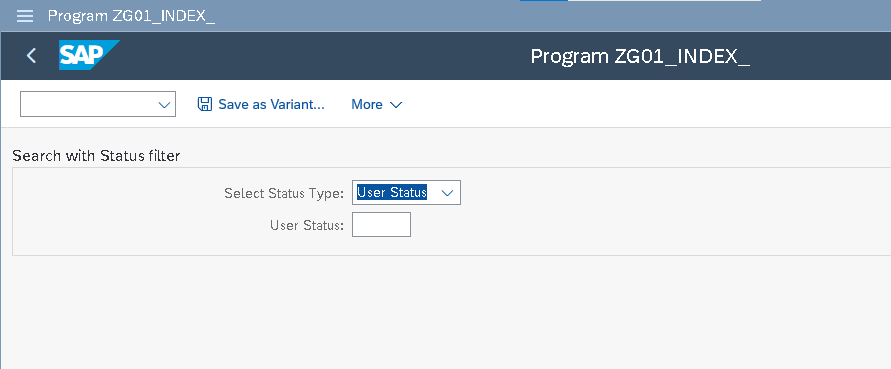
**Interface for adobe form : ZGO1\_IF\_ADOBE\_X**

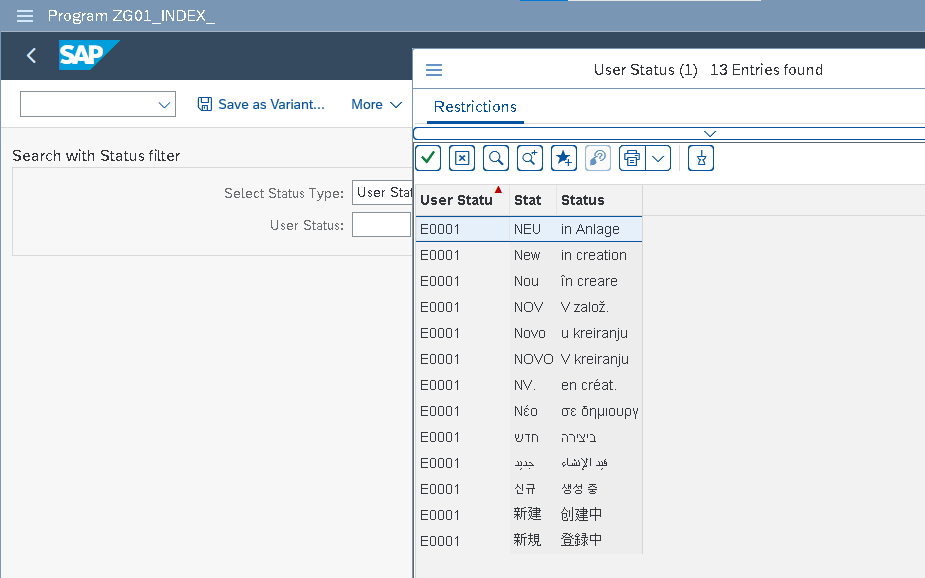
\*&---------------------------------------------------------------------\*  
\*& Include          ZWORKORDER\_ADOBE\_FORM  
\*&---------------------------------------------------------------------\*  
  
INCLUDE Zworkorder\_adobe\_top\_inc.  
  
\*Takes a parameter named p\_aufnr(Order Number) of type any (which can hold various data types).  
FORM zf\_create\_form USING p\_aufnr TYPE any.  
  
  TABLES:  aufk.  
  DATA : lv\_aufpl TYPE afko-aufpl,     " Variable to store number of operations  
         it\_tab   TYPE TABLE OF zg01\_lt\_op\_details, "Internal table for Operational Details  
         wa\_tab   TYPE zg01\_lt\_op\_details, " Workarea for operational details  
         lt\_tab3  TYPE TABLE OF zg01\_lt\_op\_details.  
  
  
\*set values for the output parameters(wa\_outputparams) of a function call (FP\_JOB\_OPEN).  
  wa\_outputparams-device = TEXT-008.  "Output Device  
  wa\_outputparams-dest = TEXT-009.    "Output Destination  
  wa\_outputparams-nodialog = 'X'.     "Supress dialogbox  
  wa\_outputparams-preview = 'X'.      "Print preview  
  
\*query to retrieve a single row of data from the table zg01\_aufk based on the value of p\_aufnr  
  SELECT SINGLE aufnr, ktext,auart,autyp,werks FROM zg01\_aufk "order details  
    INTO  @DATA(lt\_tab1)  
    WHERE aufnr EQ @p\_aufnr.  
  IF sy-subrc IS NOT INITIAL.  
    MESSAGE e003.  
  ENDIF.  
  
\*query to retrieve a single row of data from the table zg01\_qmel based on the value of p\_aufnr  
  SELECT SINGLE qmnum, qmart, qmtxt, artpr, priok "notification details  
    FROM zg01\_qmel  
    INTO @DATA(lt\_tab2)  
    WHERE aufnr EQ @p\_aufnr.  
  IF sy-subrc IS NOT INITIAL.  
    MESSAGE e003.  
  ENDIF.  
  
\*retrieves a single field(aufpl - no.of.operations) from the table zg01\_afko based on order number.  
  SELECT SINGLE aufpl FROM zg01\_afko  
    INTO lv\_aufpl  
    WHERE aufnr EQ p\_aufnr.  
  IF sy-subrc IS NOT INITIAL.  
    MESSAGE e003.  
  ENDIF.  
  
\*retrieves multiple rows of data from the table zg01\_afvc based on no.of.operations(lv\_aufpl).  
  SELECT vornr,ltxa1, steus, larnt, anzzl   "operation details  
    INTO TABLE @lt\_tab3  
    FROM zg01\_afvc  
    WHERE aufpl EQ @lv\_aufpl.  
  IF sy-subrc IS NOT INITIAL.  
    MESSAGE e003.  
  ENDIF.  
  
  LOOP AT lt\_tab3 INTO DATA(wa\_tab3).  
    MOVE-CORRESPONDING wa\_tab3 TO wa\_tab.  
\*system variable holds the current index position within the loop and  
\*assigns current loop index (row number) to a field named index\_no within the work area wa\_tab  
    MOVE sy-tabix TO wa\_tab-index\_no.  
    APPEND wa\_tab TO it\_tab.  
  ENDLOOP.  
  
  CALL FUNCTION 'FP\_JOB\_OPEN'            "Function call for output Parameters  
    CHANGING  
      ie\_outputparams = wa\_outputparams  
    EXCEPTIONS  
      cancel          = 1  
      usage\_error     = 2  
      system\_error    = 3  
      internal\_error  = 4  
      OTHERS          = 5.  
  IF sy-subrc <> 0.  
\* Implement suitable error handling here  
  ENDIF.  
  
  TRY.  
      CALL FUNCTION 'FP\_FUNCTION\_MODULE\_NAME'    "Function call for Adobe form  
        EXPORTING  
          i\_name     = 'ZGO1\_ADOBE\_FORM\_X'  
        IMPORTING  
          e\_funcname = gv\_fm\_name.  
    CATCH cx\_fp\_api\_internal.  
      MESSAGE e004.  
    CATCH cx\_fp\_api\_usage.  
      MESSAGE e004.  
    CATCH cx\_fp\_api\_repository.  
      MESSAGE e004.  
  
  
  ENDTRY.  
  
  CALL FUNCTION gv\_fm\_name "'/1BCDWB/SM00000458' "Auto generated function module in form  
    EXPORTING  
      gt\_tab         = it\_tab  
      wa\_order       = lt\_tab1  
      wa\_notif       = lt\_tab2  
    EXCEPTIONS  
      usage\_error    = 1  
      system\_error   = 2  
      internal\_error = 3  
      OTHERS         = 4.  
  IF sy-subrc <> 0.  
\* Implement suitable error handling here  
  ENDIF.  
  .  
  
  CALL FUNCTION 'FP\_JOB\_CLOSE'  
    EXCEPTIONS  
      usage\_error    = 1  
      system\_error   = 2  
      internal\_error = 3  
      OTHERS         = 4.  
  IF sy-subrc <> 0.  
\* Implement suitable error handling here  
  ENDIF.  
ENDFORM.

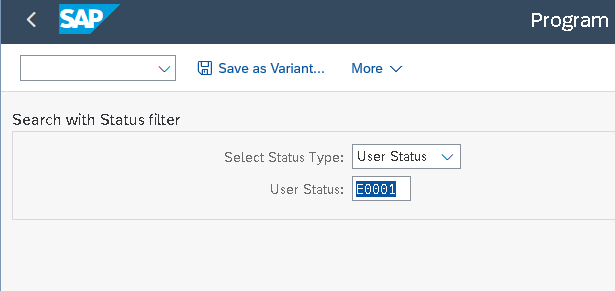
\*&---------------------------------------------------------------------\*  
\*& Include          ZWORKORDER\_PERFORM  
\*&---------------------------------------------------------------------\*  
  
START-OF-SELECTION.  "beginning of the report execution  
  
\*set the value of a local variable(lv\_status) based on another variable(ps\_parm)  
IF ps\_parm = TEXT-001. "User selection from the dropdown.  
  lv\_status = usr\_stat. "local variable to store a status value based on the condition  
ELSE.  
  lv\_status = stm\_stat.  
ENDIF.  
  
\*calls a subroutine get\_data responsible for retrieving data based on status type set by the user.  
perform zf\_get\_data.  
  
\*Calls subroutines to send notification emails for the respective distribution lists based on the order type.  
PERFORM zf\_send\_email\_notification USING text-005 . "DL\_BP\_PM01  
PERFORM zf\_send\_email\_notification USING text-006 . "DL\_BP\_PM02

\*&---------------------------------------------------------------------\*  
\*& Include          ZG01\_WORKORDER\_OOALV  
\*&---------------------------------------------------------------------\*  
  
PERFORM :   ZF\_ALV\_fcat USING TEXT-010 TEXT-011, "  'ID Num'                'AUFNR',  
            ZF\_ALV\_fcat USING TEXT-012 TEXT-013, "  'TEXT DESC'             'KTEXT',  
            ZF\_ALV\_fcat USING TEXT-014 TEXT-015, "  'ORD Type'              'AUART',  
            ZF\_ALV\_fcat USING TEXT-016 TEXT-017, "  'ORD Category'          'AUTYP',  
            ZF\_ALV\_fcat USING TEXT-018 TEXT-019, "  'Location'              'STORT',  
            ZF\_ALV\_fcat USING TEXT-020 TEXT-021, "  'Num.of Operations'     'AUFPL',  
            ZF\_ALV\_fcat USING TEXT-022 TEXT-023, "  'Activity Num'          'VORNR',  
            ZF\_ALV\_fcat USING TEXT-024 TEXT-025, "  'Control Key'           'STEUS',  
            ZF\_ALV\_fcat USING TEXT-026 TEXT-027, "  'Operation Short Text'  'LTXA1',  
            ZF\_ALV\_fcat USING TEXT-028 TEXT-029, "  'Activity Type'         'LARNT',  
            ZF\_ALV\_fcat USING TEXT-030 TEXT-031. " 'Num.of Capacity'        'ANZZL'.  
  
\*CALL SCREEN statement triggers the PBO section of the called screen immediately  
CALL SCREEN 0100.  
  
MODULE status\_0100 OUTPUT.  
  SET PF-STATUS space.   "No User Commands  
ENDMODULE.  
  
\*  &---------------------------------------------------------------------\*  
\*  &      Module  USER\_COMMAND\_0100  INPUT  
\*  &---------------------------------------------------------------------\*  
MODULE user\_command\_0100 INPUT.  
  DATA ok\_code TYPE sy-ucomm. "variable OK\_CODE to store the user command identifier.  
\*If OK\_CODE is equal to "Back" , the program leaves the screen and returns to selection screen  
  CASE ok\_code.  
    WHEN TEXT-032. " 'BACK'.  
      LEAVE TO SCREEN 0.  
  ENDCASE.  
ENDMODULE.  
  
\*sets up the ALV grid display within screen 0100.  
MODULE ooalv\_out OUTPUT.  
  
ls\_layout-grid\_title  = TEXT-033. " 'Finding WorkOrder Notification Based on Order Type'.  
  
\*creates a container object named CONT1 to hold the ALV grid  
  CREATE OBJECT i\_container  
    EXPORTING  
      container\_name = TEXT-034. " 'CONT1'  
  .  
  IF sy-subrc <> 0.  
  
  ENDIF.  
  
\*creates an ALV grid object as a child of the container(i\_container).  
  CREATE OBJECT i\_ALVgrid  
    EXPORTING  
      i\_parent = i\_container.  
    IF sy-subrc <> 0.  
\*   MESSAGE ID SY-MSGID TYPE SY-MSGTY NUMBER SY-MSGNO  
\*              WITH SY-MSGV1 SY-MSGV2 SY-MSGV3 SY-MSGV4.  
    ENDIF.  
  
  DATA : i\_handler TYPE REF TO lcl\_alv\_inter. "variable i\_handler refers to class lcl\_alv\_inter.  
  CREATE OBJECT i\_handler.  
  
\*sets a double-click handler for the ALV grid(i\_ALVgrid)  
  SET HANDLER : i\_handler->handle\_double\_click FOR i\_ALVgrid.  
  
\*calls a method of  ALV grid object(i\_ALVgrid) for its initial display.  
  CALL METHOD i\_ALVgrid->set\_table\_for\_first\_display  
    EXPORTING  
      i\_structure\_name              = TEXT-035 "'COMBO\_STRUCT'  
      is\_layout                     = ls\_layout  
    CHANGING  
      it\_outtab                     = it\_pmx  " hold the data to be displayed in the grid  
      it\_fieldcatalog               = lt\_fcat2 "defines the mapping between table fields and grid columns  
    EXCEPTIONS  
      invalid\_parameter\_combination = 1  
      program\_error                 = 2  
      too\_many\_lines                = 3  
      OTHERS                        = 4.  
  IF sy-subrc <> 0.  
\*   Implement suitable error handling here  
  ENDIF.  
  
ENDMODULE.  
  
\*p\_text: holds the text for the column header.  
\*p\_field: specifies the field name that will be displayed in the column.  
FORM ZF\_alv\_fcat USING p\_text TYPE any p\_field TYPE any  .  
  
  STATICS index TYPE i VALUE 1. "used to assign a unique position to each entry in field catalog  
  
  wa\_fcat2-col\_pos = index.    "Column Position  
  wa\_fcat2-coltext = p\_text.   "Column Text  
  wa\_fcat2-tabname = TEXT-036. " 'IT\_PMX'  
  wa\_fcat2-fieldname = p\_field."Field name  
  APPEND wa\_fcat2 TO lt\_fcat2.  
  CLEAR wa\_fcat2.  
  
  index = index + 1.  
  
ENDFORM.

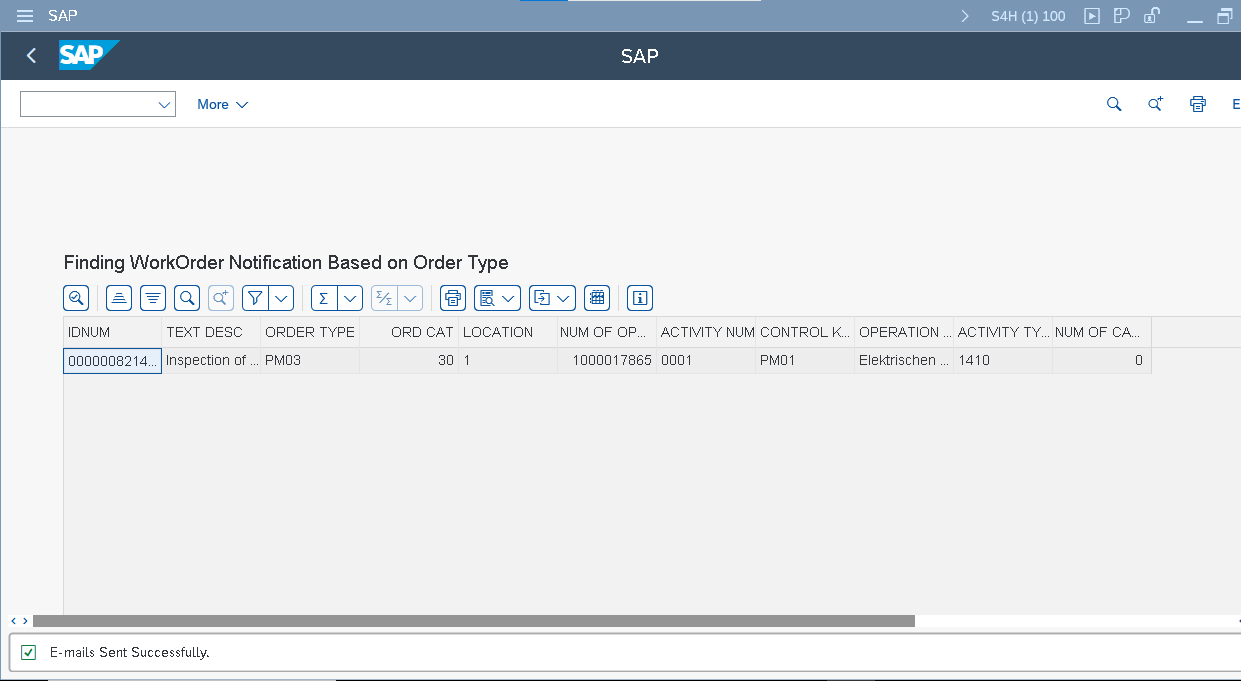
**OUTPUT**



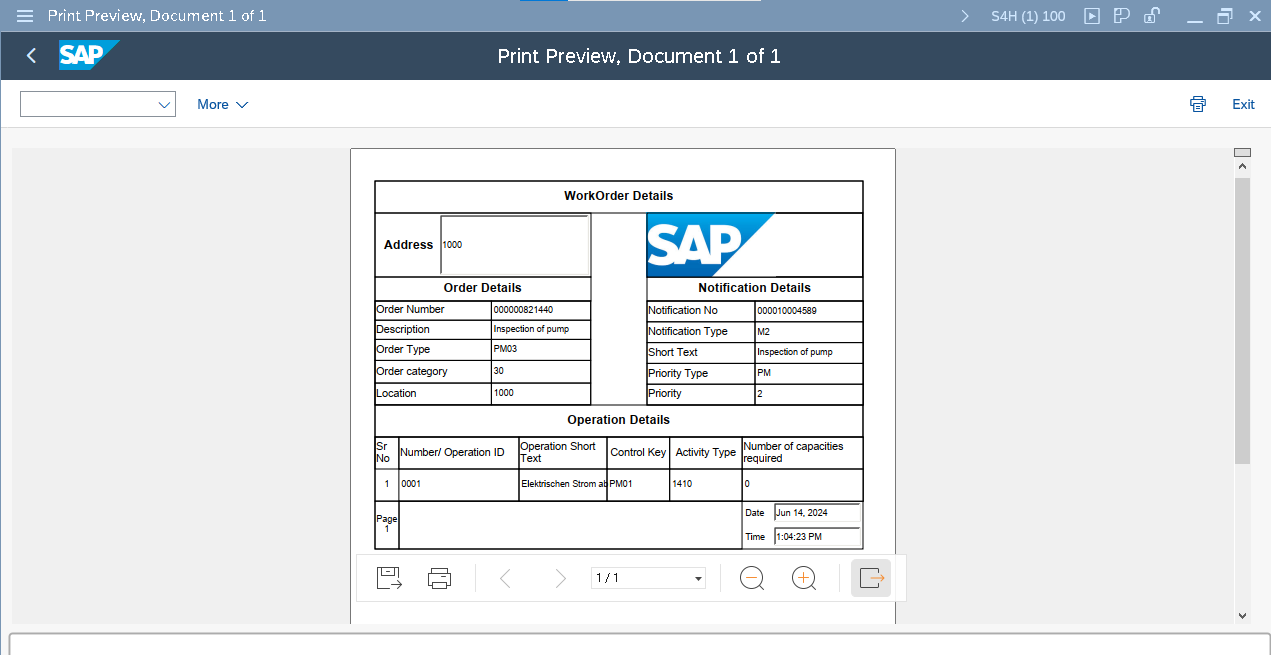




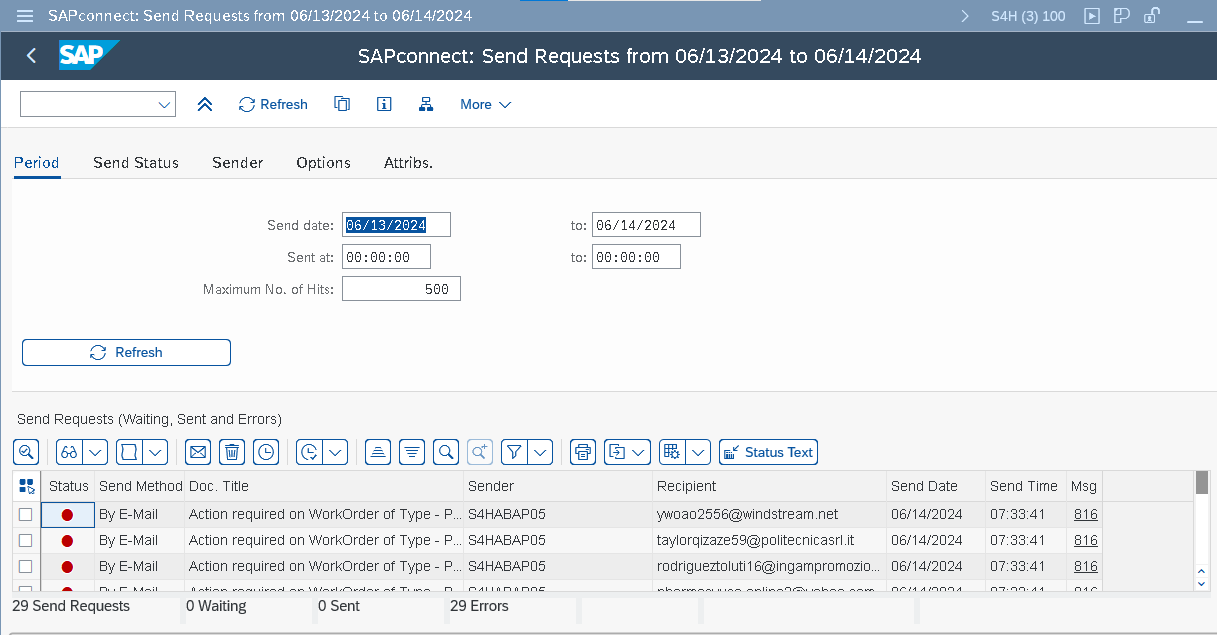
We’ll get the ALV report for the order numbers whose order type is not PM01 AND PM02.



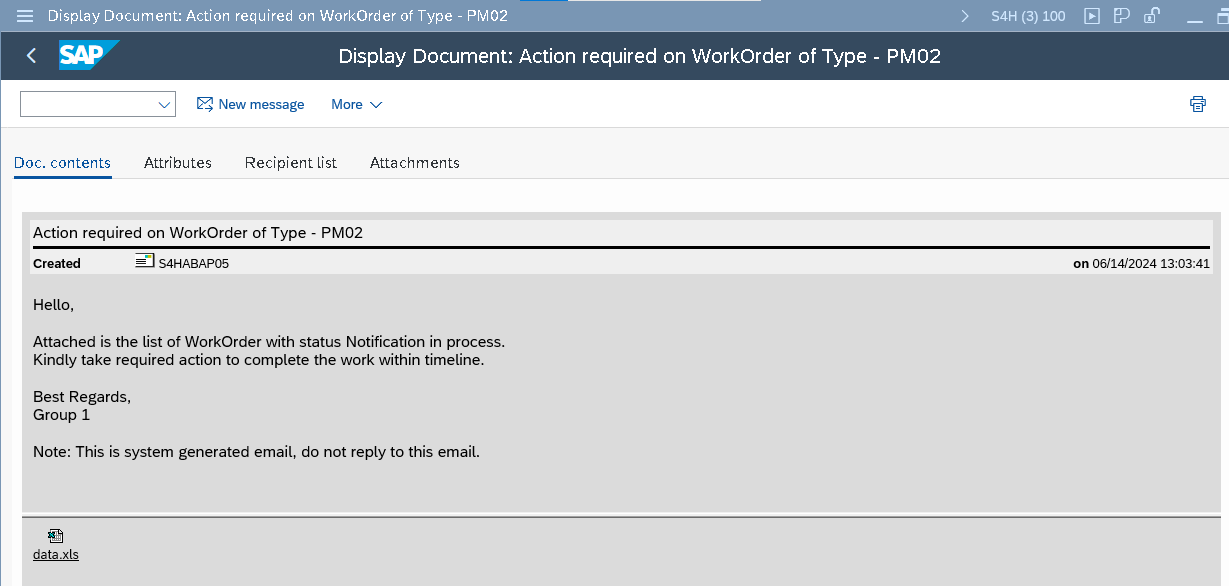
Double click on IDNUM, you’ll be redirected to the Adobe form of that particular order number

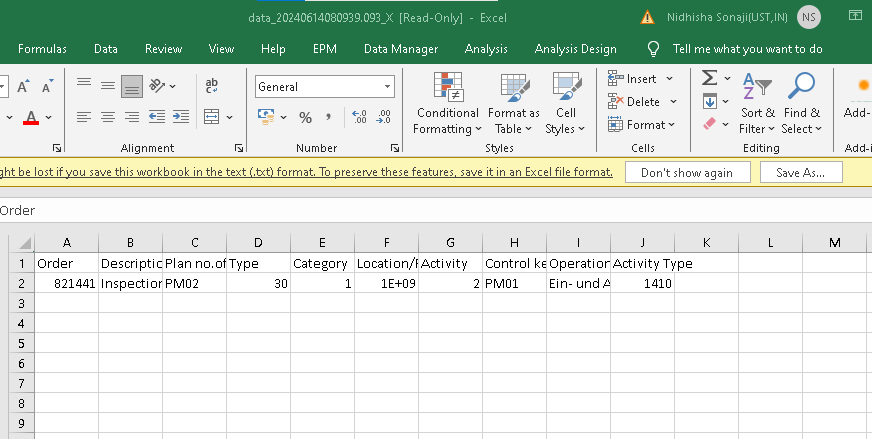


In the backend, emails are sent to the recipients whose order type is PM01 and PM02



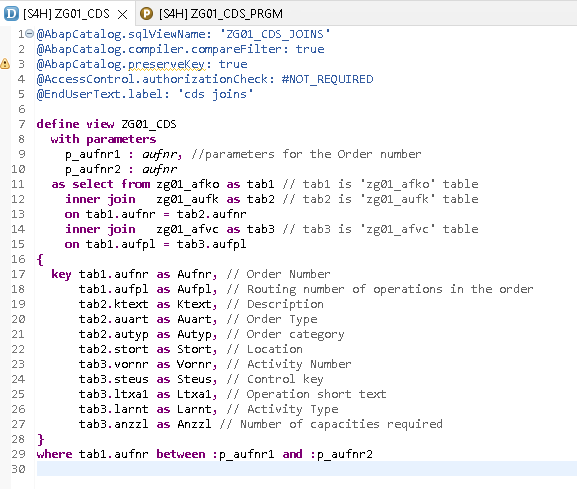
Email Body which is sent to the recipients

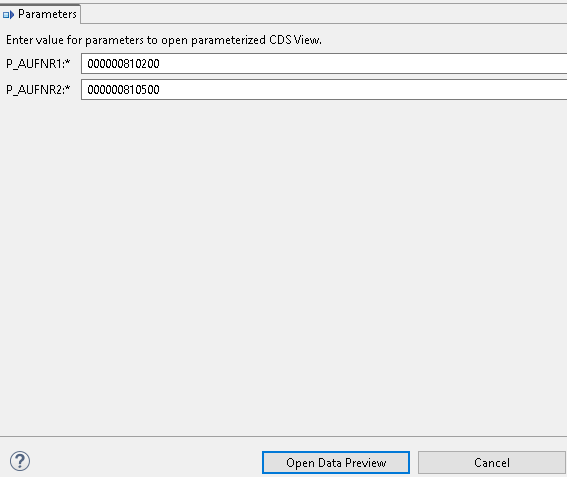


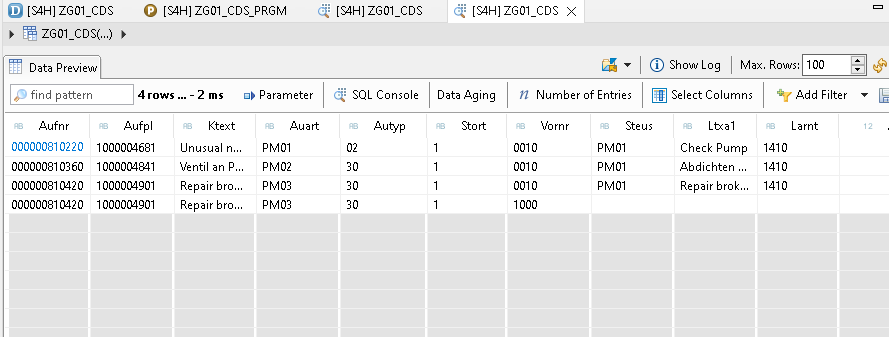


**CDS View**

CDS view performed on ZG01\_AFKO, ZG01\_AUFK and ZG01\_AFVC tables







**DRIVER PROGRAM TO CALL CDS VIEW**

\*&---------------------------------------------------------------------\*  
\*& Report zg01\_cds\_prgm  
\*&---------------------------------------------------------------------\*  
\*&  
\*&---------------------------------------------------------------------\*  
REPORT zg01\_cds\_prgm.  
  
data lv\_aufnr type zg01\_aufnr. "creating local variable for AUFNR  
select-options s\_aufnr for lv\_aufnr. "select options  'range low to high  
  
data(i\_salv) = cl\_salv\_gui\_table\_ida=>create\_for\_cds\_view(  
                 iv\_cds\_view\_name      = 'ZG01\_CDS').  "giving the CDS view name which we created  
i\_salv->set\_view\_parameters( it\_parameters = VALUE #( ( name = 'p\_aufnr1' value = s\_aufnr-low )  
                                                      ( name = 'p\_aufnr2' value = s\_aufnr-high ) ) ).  
data(o\_fcat) = i\_salv->field\_catalog(  ).  
  o\_fcat->set\_field\_header\_texts(  
  iv\_field\_name        = 'AUFNR'  
  iv\_header\_text       = 'Order No'  
  ).  
  o\_fcat->set\_field\_header\_texts(  
  iv\_field\_name        = 'AUFPL'  
  iv\_header\_text       = 'Routine no of Operations'  
  ).  
  o\_fcat->set\_field\_header\_texts(  
  iv\_field\_name        = 'KTEXT'  
  iv\_header\_text       = 'Description'  
  ).  
  o\_fcat->set\_field\_header\_texts(  
  iv\_field\_name        = 'AUART'  
  iv\_header\_text       = 'Order Type'  
  ).  
  o\_fcat->set\_field\_header\_texts(  
  iv\_field\_name        = 'AUTYP'  
  iv\_header\_text       = 'Order Category'  
  ).  
  o\_fcat->set\_field\_header\_texts(  
  iv\_field\_name        = 'STORT'  
  iv\_header\_text       = 'Location'  
  ).  
  o\_fcat->set\_field\_header\_texts(  
  iv\_field\_name        = 'VORNR'  
  iv\_header\_text       = 'Activity No'  
  ).  
  o\_fcat->set\_field\_header\_texts(  
  iv\_field\_name        = 'STEUS'  
  iv\_header\_text       = 'Control Key'  
  ).  
  o\_fcat->set\_field\_header\_texts(  
  iv\_field\_name        = 'LTXA1'  
  iv\_header\_text       = 'Operation Short Text'  
  ).  
  o\_fcat->set\_field\_header\_texts(  
  iv\_field\_name        = 'LARNT'  
  iv\_header\_text       = 'Activity Type'  
  ).  
  o\_fcat->set\_field\_header\_texts(  
  iv\_field\_name        = 'ANZZL'  
  iv\_header\_text       = 'No Of Capacities Required'  
  ).  
  i\_salv->fullscreen(  )->display(  ).

**OUTPUT**

