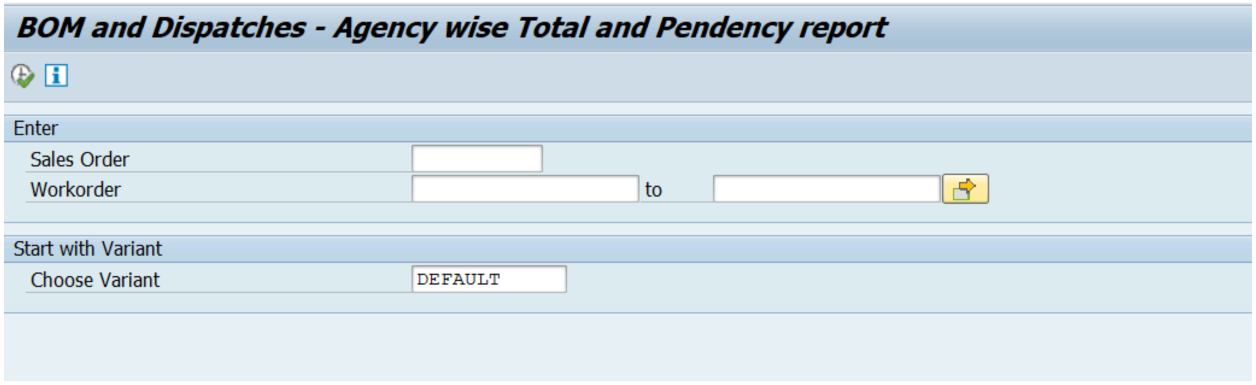
Project Dispatch Monitoring System:



BOM and Dispatches - Agency wise Total and Pendency report

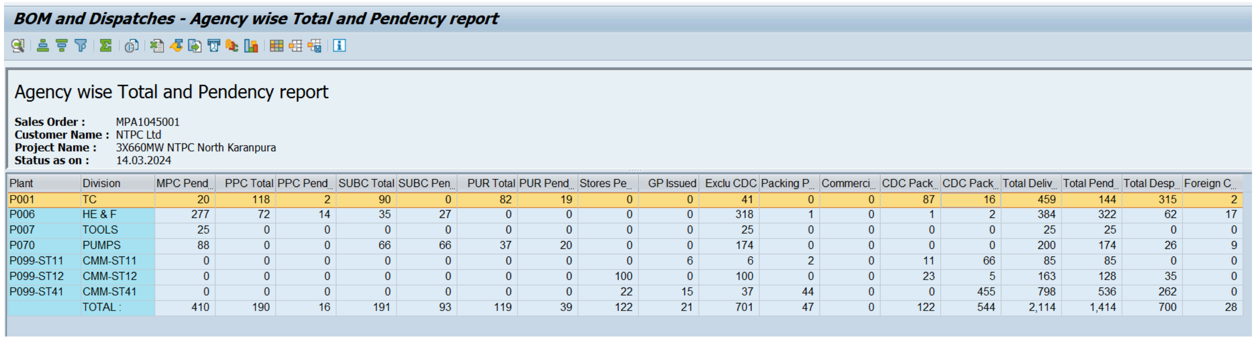
Enter

Sales Order			
Workorder		to	

Start with Variant

Choose Variant	DEFAULT

Sample Report



BOM and Dispatches - Agency wise Total and Pendency report

Agency wise Total and Pendency report

Sales Order : MPA1045001
Customer Name : NTPC Ltd
Project Name : 3X660MW NTPC North Karanpura
Status as on : 14.03.2024

Plant	Division	MPC Pend...	PPC Total	PPC Pend...	SUBC Total	SUBC Pen...	PUR Total	PUR Pend...	Stores Pe...	GP Issued	Exclu CDC	Packing P...	Commerci...	CDC Pack...	CDC Pack...	Total Deliv...	Total Pend...	Total Desp...	Foreign C...
P001	TC	20	118	2	90	0	82	19	0	0	41	0	0	87	16	459	144	315	2
P006	HE & F	277	72	14	35	27	0	0	0	0	318	1	0	1	2	384	322	62	17
P007	TOOLS	25	0	0	0	0	0	0	0	0	25	0	0	0	0	25	25	0	0
P070	PUMPS	88	0	0	66	66	37	20	0	0	174	0	0	0	0	200	174	26	9
P099-ST11	CMM-ST11	0	0	0	0	0	0	0	0	6	6	2	0	11	66	85	85	0	0
P099-ST12	CMM-ST12	0	0	0	0	0	0	0	100	0	100	0	0	23	5	163	128	35	0
P099-ST41	CMM-ST41	0	0	0	0	0	0	0	22	15	37	44	0	0	455	798	536	262	0
	TOTAL :	410	190	16	191	93	119	39	122	21	701	47	0	122	544	2,114	1,414	700	28

```
*& Logic for Program:
*& Before Execution: Check from VBAK that Sales Order is of type ZMAI.
*& For Summary Data:
*& Step 1.   Find Unit level WBSE by adding '-'s and converting to Input format
.
*& Step 2.   Find Customer Name from VBAP and ADRC Tables.
*& Step 3.   Find Project Number and Project Name from PRHI and PRPS Tables.
*& Step 4.   Get the DD Stock from MSPR table for All DD Items.
*&           (Wherever Stock is available and PGI not done, item is dispatched)
.
*& Step 5.   Get reservation, Status, Gatepass etc. from ZMMT001 table.
*& Step 6.   Get Deliveries and their PGI Status from VBFA and VBUP Tables.
*& Step 7.   Based on Status Derive the Dispatch status of the item.
*&           1. If GI Status = X, Staus will be dispached
*&           2. Else if Stock is available for DD, Staus will be dispached
*&           3. Else if Packing list is available, Ststus will be Packed
*&           4. Else if Sales_Ind = X, Status will be Copied to SO
*&           5. Else if Goods_ind = X, Status will be Acknowledged by CDC
*&           5. Else if Res Status = 12, Status will be Gate Pass Generated
*& Step 8.   Count the deliverables based on Plant and Agency
*& Step 9.   Display Summary Data
*&
*&
*& For Detailed Data:
```

```
*& Display Plant, Division, Storage Location, WorkOrder, MatlCode, Description
,
*&           DU, Qty, Unit, Weight, Partno, Reservation No, Item and Status.
*& If Agency is MPC, Display Picklist, Pickslip and Gate Pass Number & Date.
*& If Agency is SUBC, Display Picklist, Pickslip and Gate Pass Number & Date
*&           PR No, PR Release date, PO Number PO date, Vendor, PO Delivery Da
te.
*& If Agency is PUR, Display PR No, PR Release date, PO Number PO date, Vendor
,
*&           PO Delivery Date, RDR No, RDR Date, RDR Qty, LR No, LR Date, Tran
sporter.
*& If Agency is Stores, Display Picklist, Pickslip Number & Date.
*& If Agency is GP Issued, Display Picklist, Pickslip and Gate Pass Number & D
ate.
*& If item is Pending for Packing, Display SO Copying Date.
*& If item is Packed, Display OBD No & Date, Export Pkg No, B/L No & Date.
*& If Column is Total Pendency, Display CDC Status, SO Item & Created Date, OB
D No & Date, etc.
*& If Column is Total Despatched, Display OBD No & Date, Shipment Details, PR
& PO details, RDR details.
*&
*&
*& Logic for Details:
*& Reservation No, Item, Status, Picklist, Pickslip and Gate Pass Number & Dat
e:
*& Step 1.  All data is available in ZMMT001. These fields are already picked
while preparing
*&          Summary data as ZMMT001 table was used for getting reservation Sta
tus.
*&          Data can be displayed unsing available data in internal table.
*& Sales Order Item and Copying Date:
*& Step 1.  Data available in ZECMT006. These fields are already picked while
preparing
*&          Summary data as ZECMT006 table was used for getting deliverables.
*&          Data can be displayed unsing available data in internal table.
*& OBD Number, Date, Export Pkg Number, B/L Number & Date:
*& Step 1.  Pass Sales Order and Item Number into LIPS table to get VBELN fiel
d.
*&          VBELN of LIPS will be Packing list number.
*& Step 2.  Pass LIPS-VBELN (Packing List) into LIKP to get BLDAT (PL date),
*&          ZZDID (Export Pkg Number), BOLNR (B/L Number) & PODAT (Date)
*& PR No, PR Release date, PO Number PO date, Vendor,
*&          PO Delivery Date, RDR No, RDR Date, RDR Qty, LR No, LR Date, Trans
porter:
*& Step 1.  Pass Res No & Item into RESB to get system generated PRs.
*&          For Manual PRS, Pass WBSE into EBKN to get BANFN and BNFPO.
*&          Pass BANFN and BANFN and BNFPO to get corresponding Material Codes
.
*&          Based on WBSE and Material Code, get correct BANFN and BANPO.
*& Step 2.  With table EKPO System generated POs can be extracted.
*&          Using EKPO and EKKN tables, Manual POs can be extracted in the sam
es way PRs are extracted.
*& Step 3.  Pass PO Number into EKKO to get Vendor Code, PO Delivery Date.
*& Step 4.  Pass PO Number and PO Item into MSEG with 101 Movement Type, DD99
Storage Location,
*&          Q Special Stock Indicator to get MBLNR (RDR Number)
*& Step 5.  Pass MSEG-MBLNR into MKPF to get BLDAT (RDR Date)
*& Step 6.  Pass MSEG-MBLNR into VBFA to get Inbound delivery number.
*& Step 7.  Pass Inbound delivery number to LIKP in order to get LR No & Date,
 Trnasporter.
*&=====================================================================*

REPORT  zsdr0078.

TYPE-POOLS:slis.
TABLES: vbak, zecmt006.
DATA :it_events TYPE slis_t_event WITH HEADER LINE.

DATA: int_fieldcat TYPE slis_t_fieldcat_alv,
      int_detcat   TYPE slis_t_fieldcat_alv.
DATA: wa_fieldcat LIKE LINE OF int_fieldcat,
      wa_detcat   LIKE LINE OF int_fieldcat,
      gd_repid    LIKE sy-repid,
      gd_layout   TYPE slis_layout_alv,
      gx_variant  LIKE disvariant.

DATA: is_variant LIKE disvariant,
      es_variant LIKE disvariant.

DATA: BEGIN OF int_vbeln OCCURS 0,  "For F4 on input paramater
        vbeln TYPE vbak-vbeln,
        name1 TYPE adrc-name1,
      END OF int_vbeln.

DATA: BEGIN OF int_vbpaf4 OCCURS 0,  "For F4 on input paramater
        vbeln TYPE vbpa-vbeln,
        kunnr TYPE vbpa-kunnr,
        adrnr TYPE vbpa-adrnr,
      END OF int_vbpaf4.

DATA: BEGIN OF int_adrcf4 OCCURS 0,  "For F4 on input paramater
        addrnumber TYPE adrc-addrnumber,
        name1      TYPE adrc-name1,
        name2      TYPE adrc-name2,
      END OF int_adrcf4.

DATA: BEGIN OF int_ecm OCCURS 0,  "ZECMT006 Data
        projn            TYPE zecmt006-projn,
        pspid            TYPE zecmt006-pspid,
        pspnr            TYPE zecmt006-pspnr,
        posnr            TYPE zecmt006-posnr,
        werks            TYPE zecmt006-werks,
        matnr            TYPE zecmt006-matnr,
        zdesno           TYPE zecmt006-zdesno,
        partno           TYPE zecmt006-partno,
        des_serial       TYPE zecmt006-des_serial,
        posid             TYPE zecmt006-posid,
        meins             TYPE zecmt006-meins,
        maktx             TYPE zecmt006-maktx,
        menge             TYPE zecmt006-menge,
        zznetwt           TYPE zecmt006-zznetwt,
        total_price       TYPE zecmt006-total_price,
        dutype            TYPE zecmt006-dutype,
        status            TYPE zecmt006-status,
        zline_creat_date  TYPE zecmt006-zline_creat_date,
        sales_indi        TYPE zecmt006-sales_indi,
        goods_indi        TYPE zecmt006-goods_indi,
        item              TYPE zecmt006-item,
        zzlegacydes       TYPE zecmt006-zzlegacydes,
        fcwono            TYPE zecmt006-fcwono,
        pkgno             TYPE zecmt006-pkgno,   " added on 14082015
        zwono             TYPE zmmt001-zwono,
        zwoitem           TYPE zecmt006-zwoitem,
        plan_desp_date    TYPE zecmt006-plan_desp_date,
        andat             TYPE zecmt006-andat,            " Added by Siva
      END OF int_ecm.

DATA: BEGIN OF int_temp OCCURS 0,      "ZECMT006 Deliverables for MA/MD DU Typ
es
        projn             TYPE zecmt006-projn,
        pspid             TYPE zecmt006-pspid,
        pspnr             TYPE zecmt006-pspnr,
        zdesno            TYPE zecmt006-zdesno,
        posnr             TYPE zecmt006-posnr,
        werks             TYPE zecmt006-werks,
        partno            TYPE zecmt006-partno,
        des_serial        TYPE zecmt006-des_serial,
        meins             TYPE zecmt006-meins,
        maktx             TYPE makt-maktx,
        menge             TYPE zecmt006-menge,
        zznetwt           TYPE zntgew,
        total_price       TYPE zecmt006-total_price,
        dutype            TYPE zecmt006-dutype,
        status            TYPE zecmt006-status,
        zline_creat_date  TYPE zecmt006-zline_creat_date,
        sales_indi        TYPE zecmt006-sales_indi,
        goods_indi        TYPE zecmt006-goods_indi,
        item              TYPE zecmt006-item,
        zzlegacydes       TYPE zecmt006-zzlegacydes,
        fcwono            TYPE zecmt006-fcwono,
        zwono             TYPE zmmt001-zwono,
        pkgno_pgma        TYPE zecmt006-pkgno, " added on 03092015
        zwoitem           TYPE zecmt006-zwoitem,
        plan_desp_date    TYPE zecmt006-plan_desp_date,
        andat             TYPE zecmt006-andat,
      END OF int_temp.

DATA: BEGIN OF int_zmmt001 OCCURS 0,     "ZMMT001 Data for status
        rsnum        TYPE zmmt001-rsnum,
        rspos       TYPE zmmt001-rspos,
        zstatu1     TYPE zmmt001-zstatu1,
        bwart       TYPE zmmt001-bwart,
        xloek       TYPE zmmt001-xloek,
        kzear       TYPE zmmt001-kzear,
        zwono       TYPE zmmt001-zwono,
        matnr       TYPE zmmt001-matnr,
        pspel       TYPE zmmt001-pspel,
        werks       TYPE zmmt001-werks,
        lgort       TYPE zmmt001-lgort,
        aufnr       TYPE zmmt001-aufnr,    " ADDED ON 07.11.2015
        sortf       TYPE zmmt001-sortf,
        zpickslpno  TYPE zmmt001-zpickslpno,
        zplbldat    TYPE zmmt001-zplbldat,
        zpickslpno1 TYPE zmmt001-zpickslpno1,
        zpsbldat    TYPE zmmt001-zpsbldat,
        zgpno       TYPE zmmt001-zgpno,
        zgpdate     TYPE zmmt001-zgpdate,
      END OF int_zmmt001.

DATA: BEGIN OF int_makt OCCURS 0,
        matnr TYPE makt-matnr,
        maktx TYPE makt-maktx,
      END OF int_makt.

DATA: BEGIN OF int_vbfa OCCURS 0,
        vbeln TYPE vbfa-vbeln,
        posnn TYPE vbfa-posnn,
        vbelv TYPE vbfa-vbelv,
        posnv TYPE vbfa-posnv,
        wbsta TYPE vbup-wbsta,
        fksta TYPE vbup-fksta,
      END OF int_vbfa.

DATA: BEGIN OF int_vbup OCCURS 0,
        vbeln TYPE vbup-vbeln,
        posnr TYPE vbup-posnr,
        wbsta TYPE vbup-wbsta,
        fksta TYPE vbup-fksta,
      END OF int_vbup.

DATA: BEGIN OF it_zmmt001_temp OCCURS 0,
        rsnum TYPE zmmt001-rsnum,
        rspos TYPE zmmt001-rspos,
        aufnr TYPE zmmt001-aufnr,
      END OF it_zmmt001_temp.

DATA: BEGIN OF int_lips1 OCCURS 0,
        vbeln TYPE lips-vbeln,
        posnr TYPE lips-posnr,
        qplos TYPE lips-qplos,

      END OF int_lips1.
```

```
DATA: BEGIN OF int_final OCCURS 0,     "Final table for data
        werks(10)             TYPE c,
        projn                 TYPE zecmt006-projn,
        pspid                 TYPE zecmt006-pspid,
        pspnr                 TYPE zecmt006-pspnr,
        posnr                 TYPE zecmt006-posnr,
        zdesno                TYPE zecmt006-zdesno,
        partno                TYPE zecmt006-partno,
        des_serial            TYPE zecmt006-des_serial,
        meins                 TYPE zecmt006-meins,
        maktx                 TYPE zecmt006-maktx,
        menge                 TYPE zecmt006-menge,
        zznetwt               TYPE zecmt006-zznetwt,
        total_price           TYPE zecmt006-total_price,
        dutype                TYPE zecmt006-dutype,
        status                TYPE zecmt006-status,
        zline_creat_date      TYPE zecmt006-zline_creat_date,
        sales_indi            TYPE zecmt006-sales_indi,
        goods_indi            TYPE zecmt006-goods_indi,
        item                  TYPE zecmt006-item,
        zzlegacydes           TYPE zecmt006-zzlegacydes,
        fcwono                TYPE zecmt006-fcwono,
        pkgno_pgma            TYPE zecmt006-pkgno , "added on 140815
        zwoitem               TYPE zecmt006-zwoitem,
        deliv                 TYPE vbup-vbeln,
        delin                 TYPE vbup-posnr,
        wbsta                 TYPE vbup-wbsta,
        fksta                 TYPE vbup-wbsta,
        zwono                 TYPE zmmt001-zwono,
        rsnum                 TYPE zmmt001-rsnum,
        rspos                 TYPE zmmt001-rspos,
        zstatu1               TYPE zmmt001-zstatu1,
        bwart                 TYPE zmmt001-bwart,
        xloek                 TYPE zmmt001-xloek,
        kzear                 TYPE zmmt001-kzear,
        lgort                 TYPE zmmt001-lgort,
        aufnr                 TYPE zmmt001-aufnr,  " ADDED ON 07.11.2015
        zpickslpno            TYPE zmmt001-zpickslpno,
        zplbldat              TYPE zmmt001-zplbldat,
        zpickslpno1           TYPE zmmt001-zpickslpno1,
        zpsbldat              TYPE zmmt001-zpsbldat,
        zgpno                 TYPE zmmt001-zgpno,
        zgpdate               TYPE zmmt001-zgpdate,
        mpctot                TYPE i,
        mpcpen                TYPE i,
        ppctot                TYPE i,
        ppcpen                TYPE i,
        subtot                TYPE i,
        subpen                TYPE i,
        purtot                TYPE i,
        purpen                TYPE i,
        strpen                TYPE i,
        gpissu                TYPE i,
        cdcack                TYPE i,
        cmlpen                TYPE i,
        cdcpak                TYPE i,
        totdesp               TYPE i,
        totfcwono             TYPE i,
        vbeln                 TYPE vbak-vbeln,
        werks_original        TYPE zecmt006-werks,
        banfn                 TYPE eban-banfn,
        bnfpo                 TYPE eban-bnfpo,
        ebeln                 TYPE ekpo-ebeln,
        ebelp                 TYPE ekpo-ebelp,
        post1                 TYPE prps-post1,
        plan_desp_date        TYPE zecmt006-plan_desp_date,
        zzmdcc                TYPE qals-zzmdcc,
        andat                 TYPE zecmt006-andat,              " Added by Siva
        cdcpq                 TYPE i,
        cdc_quality_status(3) TYPE c,
        pur_off(8)            TYPE c,
        pur_name              TYPE pa0001-sname,
      END OF int_final."same table structure to be maintained in MZPP075TOP

DATA: BEGIN OF int_count OCCURS 0,  "Final table to store count
        werks(10) TYPE c,
        divis(15) TYPE c,
        pspnr     TYPE zecmt006-pspnr,
        pspid     TYPE zecmt006-pspid,
        name1     TYPE adrc-name1,
        post1     TYPE prps-post1,
        mpctot    TYPE i,
        mpcpen    TYPE i,
        ppctot    TYPE i,
        ppcpen    TYPE i,
        subtot    TYPE i,
        subpen    TYPE i,
        purtot    TYPE i,
        purpen    TYPE i,
        strpen    TYPE i,
        notcdc    TYPE i,
        gpissu    TYPE i,
        cdcack    TYPE i,
        cmlpen    TYPE i,
        cdcpak    TYPE i,
        tottot    TYPE i,
        totpen    TYPE i,
        totdesp   TYPE i,
        totfcwono TYPE i,
        cdcpq     TYPE i,
      END OF int_count. "same table structure to be maintained in MZPP075TOP

DATA: BEGIN OF int_fcdetails OCCURS 0,
        projn       TYPE zecmt006-projn,
        pspid       TYPE zecmt006-pspid,
        pspnr       TYPE zecmt006-pspnr,
        posnr       TYPE zecmt006-posnr,
        zdesno      TYPE zecmt006-zdesno,
        matnr       TYPE zecmt006-matnr,
        werks(10)   TYPE c,
        partno      TYPE zecmt006-partno,
        des_serial  TYPE zecmt006-des_serial,
        meins       TYPE zecmt006-meins,
        maktx       TYPE zecmt006-maktx,
        menge       TYPE zecmt006-menge,
        zznetwt     TYPE zecmt006-zznetwt,
        total_price TYPE zecmt006-total_price,
        dutype      TYPE zecmt006-dutype,
        status      TYPE zecmt006-status,
        sales_indi  TYPE zecmt006-sales_indi,
        goods_indi  TYPE zecmt006-goods_indi,
        item        TYPE zecmt006-item,
        deliv       TYPE vbup-vbeln,
        deldt       TYPE likp-bldat,
        tknum       TYPE vttk-tknum,
        datbg       TYPE vttk-datbg,
        exti1       TYPE vttk-exti1,
        slno        TYPE i,

        divis(15)   TYPE c,
        sono        TYPE vbfa-vbelv,
        lgort       TYPE zmmt001-lgort,
      END OF int_fcdetails.

DATA: wa_fcdetails LIKE LINE OF int_fcdetails.

TYPES:BEGIN OF ty_details,"same table structure to be maintained in MZPP075TOP
        werks(10)             TYPE c,
        divis(15)             TYPE c,
        projn                 TYPE zecmt006-projn,
        pspid                 TYPE zecmt006-pspid,
        pspnr                 TYPE zecmt006-pspnr,
        posnr                 TYPE zecmt006-posnr,
        matnr                 TYPE zecmt006-matnr,
        zdesno                TYPE zecmt006-zdesno,
        partno                TYPE zecmt006-partno,
        des_serial            TYPE zecmt006-des_serial,
        meins                 TYPE zecmt006-meins,
        maktx                 TYPE zecmt006-maktx,
        menge                 TYPE zecmt006-menge,
        zznetwt               TYPE zecmt006-zznetwt,
        total_price           TYPE zecmt006-total_price,
        dutype                TYPE zecmt006-dutype,
        status                TYPE zecmt006-status,
        zline_creat_date      TYPE zecmt006-zline_creat_date,
        sales_indi            TYPE zecmt006-sales_indi,
        goods_indi            TYPE zecmt006-goods_indi,
        item                  TYPE zecmt006-item,
        zzlegacydes           TYPE zecmt006-zzlegacydes,
        fcwono                TYPE zecmt006-fcwono,
        pkgno_pgma            TYPE zecmt006-pkgno, " added on 140815
        zwoitem               TYPE zecmt006-zwoitem,
        deliv                 TYPE vbup-vbeln,
        delin                 TYPE vbup-posnr,
        zwono                 TYPE zmmt001-zwono,
        rsnum                 TYPE zmmt001-rsnum,
        rspos                 TYPE zmmt001-rspos,
        zstatu1               TYPE zmmt001-zstatu1,
        kzear                 TYPE zmmt001-kzear,
        lgort                 TYPE zmmt001-lgort,
        zpickslpno            TYPE zmmt001-zpickslpno,
        zplbldat              TYPE zmmt001-zplbldat,
        zpickslpno1           TYPE zmmt001-zpickslpno1,
        zpsbldat              TYPE zmmt001-zpsbldat,
        zgpno                 TYPE zmmt001-zgpno,
        zgpdate               TYPE zmmt001-zgpdate,
        tknum                 TYPE vttk-tknum,
        signi                 TYPE vttk-signi,
        exti1                 TYPE vttk-exti1,
        tdlnr                 TYPE vttk-tdlnr,
        sdabw                 TYPE vttk-sdabw,
        datbg                 TYPE vttk-datbg,
        slno                  TYPE i,
        deldt                 TYPE likp-bldat,
        bolnr                 TYPE likp-bolnr,
        podat                 TYPE likp-podat,
        zzdid                 TYPE likp-zzdid,
        pkgno(35)             TYPE c,
        bdmng                 TYPE resb-bdmng,
        frgdt                 TYPE eban-frgdt,
*       PRMEINS TYPE RESB-MEINS,
        banfn                 TYPE resb-banfn,
        bnfpo                 TYPE resb-bnfpo,
        prdat                 TYPE ebkn-erdat,
        ebeln                 TYPE ekpo-ebeln,
        ebelp                 TYPE ekpo-ebelp,
        pomenge               TYPE ekpo-menge,
        pomeins               TYPE ekpo-meins,
        lifnr                 TYPE ekko-lifnr,
        bedat                 TYPE ekko-bedat,
        eq_eindt              TYPE ekko-eq_eindt,
        name1                 TYPE lfa1-name1,
        eindt                 TYPE eket-eindt,
        rdrno                 TYPE mseg-mblnr,
        rdryr                 TYPE mseg-mjahr,
        rdrvl                 TYPE mseg-dmbtr,
        rdrqt                 TYPE mseg-menge,
        rdrdt                 TYPE mkpf-bldat,
        ibdno                 TYPE likp-vbeln,
        rlrno                 TYPE likp-zzrlr,
        rlrdt                 TYPE likp-zzrld,
        zztnm                   TYPE likp-zztnm,
        ack_txt(12)             TYPE c,
        remarks(20)             TYPE c,
        status_desc(50)         TYPE c,  " DES STATUS DESCRIPTION
        post1                   TYPE prps-post1,

        plan_desp_date          TYPE zecmt006-plan_desp_date,  " Planned Despatch
 date in ZECMT006 - to be editable field
        zzmdcc                  TYPE qals-zzmdcc,
        andat                   TYPE zecmt006-andat,
        cdc_quality_status(3)   TYPE c,
        l2_date                 TYPE zsdt135-l2date,
        site_reqdate            TYPE zsdt135-site_reqdate,
        bbu                     TYPE zecmt006-bbu,
*         total_price             TYPE zecmt006-total_price,
        pur_off                 TYPE pa0001-pernr,
        engg_off                TYPE pa0001-pernr,
        cdc_off(8)              TYPE c,
        pur_name                TYPE pa0001-sname,
        engg_name               TYPE pa0001-sname,
        cdc_name                TYPE pa0001-sname,
      END OF ty_details."same table structure to be maintained in MZPP075TOP

DATA: BEGIN OF int_vttp OCCURS 0,
        tknum TYPE vttp-tknum,
        tpnum TYPE vttp-tpnum,
        vbeln TYPE vttp-vbeln,
      END OF int_vttp.

DATA: BEGIN OF int_vttk OCCURS 0,
        tknum TYPE vttk-tknum,
        signi TYPE vttk-signi,
        exti1 TYPE vttk-exti1,
        tdlnr TYPE vttk-tdlnr,
        sdabw TYPE vttk-sdabw,
        datbg TYPE vttk-datbg,
      END OF int_vttk.

DATA: BEGIN OF int_likp OCCURS 0,
        vbeln TYPE likp-vbeln,
        deldt TYPE likp-bldat,
        bolnr TYPE likp-bolnr,
        podat TYPE likp-podat,
        zzdid TYPE likp-zzdid,
        ernam TYPE likp-ernam,
      END OF int_likp.

DATA: BEGIN OF int_mspr OCCURS 0,
        matnr TYPE mspr-matnr,
        werks TYPE mspr-werks,
        lgort TYPE mspr-lgort,
        charg TYPE mspr-charg,
        pspnr TYPE mspr-pspnr,
        prlab TYPE mspr-prlab,
      END OF int_mspr.

DATA: BEGIN OF it_mseg OCCURS 0,
        mblnr TYPE mseg-mblnr,
        mjahr TYPE mseg-mjahr,
        bwart TYPE mseg-bwart,
        menge TYPE mseg-menge,
        ebeln TYPE mseg-ebeln,
        ebelp TYPE mseg-ebelp,
        lfbja TYPE mseg-lfbja,
        lfbnr TYPE mseg-lfbnr,
        sjahr TYPE mseg-sjahr,
      END OF it_mseg,
      it_mseg_ekkn   LIKE TABLE OF it_mseg WITH HEADER LINE,
      it_mseg_ekpo   LIKE TABLE OF it_mseg WITH HEADER LINE,
      it_mseg_upload LIKE TABLE OF it_mseg WITH HEADER LINE,
      it_mseg102     LIKE TABLE OF it_mseg WITH HEADER LINE,

      BEGIN OF it_eban OCCURS 0,
        banfn      TYPE eban-banfn,
        bnfpo      TYPE eban-bnfpo,
        matnr      TYPE eban-matnr,
        bednr      TYPE eban-bednr,
        ps_psp_pnr TYPE ebkn-ps_psp_pnr,
        ebeln      TYPE ekpo-ebeln,
        ebelp      TYPE ekpo-ebelp,
        werks      TYPE eban-werks,
      END OF it_eban,

      BEGIN OF it_ekkn OCCURS 0,
        ebeln      TYPE ekpo-ebeln,
        ebelp      TYPE ekpo-ebelp,
        matnr      TYPE eban-matnr,
        bednr      TYPE eban-bednr,
        ps_psp_pnr TYPE ekkn-ps_psp_pnr,
        werks      TYPE ekpo-werks,
        zzpurexec  TYPE ekko-zzpurexec,
      END OF it_ekkn,

      BEGIN OF it_ekpo OCCURS 0,
        ebeln     TYPE ekpo-ebeln,
        ebelp     TYPE ekpo-ebelp,
        banfn     TYPE eban-banfn,
        bnfpo     TYPE eban-bnfpo,
        zzpurexec TYPE ekko-zzpurexec,
      END OF it_ekpo,

      BEGIN OF int_pr OCCURS 0,
        banfn TYPE eban-banfn,
        bnfpo TYPE eban-bnfpo,
      END OF int_pr,
      BEGIN OF int_po OCCURS 0,
        ebeln TYPE ekpo-ebeln,
        ebelp TYPE ekpo-ebelp,
      END OF int_po.

DATA: BEGIN OF it_zmmt001 OCCURS 0,
        rsnum TYPE zmmt001-rsnum,
        rspos TYPE zmmt001-rspos,
        aufnr TYPE zmmt001-aufnr,
      END OF it_zmmt001.

DATA : v_ddqty TYPE mseg-menge.
DATA: BEGIN OF it_sowo_validate OCCURS 0,
        pspid TYPE zecmt006-pspid,
        pspnr TYPE zecmt006-pspnr,
      END OF it_sowo_validate.

DATA: temp_werks            TYPE zecmt006-werks,
      temp_zznetwt          TYPE zecmt006-zznetwt,
      temp_price            TYPE zecmt006-total_price,
      temp_dutype           TYPE zecmt006-dutype,
      temp_status           TYPE zecmt006-status,
      temp_zline_creat_date TYPE zecmt006-zline_creat_date,
      temp_sales_indi       TYPE zecmt006-sales_indi,
      temp_goods_indi       TYPE zecmt006-goods_indi,
      temp_item             TYPE zecmt006-item,
      temp_zzlegacydes      TYPE zecmt006-zzlegacydes,
      temp_zwono            TYPE zmmt001-zwono,
      temp_fcwono           TYPE zecmt006-fcwono,
      temp_pkgno            TYPE zecmt006-pkgno,
      temp_zwoitem          TYPE zecmt006-zwoitem,
      temp_plandt           TYPE zecmt006-plan_desp_date,
      temp_posnr            TYPE zecmt006-posnr,
      v_pspnr               TYPE zecmt006-pspnr,
      temp_andat            TYPE zecmt006-andat.
DATA:int_zsdt077 LIKE STANDARD TABLE OF zsdt077 WITH HEADER LINE.
DATA: int_details    TYPE STANDARD TABLE OF ty_details WITH HEADER LINE,
      int_details_cd TYPE STANDARD TABLE OF ty_details WITH HEADER LINE,   " ADD
ED ON 16.11.2011
      int_rdr_det    TYPE STANDARD TABLE OF ty_details WITH HEADER LINE.

*DATA: it_zmmt170 LIKE STANDARD TABLE OF zmmt170 WITH HEADER LINE.

DATA: int_det_tem TYPE STANDARD TABLE OF ty_details WITH HEADER LINE.

DATA: e_network TYPE TABLE OF bapi_network_list WITH HEADER LINE.
*DATA: e_component  TYPE TABLE OF bapi_component_exp WITH HEADER LINE.
DATA: e_component  TYPE TABLE OF  bapi_network_comp_detail WITH HEADER LINE,
      e_component1 TYPE TABLE OF  bapi_network_comp_detail WITH HEADER LINE.

DATA: t_auart     LIKE vbak-auart,
      temp_pspid  TYPE zecmt006-pspid,
      zzpcd       LIKE vbak-zzpcd,
      p_pspidin   LIKE zecmt006-pspid,
      p_pspid(25) TYPE c,
      t_wono(25)  TYPE c,
      rep_dt(12)  TYPE c,
      t_adrnr     LIKE vbpa-adrnr,
      t_name1     LIKE adrc-name1,
      t_projid    LIKE prps-posid,
      t_projname  LIKE prps-post1,
      t_projname1 LIKE prps-post1,
      sl          TYPE i,
      mpctot      TYPE i,
      mpcpen      TYPE i,
      ppctot      TYPE i,
      ppcpen      TYPE i,
      subtot      TYPE i,
      subpen      TYPE i,
      purtot      TYPE i,
      purpen      TYPE i,
      strpen      TYPE i,
      notcdc      TYPE i,
      gpissu      TYPE i,
      cdcack      TYPE i,
      cmlpen      TYPE i,
      cdcpak      TYPE i,
      cdcpq       TYPE i,
      tottot      TYPE i,
      totpen      TYPE i,
      totdesp     TYPE i,
      totfcwono   TYPE i,
      temp_vbeln  TYPE vbak-vbeln.
DATA:int_zmmt208 LIKE STANDARD TABLE OF zmmt208 WITH HEADER LINE.
DATA :rowsel(24)  TYPE c,
      colsel(24)  TYPE c,
      d_divis(15) TYPE c.
DATA:BEGIN OF int_prps OCCURS 0,
       pspnr LIKE prps-pspnr,
       psphi LIKE prps-psphi,
     END OF int_prps.
DATA:wa_werks(4) TYPE c.
***********************Customer Name, Project Name**************

DATA: BEGIN OF it_vbpa_adrnr OCCURS 0,
        vbeln TYPE vbpa-vbeln,
        adrnr TYPE vbpa-adrnr,
      END OF it_vbpa_adrnr.

DATA: BEGIN OF it_adrc OCCURS 0,
        addrnumber TYPE adrc-addrnumber,
        name1      TYPE adrc-name1,
      END OF it_adrc.

DATA : BEGIN OF int_qals OCCURS 0,
         prueflos TYPE qals-prueflos,                     "lot number
         zzmdcc   TYPE    qals-zzmdcc,                 "MDCC number
         ls_vbeln TYPE qals-ls_vbeln,
         ls_posnr TYPE qals-ls_posnr,
       END OF int_qals.


DATA : BEGIN OF int_qave OCCURS 0,
         prueflos TYPE qals-prueflos,

       END OF int_qave.

DATA: BEGIN OF it_prhi_pname OCCURS 0,
        posnr TYPE prhi-posnr,
        up    TYPE prhi-up,
      END OF it_prhi_pname.

DATA: BEGIN OF it_prps_pname OCCURS 0,
        pspnr TYPE prps-pspnr,
        post1 TYPE prps-post1,
      END OF it_prps_pname.

DATA : w_name(50) TYPE c,
       w_accnt    TYPE likp-ernam,
       w_pernr(8) TYPE c,
       w_submi    TYPE zmmt003-submi.

DATA : int_desc  TYPE TABLE OF dd07v WITH HEADER LINE,
       int_desc1 TYPE TABLE OF dd07v WITH HEADER LINE.

DATA : BEGIN OF int_prpspost1 OCCURS 0,
         pspnr TYPE prps-pspnr,
         post1 TYPE prps-post1,
       END OF int_prpspost1.

SELECTION-SCREEN BEGIN OF BLOCK b1 WITH FRAME TITLE t1.
PARAMETERS: p_vbeln TYPE vbak-vbeln,
            ind(10) NO-DISPLAY, "used in ZPMG_PPC (Project Management Report)
            col(6)  NO-DISPLAY, "used in ZPMG_PPC (Project Management Report)
            wo      TYPE zecmt006-pspnr NO-DISPLAY. "used in ZPMG_PPC (Project
Management Report)
SELECT-OPTIONS: s_pspnr FOR zecmt006-pspnr.
SELECTION-SCREEN END OF BLOCK b1.

SELECTION-SCREEN BEGIN OF BLOCK b2 WITH FRAME TITLE title2.
PARAMETERS: p_var LIKE disvariant-variant DEFAULT 'DEFAULT'.
SELECTION-SCREEN END OF BLOCK b2.

INITIALIZATION.
  t1 = 'Enter'.
  title2 = 'Start with Variant'.

AT SELECTION-SCREEN .
  IF p_vbeln IS NOT INITIAL.
```

```
    SELECT SINGLE auart INTO t_auart FROM vbak WHERE vbeln = p_vbeln.    "Check
ing Order Existance & Type
    IF sy-subrc = 0.
      SELECT SINGLE auart INTO t_auart FROM vbak WHERE vbeln = p_vbeln AND auar
t = 'ZMAI'.
      IF sy-subrc = 4.
        MESSAGE 'Order is not a Main Order' TYPE 'E'.
      ENDIF.
    ELSE.
      MESSAGE 'Order is not available in System' TYPE 'E'.
    ENDIF.

    IF s_pspnr IS NOT INITIAL.
      SELECT pspid pspnr
        INTO TABLE it_sowo_validate
      FROM zecmt006
      WHERE pspnr IN s_pspnr.

      CALL FUNCTION 'CONVERSION_EXIT_ABPSP_INPUT'
        EXPORTING
          input     = p_vbeln
        IMPORTING
          output    = temp_pspid
        EXCEPTIONS
          not_found = 1
          OTHERS    = 2.
      IF sy-subrc <> 0.
        MESSAGE ID sy-msgid TYPE sy-msgty NUMBER sy-msgno
                WITH sy-msgv1 sy-msgv2 sy-msgv3 sy-msgv4.
      ENDIF.

      LOOP AT it_sowo_validate WHERE pspid NE temp_pspid.
        MESSAGE 'Workorder does not belong to the salesorder' TYPE 'E'.
        EXIT.
      ENDLOOP.
    ENDIF.
  ENDIF.


AT SELECTION-SCREEN ON VALUE-REQUEST FOR p_vbeln.
  SELECT vbeln INTO TABLE int_vbeln FROM vbak WHERE auart = 'ZMAI'.
  IF int_vbeln[] IS NOT INITIAL.
    SELECT vbeln kunnr adrnr FROM vbpa INTO TABLE int_vbpaf4
      FOR ALL ENTRIES IN int_vbeln
      WHERE vbeln = int_vbeln-vbeln.
  ENDIF.
  IF int_vbpaf4[] IS NOT INITIAL.
    SELECT addrnumber name1 name2 FROM adrc INTO TABLE int_adrcf4
      FOR ALL ENTRIES IN int_vbpaf4
      WHERE addrnumber = int_vbpaf4-adrnr.
  ENDIF.
  LOOP AT int_vbeln.
    READ TABLE int_vbpaf4 WITH KEY vbeln = int_vbeln-vbeln.
```

```
      IF sy-subrc EQ 0.
        READ TABLE int_adrcf4 WITH KEY addrnumber = int_vbpaf4-adrnr.
        IF sy-subrc EQ 0.
          int_vbeln-name1 = int_adrcf4-name1.
          MODIFY int_vbeln.
        ENDIF.
      ENDIF.
    ENDLOOP.

    SORT int_vbeln BY  name1 vbeln.

    IF int_vbeln IS NOT INITIAL.
      CALL FUNCTION 'F4IF_INT_TABLE_VALUE_REQUEST'
        EXPORTING
          retfield        = 'VBELN'
          dynpprog        = 'ZSDR0078'
          dynpnr          = sy-dynnr
          dynprofield     = 'P_VBELN'
          value_org       = 'S'
        TABLES
          value_tab       = int_vbeln
        EXCEPTIONS
          parameter_error = 1
          no_values_found = 2
          OTHERS          = 3.
    ENDIF.

  AT SELECTION-SCREEN ON VALUE-REQUEST FOR p_var.
    is_variant-report = sy-cprog.
    is_variant-username = sy-uname.
    CALL FUNCTION 'REUSE_ALV_VARIANT_F4'
      EXPORTING
        is_variant = is_variant
        i_save     = 'A'
      IMPORTING
        es_variant = es_variant
      EXCEPTIONS
        not_found  = 1.

    IF sy-subrc = 0.
      p_var = es_variant-variant.
    ENDIF.


  START-OF-SELECTION.
  ************ Commented on 02.06.16 as new program zsdp0036 has created *******
  **************

  *  IF sy-batch eq 'X'.
  *    break 6172601.
  *    CLEAR:int_zmmt208[].
  *    SELECT * FROM zmmt208 INTO CORRESPONDING FIELDS OF TABLE int_zmmt208.
  *    IF int_zmmt208[] IS NOT INITIAL.
```

```
*      CLEAR:int_prps[].
*      SELECT pspnr psphi FROM prps INTO CORRESPONDING FIELDS OF TABLE int_prp
s
*                                 FOR ALL ENTRIES IN int_zmmt208
*                                 WHERE psphi = int_zmmt208-pspnr
*                                 AND stufe = '3'.
*      DATA:wa_pspnr(24) TYPE c.
*      LOOP AT int_prps.
*        CLEAR:s_pspnr.
*        CALL FUNCTION 'CONVERSION_EXIT_ABPSP_OUTPUT'
*          EXPORTING
*            input  = int_prps-pspnr
*          IMPORTING
*            output = wa_pspnr.
*        IF wa_pspnr+2(2) EQ '10'
*           OR wa_pspnr+2(2) EQ '16'
*           OR wa_pspnr+2(2) EQ '18'.
*          s_pspnr-sign = 'I'.
*          s_pspnr-option = 'EQ'.
*          s_pspnr-low = int_prps-pspnr.
*          APPEND s_pspnr.
*        ENDIF.
*      ENDLOOP.
*      PERFORM get_data.
*      break 6172601.
*      DELETE int_count WHERE werks = ''.
*      SORT int_count BY werks.
*
*      LOOP AT int_count.
*        READ TABLE int_prps WITH KEY pspnr = int_count-pspnr.
*        IF sy-subrc EQ 0.
*          IF int_count-werks+0(4) NE 'P099'.
*            wa_werks = int_count-werks.
*          ELSE.
*            wa_werks = int_count-werks+0(4).
*          ENDIF.
*          READ TABLE int_zsdt077 WITH KEY psphi = int_prps-psphi werks = wa_w
erks.
*          IF sy-subrc EQ 0.
*            PERFORM incr_count.
*          ELSE.
*            PERFORM add_record USING wa_werks.
*          ENDIF.
*        ENDIF.
*      ENDLOOP.
*      break 6172601.
*      IF int_zsdt077[] IS NOT INITIAL.
*        MODIFY zsdt077 FROM TABLE int_zsdt077[].
*      ENDIF.
*    ENDIF.
*  ELSE.
```

```
  IF ind = 'ZPMS_PR'.
    PERFORM get_data.
    DELETE FROM MEMORY ID 'ZPMS_PR'.
    EXPORT int_count TO MEMORY ID 'ZPMS_PR_SUM'.
    DELETE FROM MEMORY ID 'ZPMS_PR_DET'.
    EXPORT int_final TO MEMORY ID 'ZPMS_PR_DET'.
  ELSEIF ind = 'ZPROJMGMT'.
    PERFORM get_data.
    DELETE FROM MEMORY ID 'ZPROJMGMTSUM'.
    EXPORT int_count TO MEMORY ID 'ZPROJMGMTSUM'.
    DELETE FROM MEMORY ID 'ZPROJMGMTFIN'.
    EXPORT int_final TO MEMORY ID 'ZPROJMGMTFIN'.
  ELSEIF ind = 'ZPROJDET'.
    IMPORT int_count FROM MEMORY ID 'ZPROJMGMTSUM'.
    IMPORT int_final FROM MEMORY ID 'ZPROJMGMTFIN'.
    READ TABLE int_count WITH KEY pspnr = wo.
    IF col = 'MPCPEN'.
      PERFORM get_details USING int_count-werks 'INT_COUNT-MPCPEN' wo.
    ELSEIF col = 'PPCPEN'.
      PERFORM get_details USING int_count-werks 'INT_COUNT-PPCPEN' wo.
    ELSEIF col = 'SUBPEN' .
      PERFORM get_details USING int_count-werks 'INT_COUNT-SUBPEN' wo.
    ELSEIF col = 'PURPEN'.
      PERFORM get_details USING int_count-werks 'INT_COUNT-PURPEN' wo.
    ELSEIF col = 'STRPEN'.
      PERFORM get_details USING int_count-werks 'INT_COUNT-STRPEN' wo.
    ELSEIF col = 'TOTPEN' .
      PERFORM get_details USING int_count-werks 'INT_COUNT-TOTPEN' wo.
    ELSEIF col = 'CDCACK'.
      PERFORM get_details USING int_count-werks 'INT_COUNT-CDCACK' wo.
    ELSEIF col = 'CMLPEN'.
      PERFORM get_details USING int_count-werks 'INT_COUNT-CMLPEN' wo.
    ELSEIF col = 'CDCPAK' .
      PERFORM get_details USING int_count-werks 'INT_COUNT-CDCPAK' wo.
    ELSEIF col = 'CDCPQ' .
      PERFORM get_details USING int_count-werks 'INT_COUNT-CDCPQ' wo.
    ENDIF.
    DELETE FROM MEMORY ID 'ZPROJMGMTDET'.
    EXPORT int_details TO MEMORY ID 'ZPROJMGMTDET'.
  ELSE.
    PERFORM get_data.
    CALL FUNCTION 'ZBHEL_HIT_COUNT'.
    PERFORM build_catalog.
    PERFORM display_data.
  ENDIF.
*  ENDIF.


*&---------------------------------------------------------------------*
*&      Form  GET_DATA
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
```

```
FORM get_data.

  REFRESH: int_ecm, int_temp, int_zmmt001, int_makt, int_vbfa, int_vbup, int_fi
nal, int_count, int_vttp, int_vttk, int_likp, int_mspr, int_details, int_rdr_de
t.
  CLEAR  : int_ecm, int_temp, int_zmmt001, int_makt, int_vbfa, int_vbup, int_fi
nal, int_count, int_vttp, int_vttk, int_likp, int_mspr, int_details, int_rdr_de
t.
  CLEAR: sl, mpctot, mpcpen, ppctot, ppcpen, subtot, subpen, purtot, purpen, st
rpen, notcdc, gpissu, cdcack, cdcpak, tottot, totpen, totdesp,cdcpq.

  IF p_vbeln IS NOT INITIAL.
    CONCATENATE p_vbeln+0(1) '-' p_vbeln+1(2) '-' p_vbeln+3(4) '-' p_vbeln+7(3)
 INTO p_pspid.   "Getting WBS
    CALL FUNCTION 'CONVERSION_EXIT_ABPSP_INPUT'
      EXPORTING
        input  = p_pspid
      IMPORTING
        output = p_pspidin.

* projn             TYPE zecmt006-projn,
*        pspid            TYPE zecmt006-pspid,
*        pspnr            TYPE zecmt006-pspnr,
*        posnr            TYPE zecmt006-posnr,
*        werks            TYPE zecmt006-werks,
*        matnr            TYPE zecmt006-matnr,
*        zdesno           TYPE zecmt006-zdesno,
*        partno           TYPE zecmt006-partno,
*        des_serial       TYPE zecmt006-des_serial,
*        meins            TYPE zecmt006-meins,
*        maktx            TYPE zecmt006-maktx,
*        menge            TYPE zecmt006-menge,
*        zznetwt          TYPE zecmt006-zznetwt,
*        total_price      TYPE zecmt006-total_price,
*        dutype           TYPE zecmt006-dutype,
*        status           TYPE zecmt006-status,
*        zline_creat_date TYPE zecmt006-zline_creat_date,
*        sales_indi       TYPE zecmt006-sales_indi,
*        goods_indi       TYPE zecmt006-goods_indi,
*        item             TYPE zecmt006-item,
*        zzlegacydes      TYPE zecmt006-zzlegacydes,
*        fcwono           TYPE zecmt006-fcwono,
*        pkgno            TYPE zecmt006-pkgno,  " added on 14082015
*        zwono            TYPE zmmt001-zwono,
*        zwoitem          TYPE zecmt006-zwoitem,
*        plan_desp_date   TYPE zecmt006-plan_desp_date,


    SELECT projn pspid pspnr posnr werks matnr zdesno partno des_serial posid m
eins maktx menge zznetwt total_price dutype status zline_creat_date sales_indi
goods_indi item zzlegacydes fcwono pkgno zwoitem  zwoitem plan_desp_date andat
    FROM zecmt006 INTO TABLE int_ecm
    WHERE pspid = p_pspidin
```

```
      AND pspnr IN s_pspnr.

  ELSEIF s_pspnr IS NOT INITIAL.
    SELECT projn pspid pspnr posnr werks matnr zdesno partno des_serial posid m
eins maktx menge zznetwt total_price dutype status zline_creat_date sales_indi
goods_indi item zzlegacydes fcwono pkgno zwoitem zwoitem plan_desp_date andat
    FROM zecmt006 INTO TABLE int_ecm
    WHERE pspnr IN s_pspnr.
  ELSE.
    MESSAGE 'Please Enter at least one input' TYPE 'E'.
  ENDIF.


  IF int_ecm[] IS NOT INITIAL.

    LOOP AT int_ecm.
*       CALL FUNCTION 'CONVERSION_EXIT_ABPSP_OUTPUT'
*         EXPORTING
*           INPUT   = INT_ECM-PSPNR
*         IMPORTING
*           OUTPUT = INT_ECM-ZWONO.
*       REPLACE ALL OCCURRENCES OF '-' IN INT_ECM-ZWONO WITH SPACE.
*       MODIFY INT_ECM TRANSPORTING ZWONO.
      IF int_ecm-matnr NE int_ecm-zdesno AND int_ecm-zdesno+0(1) = 'Z'.
        int_temp-projn = int_ecm-projn.
        int_temp-pspid = int_ecm-pspid.
        int_temp-pspnr = int_ecm-pspnr.

        CALL FUNCTION 'CONVERSION_EXIT_ABPSP_INPUT'
          EXPORTING
            input   = int_ecm-posid
          IMPORTING
            output = v_pspnr
*          EXCEPTIONS
*            NOT_FOUND       = 1
*            OTHERS = 2
          .
        IF sy-subrc <> 0.
* Implement suitable error handling here
        ENDIF.

        int_temp-posnr = v_pspnr.
*         INT_TEMP-POSNR = ''.             "INT_ECM-POSNR.
        int_temp-werks = int_ecm-werks.
        int_temp-zdesno = int_ecm-zdesno.
*         INT_TEMP-PARTNO = ''.
*         INT_TEMP-DES_SERIAL = ''.
        int_temp-meins = 'EA'.
*         INT_TEMP-MAKTX = ''.
        int_temp-menge = 1.
        int_temp-zznetwt = int_ecm-zznetwt.
        int_temp-total_price = int_ecm-total_price.
        int_temp-dutype = int_ecm-dutype.
        int_temp-status = int_ecm-status.
        int_temp-zline_creat_date = int_ecm-zline_creat_date.
        int_temp-sales_indi = int_ecm-sales_indi.
        int_temp-goods_indi = int_ecm-goods_indi.
        int_temp-item = int_ecm-item.
        int_temp-zzlegacydes = int_ecm-zzlegacydes.
        int_temp-fcwono = int_ecm-fcwono.
        int_temp-zwono = int_ecm-zwono.
        int_temp-pkgno_pgma = int_ecm-pkgno. " added on 03092015
        int_temp-zwoitem = int_ecm-zwoitem.
        int_temp-plan_desp_date = int_ecm-plan_desp_date.
        int_temp-andat = int_ecm-andat.
        APPEND int_temp.
        CLEAR int_temp.
      ELSE.
        CALL FUNCTION 'CONVERSION_EXIT_ABPSP_OUTPUT'
          EXPORTING
            input  = int_ecm-pspnr
          IMPORTING
            output = int_ecm-zwono.
        REPLACE ALL OCCURRENCES OF '-' IN int_ecm-zwono WITH space.
*      MODIFY INT_ECM TRANSPORTING ZWONO.
        int_final-projn = int_ecm-projn.
        int_final-pspid = int_ecm-pspid.
        int_final-pspnr = int_ecm-pspnr.
        int_final-posnr = int_ecm-posnr.
        int_final-werks = int_ecm-werks.
        int_final-zdesno = int_ecm-zdesno.
        int_final-partno = int_ecm-partno.
        int_final-des_serial = int_ecm-des_serial.
        int_final-meins = int_ecm-meins.
        int_final-maktx = int_ecm-maktx.
        int_final-menge = int_ecm-menge.
        int_final-zznetwt = int_ecm-zznetwt.
        int_final-total_price = int_ecm-total_price.
        int_final-dutype = int_ecm-dutype.
        int_final-status = int_ecm-status.
        int_final-zline_creat_date = int_ecm-zline_creat_date.
        int_final-sales_indi = int_ecm-sales_indi.
        int_final-goods_indi = int_ecm-goods_indi.
        int_final-item = int_ecm-item.
        int_final-zzlegacydes = int_ecm-zzlegacydes.
        int_final-fcwono = int_ecm-fcwono.
        int_final-zwono = int_ecm-zwono.
        int_final-pkgno_pgma = int_ecm-pkgno. " added on 14082015
        int_final-zwoitem = int_ecm-zwoitem.
        int_final-plan_desp_date = int_ecm-plan_desp_date.
        int_final-andat = int_ecm-andat.
        APPEND int_final.
        CLEAR int_final.
      ENDIF.
    ENDLOOP.
*
```

```
*    SORT INT_TEMP BY PSPNR ZDESNO.
*    LOOP AT INT_TEMP.
*        AT NEW ZDESNO.
*          SUM.
*          MODIFY INT_TEMP.
*          CONTINUE.
*        ENDAT.
*        DELETE INT_TEMP.
*    ENDLOOP.

    IF int_final[] IS NOT INITIAL.      "Getting Status
      SELECT rsnum rspos zstatu1 bwart xloek kzear zwono matnr pspel werks lgor
t aufnr sortf zpickslpno zplbldat zpickslpno1 zpsbldat zgpno zgpdate
        FROM zmmt001 INTO TABLE int_zmmt001
        FOR ALL ENTRIES IN int_final
        WHERE zwono = int_final-zwono AND pspel = int_final-posnr AND matnr = i
nt_final-zdesno AND sortf = int_final-partno
        AND xloek <> 'X' AND bwart = '281'.

      SELECT matnr werks lgort charg pspnr prlab
        FROM mspr INTO TABLE int_mspr
        FOR ALL ENTRIES IN int_ecm
        WHERE matnr = int_ecm-zdesno AND werks = int_ecm-werks AND pspnr = int_
ecm-pspnr
        AND lgort = 'DD99' AND sobkz = 'Q' AND prlab GT int_ecm-menge.
    ENDIF.

    IF int_temp[] IS NOT INITIAL.     "Getting MA/MD Deliverables
      SORT int_temp BY pspnr zdesno.

      CLEAR: temp_werks, temp_zznetwt, temp_price, temp_dutype, temp_status, te
mp_zline_creat_date, temp_sales_indi, temp_goods_indi, temp_item, temp_zzlegacy
des, temp_zwono , temp_plandt.
      LOOP AT int_temp.
        temp_werks = int_temp-werks.
        temp_zznetwt = temp_zznetwt + int_temp-zznetwt.
        temp_price = int_temp-total_price.
        temp_dutype = int_temp-dutype.
        temp_status = int_temp-status.
        temp_zline_creat_date = int_temp-zline_creat_date.
        temp_sales_indi = int_temp-sales_indi.
        temp_goods_indi = int_temp-goods_indi.
        temp_item = int_temp-item.
        temp_zzlegacydes = int_temp-zzlegacydes.
        temp_fcwono = int_temp-fcwono.
        temp_zwono = int_temp-zwono.
        temp_posnr = int_temp-posnr.
        temp_pkgno  = int_temp-pkgno_pgma.
        temp_zwoitem = int_temp-zwoitem.
        temp_andat = int_temp-andat.
        temp_plandt = int_temp-plan_desp_date.
        AT END OF zdesno.
          int_temp-posnr = temp_posnr.
          int_temp-werks = temp_werks.
          int_temp-partno = ''.
          int_temp-des_serial = ''.
          int_temp-meins = 'EA'.
          int_temp-maktx = ''.
          int_temp-menge = 1.
          int_temp-zznetwt = temp_zznetwt.
          int_temp-total_price = temp_price.
          int_temp-dutype = temp_dutype.
          int_temp-status = temp_status.
          int_temp-zline_creat_date = temp_zline_creat_date.
          int_temp-sales_indi = temp_sales_indi.
          int_temp-goods_indi = temp_goods_indi.
          int_temp-item = temp_item.
          int_temp-zzlegacydes = temp_zzlegacydes.
          int_temp-fcwono = temp_fcwono.
          int_temp-zwono = temp_zwono.
          int_temp-pkgno_pgma = temp_pkgno.
          int_temp-zwoitem = temp_zwoitem.
          int_temp-plan_desp_date = temp_plandt.
          int_temp-andat = temp_andat.
          MODIFY int_temp.
          CLEAR: temp_werks, temp_zznetwt, temp_price, temp_dutype, temp_status
, temp_zline_creat_date,
                temp_sales_indi, temp_goods_indi, temp_item, temp_zzlegacydes,t
emp_fcwono, temp_zwono , temp_pkgno, temp_zwoitem,temp_plandt.
          CONTINUE.
        ENDAT.
        DELETE int_temp.
      ENDLOOP.

      SELECT matnr maktx
        FROM makt INTO TABLE int_makt
        FOR ALL ENTRIES IN int_temp
        WHERE matnr = int_temp-zdesno.

      LOOP AT int_temp.
        READ TABLE int_makt WITH KEY matnr = int_temp-zdesno .
        IF sy-subrc = 0.
          int_temp-maktx = int_makt-maktx.
          MODIFY int_temp TRANSPORTING maktx.
        ENDIF.
      ENDLOOP.

      LOOP AT int_temp.
        MOVE-CORRESPONDING int_temp TO int_final.
        APPEND int_final.
        CLEAR int_final.
      ENDLOOP.
    ENDIF.

    break 1896563.
    LOOP AT int_final WHERE vbeln IS INITIAL.
      IF p_vbeln IS NOT INITIAL.
        int_final-vbeln = p_vbeln.
        MODIFY int_final TRANSPORTING vbeln WHERE vbeln IS INITIAL.
      ELSE.
        CLEAR p_pspid.

        CALL FUNCTION 'CONVERSION_EXIT_ABPSP_OUTPUT'
          EXPORTING
            input  = int_final-pspid
          IMPORTING
            output = p_pspid.


        REPLACE ALL OCCURRENCES OF '-' IN p_pspid WITH ''.
        int_final-vbeln = p_pspid.
        MODIFY int_final TRANSPORTING vbeln WHERE pspid = int_final-pspid.
      ENDIF.

    ENDLOOP.

    IF int_final[] IS NOT INITIAL.            "Getting current delivery status

      SELECT vbeln posnn vbelv posnv
        FROM vbfa INTO TABLE int_vbfa
        FOR ALL ENTRIES IN int_final
        WHERE posnv = int_final-item
        AND vbelv = int_final-vbeln
        AND vbtyp_n = 'J'.

      IF int_vbfa[] IS NOT INITIAL.
        SELECT vbeln posnr wbsta fksta
          FROM vbup INTO TABLE int_vbup
          FOR ALL ENTRIES IN int_vbfa
          WHERE vbeln = int_vbfa-vbeln AND posnr = int_vbfa-posnn.

        LOOP AT int_vbfa.
          READ TABLE int_vbup WITH KEY vbeln = int_vbfa-vbeln posnr = int_vbfa-
posnn.
          IF sy-subrc = 0.
            MOVE int_vbup-wbsta TO int_vbfa-wbsta.
            MOVE int_vbup-fksta TO int_vbfa-fksta.
            MODIFY int_vbfa.
          ENDIF.
        ENDLOOP.
      ENDIF.

      DATA: w_vcode TYPE qave-vcode.

      LOOP AT int_final. "Appending Delivery related data
        READ TABLE int_vbfa WITH KEY vbelv = int_final-vbeln posnv = int_final-
item.
        IF sy-subrc = 0.
```

```
          CLEAR: w_vcode.
          MOVE int_vbfa-vbeln TO int_final-deliv.
          MOVE int_vbfa-posnn TO int_final-delin.
          MOVE int_vbfa-wbsta TO int_final-wbsta.
          MOVE int_vbfa-fksta TO int_final-fksta.
*         MODIFY INT_FINAL.

          " Added by Siva 21.12.23
          SELECT SINGLE a~vcode
          FROM lips AS b INNER JOIN qals AS c ON b~qplos = c~prueflos
                         INNER JOIN qave AS a ON a~prueflos = c~prueflos
          INTO w_vcode
            WHERE b~vbeln = int_vbfa-vbeln AND b~posnr = int_vbfa-posnn.
          IF w_vcode = 'AA'.
            int_final-cdc_quality_status  = 'YES'.
          ENDIF.

        ENDIF.
        READ TABLE int_zmmt001 WITH KEY zwono = int_final-zwono pspel = int_fin
al-posnr matnr = int_final-zdesno sortf = int_final-partno bwart = '281'.
        IF sy-subrc = 0.
          MOVE int_zmmt001-werks TO int_final-werks.
          MOVE int_zmmt001-rsnum TO int_final-rsnum.
          MOVE int_zmmt001-rspos TO int_final-rspos.
          MOVE int_zmmt001-zstatu1 TO int_final-zstatu1.
          MOVE int_zmmt001-bwart TO int_final-bwart.
          MOVE int_zmmt001-kzear TO int_final-kzear.
          MOVE int_zmmt001-lgort TO int_final-lgort.
          MOVE int_zmmt001-aufnr TO int_final-aufnr.  " ADDED ON 07.11.2015
          MOVE int_zmmt001-zpickslpno TO int_final-zpickslpno.
          MOVE int_zmmt001-zplbldat TO int_final-zplbldat.
          MOVE int_zmmt001-zpickslpno1 TO int_final-zpickslpno1.
          MOVE int_zmmt001-zpsbldat TO int_final-zpsbldat.
          MOVE int_zmmt001-zgpno TO int_final-zgpno.
          MOVE int_zmmt001-zgpdate TO int_final-zgpdate.
          int_final-werks_original = int_final-werks.
        ENDIF.
        IF int_final-werks = 'P099'.
          CONCATENATE int_final-werks '-' int_final-lgort INTO int_final-werks.
        ENDIF.
        CLEAR: int_final-mpctot, int_final-mpcpen, int_final-ppctot, int_final-
ppcpen, int_final-subtot, int_final-subpen, int_final-purtot, int_final-purpen,
 int_final-strpen, int_final-gpissu, int_final-cdcack, int_final-cdcpak, int_fi
nal-totdesp,
        int_final-cdcpq.
        IF int_final-status EQ 'MVFC' AND int_final-fcwono IS NOT INITIAL.
          int_final-totfcwono = 1.
        ELSEIF int_final-dutype = 'DS'.
          int_final-mpctot = 1.
          IF int_final-status = 'SCLD'.
            int_final-totdesp = 1.
          ELSEIF int_final-wbsta = 'C'.
            int_final-totdesp = 1.
          ELSEIF int_final-deliv <> '' AND int_final-cdc_quality_status = ''.
            int_final-cdcpq = 1.
          ELSEIF int_final-deliv <> '' AND int_final-cdc_quality_status = 'YES'
.
            int_final-cdcpak = 1.
          ELSEIF int_final-goods_indi = 'X' AND int_final-sales_indi = 'X'.
            int_final-cdcack = 1.
          ELSEIF int_final-goods_indi = 'X' AND int_final-sales_indi = ''.
            int_final-cmlpen = 1.
          ELSEIF int_final-zgpno > 0.
            int_final-gpissu = 1.
          ELSEIF  int_final-zpickslpno <> ''.
            int_final-strpen = 1.
          ELSEIF int_final-zpickslpno = ''.
            int_final-mpcpen = 1.
          ENDIF.
        ELSEIF int_final-dutype = 'CD'.
          int_final-subtot = 1.
          IF int_final-status = 'SCLD'.
            int_final-totdesp = 1.
          ELSEIF int_final-wbsta = 'C'.
            int_final-totdesp = 1.
          ELSEIF int_final-deliv <> '' AND int_final-cdc_quality_status = ''.
            int_final-cdcpq = 1.
          ELSEIF int_final-deliv <> '' AND int_final-cdc_quality_status = 'YES'
.
            int_final-cdcpak = 1.
          ELSEIF int_final-goods_indi = 'X' AND int_final-sales_indi = 'X'.
            int_final-cdcack = 1.
          ELSEIF int_final-goods_indi = 'X' AND int_final-sales_indi = ''.
            int_final-cmlpen = 1.
          ELSEIF int_final-zgpno > 0.
            int_final-gpissu = 1.
          ELSEIF  int_final-zpickslpno <> ''.
            int_final-strpen = 1.
          ELSEIF int_final-zpickslpno = ''.
            int_final-subpen = 1.
          ENDIF.
        ELSEIF ( int_final-dutype = 'MA' OR int_final-dutype = 'MD' ).
          int_final-ppctot = 1.
          IF int_final-status = 'SCLD'.
            int_final-totdesp = 1.
          ELSEIF int_final-wbsta = 'C'.
            int_final-totdesp = 1.
          ELSEIF int_final-deliv <> '' AND int_final-cdc_quality_status = ''.
            int_final-cdcpq = 1.
          ELSEIF int_final-deliv <> '' AND int_final-cdc_quality_status = 'YES'
.
            int_final-cdcpak = 1.
          ELSEIF int_final-goods_indi = 'X' AND int_final-sales_indi = 'X'.
            int_final-cdcack = 1.
          ELSEIF int_final-goods_indi = 'X' AND int_final-sales_indi = ''.
            int_final-cmlpen = 1.
          ELSE.
            int_final-ppcpen = 1.
          ENDIF.
        ELSEIF int_final-dutype = 'DD'.
          int_final-purtot = 1.
          IF int_final-status = 'SCLD'.
            int_final-totdesp = 1.
          ELSEIF int_final-wbsta = 'C'.
            int_final-totdesp = 1.
          ELSEIF int_final-wbsta <> 'C'.
            int_final-purpen = 1.
          ENDIF.
        ENDIF.
        MODIFY int_final.
      ENDLOOP.

      LOOP AT int_final WHERE dutype = 'DD' AND totfcwono IS INITIAL.
        READ TABLE int_mspr WITH KEY matnr = int_final-zdesno pspnr = int_final
-pspnr werks = int_final-werks+0(4).
        IF sy-subrc = 0.
          int_final-totdesp = 1.
          int_final-purpen = ''.
          MODIFY int_final TRANSPORTING purpen totdesp.
        ENDIF.
      ENDLOOP.

      " Start of Correcting Purchase Pendency from MSEG table data
**      IF int_final[] IS NOT INITIAL.
**        SELECT ps_psp_pnr matnr charg menge ebeln ebelp
**          FROM mseg
**          INTO TABLE it_mseg
**          FOR ALL ENTRIES IN int_final
**          WHERE matnr = int_final-zdesno
**          AND werks = int_final-werks_original
**          AND lgort = 'DD99'
**          AND bwart = '101'
**          AND sobkz = 'Q'
**          AND ps_psp_pnr = int_final-posnr.
**
**        IF it_mseg[] IS NOT INITIAL.  " ADDED ON 02112015 - TO GET SORTSTRIN
G -BEDNR
**          SELECT ekpo~ebeln ekpo~ebelp "eban~bednr
**                 FROM ekpo INNER JOIN eban ON  ekpo~banfn EQ eban~banfn AND
ekpo~bnfpo EQ eban~bnfpo
**                  INTO TABLE it_eban
**            FOR ALL ENTRIES IN it_mseg
**            WHERE ekpo~ebeln = it_mseg-ebeln
**              AND ekpo~ebelp = it_mseg-ebelp.

******* START - CHANGES ON 06.11.2015
***        SELECT rsnum rspos aufnr
***               INTO TABLE it_zmmt001
***             FROM zmmt001
***               FOR ALL ENTRIES IN int_final
***             WHERE rsnum = int_final-rsnum
***               AND rspos = int_final-rspos
***               AND bwart = '281'
***               AND xloek = ''.
***
***        LOOP AT it_zmmt001.
***          e_network-network = it_zmmt001-aufnr.
***          APPEND e_network.
***        ENDLOOP.
***
***        SORT e_network BY network ASCENDING.
***        DELETE ADJACENT DUPLICATES FROM e_network.
***
***        CALL FUNCTION 'BAPI_NETWORK_GETINFO'
***          TABLES
***            i_network_list = e_network
***            e_component    = e_component.
***
***        LOOP AT e_component.
***          READ TABLE it_zmmt001 WITH KEY rsnum = e_component-reserv_no rspo
s = e_component-res_item.
***          IF sy-subrc NE 0.
***            DELETE e_component.
***          ENDIF.
***        ENDLOOP.
***
***        SELECT      eban~banfn eban~bnfpo eban~matnr eban~bednr
***                    ekpo~ebeln ekpo~ebelp
***                    FROM eban INNER JOIN ekpo ON ( eban~banfn EQ ekpo~banfn
 AND eban~bnfpo EQ ekpo~bnfpo )
***                    INTO TABLE it_ekpo
***                    FOR ALL ENTRIES IN e_component
***                    WHERE eban~banfn = e_component-preq_no
***                    AND eban~bnfpo = e_component-preq_item
***                    AND eban~loekz = ''
***                    AND ekpo~loekz = ''.
***        LOOP AT it_ekpo.
***          CLEAR e_component.
***          READ TABLE e_component WITH KEY preq_no = it_ekpo-banfn preq_item
 = it_ekpo-bnfpo.
***          IF sy-subrc EQ 0.
***            it_ekpo-rsnum = e_component-res_item.
***            it_ekpo-rspos = e_component-reserv_no .
***            MODIFY it_ekpo TRANSPORTING rsnum rspos.
***          ENDIF.
***        ENDLOOP.
***
***        SELECT  eban~matnr eban~bednr
***                ebkn~ps_psp_pnr
***                ekpo~ebeln ekpo~ebelp
***                FROM eban INNER JOIN ebkn ON ( eban~banfn EQ ebkn~banfn AND
 eban~bnfpo EQ ebkn~bnfpo )
```

```
***                          INNER JOIN ekpo ON ( eban~banfn EQ ekpo~banfn AND
 eban~bnfpo EQ ekpo~bnfpo )
***               INTO TABLE it_eban
***               FOR ALL ENTRIES IN int_final
***               WHERE eban~loekz = ''
***                 AND eban~matnr = int_final-zdesno
***                 AND eban~bednr = int_final-partno
***                 AND ebkn~loekz = ''
***                 AND ebkn~ps_psp_pnr = int_final-posnr
***                 AND ekpo~loekz = ''.
***
*********** FOR UPLOADED PRS WHERE WBS IS NOT ACOUNT ASSIGNED COMPARING WITH P
O WBS ELEMENT
***        SELECT      eban~matnr eban~bednr
***                     ekpo~ebeln ekpo~ebelp
***                     ekkn~ps_psp_pnr
***         FROM eban INNER JOIN ekpo ON ( eban~banfn EQ ekpo~banfn AND eban~b
nfpo EQ ekpo~bnfpo )
***                     INNER JOIN ekkn ON ( ekpo~ebeln EQ ekkn~ebelp AND ekpo~e
belp EQ ekkn~ebelp )
***         INTO TABLE it_ekkn
***         FOR ALL ENTRIES IN int_final
***         WHERE eban~loekz = ''
***           AND eban~matnr = int_final-zdesno
***           AND eban~bednr = int_final-partno
*****           AND ebkn~loekz = ''
*****           AND ebkn~ps_psp_pnr = int_final-posnr
***           AND ekpo~loekz = ''
***           AND ekkn~ps_psp_pnr = int_final-posnr.
***
*********** START -  FOR UPLOADED POs
***        DATA: BEGIN OF int_ekkn OCCURS 0,
***               ebeln      TYPE ekkn-ebeln,
***               ebelp      TYPE ekkn-ebelp,
***               ps_psp_pnr TYPE ekkn-ps_psp_pnr,
***             END OF int_ekkn.
***
***        DATA: BEGIN OF int_ekpo OCCURS 0,
***               ebeln      TYPE ekpo-ebeln,
***               ebelp      TYPE ekpo-ebelp,
***               matnr      TYPE ekpo-matnr,
***               menge      TYPE ekpo-menge,
***               meins      TYPE ekpo-meins,
***               banfn      TYPE ekpo-banfn,
***               bnfpo      TYPE ekpo-bnfpo,
***               ps_psp_pnr TYPE ekkn-ps_psp_pnr,
***             END OF int_ekpo.
***
***        SELECT ebeln ebelp ps_psp_pnr
***          FROM ekkn INTO TABLE int_ekkn                          "PO Items WBS
***          FOR ALL ENTRIES IN int_final
***          WHERE ps_psp_pnr = int_final-posnr
***          AND loekz = ''.
***         IF int_ekkn[] IS NOT INITIAL.
***           REFRESH int_ekpo.
***           SELECT ebeln ebelp matnr menge meins banfn bnfpo
***             FROM ekpo INTO TABLE int_ekpo                     "PO Items MATNR
***             FOR ALL ENTRIES IN int_ekkn
***             WHERE ebeln = int_ekkn-ebeln
***               AND ebelp = int_ekkn-ebelp
***               AND lgort EQ 'DD99'
***               AND loekz = ''.
***           LOOP AT int_ekpo.
***             READ TABLE int_ekkn WITH KEY ebeln = int_ekpo-ebeln ebelp = int
_ekpo-ebelp.
***             IF sy-subrc = 0.
***               MOVE int_ekkn-ps_psp_pnr TO int_ekpo-ps_psp_pnr.
***               MODIFY int_ekpo TRANSPORTING ps_psp_pnr.
***             ENDIF.
***           ENDLOOP.
***         ENDIF.
***
***         IF int_ekpo[] IS NOT INITIAL.
***           SELECT  mblnr mjahr bwart menge ebeln ebelp lfbja lfbnr sjahr
***                    FROM mseg
***                    INTO TABLE it_mseg_upload
***                    FOR ALL ENTRIES IN int_ekpo
***                    WHERE lgort = 'DD99'
***                      AND bwart IN ('101', '102','122','124')
***                      AND sobkz = 'Q'
***                      AND ebeln = int_ekpo-ebeln
***                      AND ebelp = int_ekpo-ebelp.
***
***
***           IF it_mseg_upload[] IS NOT INITIAL.
***             REFRESH : it_mseg102.
***             LOOP AT it_mseg_upload.
***               IF it_mseg_upload-bwart <> '101'.
***                 MOVE-CORRESPONDING it_mseg_upload TO it_mseg102.
***                 IF it_mseg102-sjahr IS INITIAL. " IF CONDITION ADDED ON 06.
11.2015
***                   it_mseg102-sjahr = it_mseg102-lfbja.
***                 ENDIF.
***                 APPEND it_mseg102.
***                 CLEAR  it_mseg102.
***                 DELETE it_mseg_upload.
***               ENDIF.
***             ENDLOOP.
***             LOOP AT it_mseg_upload.
***               READ TABLE it_mseg102 WITH KEY lfbnr = it_mseg_upload-mblnr
sjahr = it_mseg_upload-mjahr. "lfpos = int_mseg-zeile
***               IF sy-subrc = 0.
***                 DELETE it_mseg_upload.
***               ENDIF.
***             ENDLOOP.
***           ENDIF.
```

```
***
***        ENDIF.
***
********** END - FOR UPLOADED POs
***
***        IF it_ekpo[] IS NOT INITIAL.
***          SELECT  mblnr mjahr bwart menge ebeln ebelp lfbja lfbnr sjahr
***                     FROM mseg
***                     INTO TABLE it_mseg_ekpo
***                     FOR ALL ENTRIES IN it_ekpo
***                     WHERE lgort = 'DD99'
***                       AND bwart IN ('101', '102','122','124')
***                       AND sobkz = 'Q'
***                       AND ebeln = it_ekpo-ebeln
***                       AND ebelp = it_ekpo-ebelp.
***        ENDIF.
***
***        IF it_eban[] IS NOT INITIAL.
***          SELECT  mblnr mjahr bwart menge ebeln ebelp lfbja lfbnr sjahr
***            FROM mseg
***            INTO TABLE it_mseg
***            FOR ALL ENTRIES IN it_eban
***            WHERE lgort = 'DD99'
***              AND bwart IN ('101', '102','122','124')
***              AND sobkz = 'Q'
***              AND ebeln = it_eban-ebeln
***              AND ebelp = it_eban-ebelp.
***        ENDIF.
***
***        IF it_ekkn[] IS NOT INITIAL.
***          SELECT  mblnr mjahr bwart menge ebeln ebelp lfbja lfbnr sjahr
***            FROM mseg
***            INTO TABLE it_mseg_ekkn
***            FOR ALL ENTRIES IN it_ekkn
***            WHERE lgort = 'DD99'
***              AND bwart IN ('101', '102','122','124')
***              AND sobkz = 'Q'
***              AND ebeln = it_ekkn-ebeln
***              AND ebelp = it_ekkn-ebelp.
***
***        ENDIF.
***
***        IF it_mseg_ekpo[] IS NOT INITIAL.
***          REFRESH : it_mseg102.
***          LOOP AT it_mseg_ekpo.
***            IF it_mseg_ekpo-bwart <> '101'.
***              MOVE-CORRESPONDING it_mseg_ekpo TO it_mseg102.
***              IF it_mseg102-sjahr IS INITIAL. " IF CONDITION ADDED ON 06.11
.2015
***                it_mseg102-sjahr = it_mseg102-lfbja.
***              ENDIF.
***              APPEND it_mseg102.
***              CLEAR  it_mseg102.
***              DELETE it_mseg_ekpo.
***            ENDIF.
***          ENDLOOP.
***          LOOP AT it_mseg_ekpo.
***            READ TABLE it_mseg102 WITH KEY lfbnr = it_mseg_ekpo-mblnr  sjah
r = it_mseg_ekpo-mjahr. "lfpos = int_mseg-zeile
***            IF sy-subrc = 0.
***              DELETE it_mseg_ekpo.
***            ENDIF.
***          ENDLOOP.
***        ENDIF.
***
***        IF it_mseg[] IS NOT INITIAL.
***          REFRESH : it_mseg102.
***          LOOP AT it_mseg.
***            IF it_mseg-bwart <> '101'.
***              MOVE-CORRESPONDING it_mseg TO it_mseg102.
***              IF it_mseg102-sjahr IS INITIAL. " IF CONDITION ADDED ON 06.11
.2015
***                it_mseg102-sjahr = it_mseg102-lfbja.
***              ENDIF.
***              APPEND it_mseg102.
***              CLEAR  it_mseg102.
***              DELETE it_mseg.
***            ENDIF.
***          ENDLOOP.
***          LOOP AT it_mseg.
***            READ TABLE it_mseg102 WITH KEY lfbnr = it_mseg-mblnr  sjahr = i
t_mseg-mjahr. "lfpos = int_mseg-zeile
***            IF sy-subrc = 0.
***              DELETE it_mseg.
***            ENDIF.
***          ENDLOOP.
***        ENDIF.
***
***        IF it_mseg_ekkn[] IS NOT INITIAL.
***          REFRESH : it_mseg102.
***          LOOP AT it_mseg_ekkn.
***            IF it_mseg_ekkn-bwart <> '101'.
***              MOVE-CORRESPONDING it_mseg_ekkn TO it_mseg102.
***              IF it_mseg102-sjahr IS INITIAL. " IF CONDITION ADDED ON 06.11
.2015
***                it_mseg102-sjahr = it_mseg102-lfbja.
***              ENDIF.
***              APPEND it_mseg102.
***              CLEAR  it_mseg102.
***              DELETE it_mseg_ekkn.
***            ENDIF.
***          ENDLOOP.
***          LOOP AT it_mseg_ekkn.
***            READ TABLE it_mseg102 WITH KEY lfbnr = it_mseg_ekkn-mblnr  sjah
r = it_mseg_ekkn-mjahr. "lfpos = int_mseg-zeile
***            IF sy-subrc = 0.
***              DELETE it_mseg_ekkn.
***            ENDIF.
***          ENDLOOP.
***        ENDIF.
***
***        LOOP AT int_final WHERE dutype = 'DD' AND totfcwono IS INITIAL AND
status NE 'SCLD'.
***
***          CLEAR v_ddqty.
***
***          CLEAR it_ekpo.
***          READ TABLE it_ekpo WITH KEY rsnum = int_final-rsnum
***                                      rspos = int_final-rspos.
***          IF sy-subrc EQ 0.
***            LOOP AT it_mseg_ekpo WHERE ebeln = it_ekpo-ebeln AND ebelp = it
_ekpo-ebelp.
***              v_ddqty = v_ddqty + it_mseg_ekpo-menge.
***              CLEAR it_mseg_ekpo.
***            ENDLOOP.
***          ELSE.
***            CLEAR it_eban.
***            READ TABLE it_eban WITH KEY matnr = int_final-zdesno
***                                        bednr = int_final-partno
***                                        ps_psp_pnr = int_final-posnr.
***            IF sy-subrc EQ 0 .
***              LOOP AT it_mseg WHERE ebeln = it_eban-ebeln AND ebelp = it_eb
an-ebelp.
***                v_ddqty = v_ddqty + it_mseg-menge.
***                CLEAR it_mseg.
***              ENDLOOP.
***            ELSE.
***
***              CLEAR it_ekkn.
***              READ TABLE it_ekkn WITH KEY matnr = int_final-zdesno
***                                          bednr = int_final-partno
***                                          ps_psp_pnr = int_final-posnr.
***              IF sy-subrc EQ 0 .
***                LOOP AT it_mseg_ekkn WHERE ebeln = it_ekkn-ebeln AND ebelp
= it_ekkn-ebelp.
***                  v_ddqty = v_ddqty + it_mseg_ekkn-menge.
***                  CLEAR it_mseg_ekkn.
***                ENDLOOP.
***              ELSE.
***                CLEAR int_ekpo.
***                READ TABLE int_ekpo WITH KEY matnr = int_final-zdesno
***                                             ps_psp_pnr = int_final-posnr.
***                IF sy-subrc EQ 0 .
***                  LOOP AT it_mseg_upload WHERE ebeln = int_ekpo-ebeln AND e
belp = int_ekpo-ebelp.
***                    v_ddqty = v_ddqty + it_mseg_upload-menge.
***                    CLEAR it_mseg_upload.
***                  ENDLOOP.
***                ENDIF.
***              ENDIF.
***            ENDIF.
***          ENDIF.
***
***          IF int_final-menge GT v_ddqty.
***            int_final-totdesp = ''.
***            int_final-purpen = 1.
***          ELSE.
***            int_final-totdesp = 1.
***            int_final-purpen = ''.
***          ENDIF.
***          MODIFY int_final TRANSPORTING purpen totdesp.
***
***          CLEAR int_final.
***        ENDLOOP .
***      ENDIF.
******* END - CHANGES ON 06.11.2015


**      SORT it_mseg BY ps_psp_pnr matnr.
**
**      LOOP AT it_mseg.
**        CALL FUNCTION 'CONVERSION_EXIT_ABPSP_OUTPUT'
**          EXPORTING
**            input  = it_mseg-ps_psp_pnr
**          IMPORTING
**            output = it_mseg-wono.
**        CONCATENATE 'W-' it_mseg-wono+2(10) INTO it_mseg-wono.
**        CALL FUNCTION 'CONVERSION_EXIT_ABPSP_INPUT'
**          EXPORTING
**            input  = it_mseg-wono
**          IMPORTING
**            output = it_mseg-pspnr.
**        CLEAR it_eban.
**        READ TABLE it_eban WITH KEY ebeln  = it_mseg-ebeln ebelp = it_mseg-e
belp.
**        IF sy-subrc EQ 0.
**          it_mseg-bednr = it_eban-bednr.
**        ENDIF.
**        MODIFY it_mseg TRANSPORTING wono pspnr bednr.
**        AT NEW matnr.
**          SUM.
**          MODIFY it_mseg TRANSPORTING menge.
**          CONTINUE.
**        ENDAT.
**        DELETE it_mseg.
**      ENDLOOP.

**      LOOP AT int_final WHERE dutype = 'DD' AND totfcwono IS INITIAL.
**        CLEAR it_mseg.
**        READ TABLE it_mseg WITH KEY matnr = int_final-zdesno ps_psp_pnr = in
t_final-posnr bednr = int_final-partno."pspnr = int_final-pspnr."for partial c
ases checking at PGMA level
```

```
**        IF sy-subrc = 0.
**          IF it_mseg-menge GE int_final-menge.
**            int_final-totdesp = 1.
**            int_final-purpen = ''.
**            MODIFY int_final TRANSPORTING purpen totdesp.
**          ELSE.
**            int_final-totdesp = ''.
**            int_final-purpen = 1.
**            MODIFY int_final TRANSPORTING purpen totdesp.
**          ENDIF.
**        ELSE.
**          CLEAR it_mseg.
**          READ TABLE it_mseg WITH KEY matnr = int_final-zdesno ps_psp_pnr =
int_final-posnr.
**          IF it_mseg-menge GE int_final-menge.
**            int_final-totdesp = 1.
**            int_final-purpen = ''.
**            MODIFY int_final TRANSPORTING purpen totdesp.
**          ELSE.
**            int_final-totdesp = ''.
**            int_final-purpen = 1.
**            MODIFY int_final TRANSPORTING purpen totdesp.
**          ENDIF.
**        ENDIF.
**      ENDLOOP.

      " End of Correcting Purchase Pendency from MSEG table data


**<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<    >>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>>>***
****************  START - LOGIC CONSOLIDATION FOR PURCHASE PENDENCY - 07.11.20
15

      IF int_final[] IS NOT INITIAL.

*******   GET SYSTEM GENERATED PRs USING RESERATION DATA
        REFRESH : e_network,e_component ,it_eban , it_ekpo , it_ekkn , int_pr ,
 int_po.

        LOOP AT int_final WHERE aufnr IS NOT INITIAL.
          e_network-network = int_final-aufnr.
          APPEND e_network.
        ENDLOOP.

        SORT e_network BY network ASCENDING.
        DELETE ADJACENT DUPLICATES FROM e_network.

**        CALL FUNCTION 'BAPI_NETWORK_GETINFO'
**          TABLES
**            i_network_list = e_network
**            e_component    = e_component.
```

```
        LOOP AT e_network.        " Added by Siva 15.12.2020

          CALL FUNCTION 'BAPI_NETWORK_COMP_GETDETAIL'
            EXPORTING
              number              = e_network-network
*             MAX_ROWS             = 0
* IMPORTING
*             RETURN               =
            TABLES
*             I_ACTIVITY_RANGE     =
*             I_COMPONENTS_ID      =
              e_components_detail = e_component1.


          APPEND LINES OF e_component1 TO e_component.

          CLEAR e_network.
          REFRESH : e_component1 , e_component1[].
        ENDLOOP.

        LOOP AT e_component.
          READ TABLE int_final WITH KEY rsnum = e_component-reserv_no rspos = e
_component-res_item.
          IF sy-subrc NE 0.
            DELETE e_component.
          ENDIF.
        ENDLOOP.

***** EBAN , EBKN JOIN TO GET PRs WHEREVER ACCOUNT ASSIGNMENT IS THERE

        SELECT eban~banfn eban~bnfpo eban~matnr  eban~bednr ebkn~ps_psp_pnr eba
n~werks
               FROM eban INNER JOIN ebkn ON ( eban~banfn EQ ebkn~banfn AND eban
~bnfpo EQ ebkn~bnfpo )
               INTO CORRESPONDING FIELDS OF TABLE it_eban
               FOR ALL ENTRIES IN int_final
               WHERE eban~loekz EQ ''
                AND  eban~matnr EQ int_final-zdesno
                AND  ebkn~ps_psp_pnr EQ int_final-posnr.
        SORT it_eban BY banfn bnfpo.

*        IF it_eban[] IS NOT INITIAL.
*          SELECT * FROM zmmt170 INTO TABLE it_zmmt170 FOR ALL ENTRIES IN it_e
ban WHERE werks = it_eban-werks.
*        ENDIF.

        LOOP AT int_final.  " FILLING PR DETAILS IN THE FINAL TABLE
          CLEAR e_component.
          READ TABLE e_component WITH KEY reserv_no = int_final-rsnum res_item
= int_final-rspos.
          IF sy-subrc EQ 0 . "AND e_component-preq_no IS NOT INITIAL.
            int_final-banfn = e_component-preq_no.
            int_final-bnfpo = e_component-preq_item.
```

```
*             READ TABLE it_zmmt170 WITH KEY werks = e_component-plant.
*             IF sy-subrc = 0.
*               int_final-pur_off = it_zmmt170-pernr.
*               CLEAR: w_name.
*               SELECT SINGLE ename FROM pa0001 INTO w_name WHERE pernr = it_zm
mt170-pernr.
*               int_final-pur_name = w_name.
*           ENDIF.

          ELSE.
            CLEAR it_eban.
            READ TABLE it_eban WITH KEY matnr       = int_final-zdesno
                                        bednr       = int_final-partno
                                        ps_psp_pnr  = int_final-posnr.
            IF sy-subrc EQ 0.
              int_final-banfn = it_eban-banfn.
              int_final-bnfpo = it_eban-bnfpo.
*               READ TABLE it_zmmt170 WITH KEY werks = it_eban-werks.
*               IF sy-subrc = 0.
*                 int_final-pur_off = it_zmmt170-pernr.
*                 CLEAR: w_name.
*                 SELECT SINGLE ename FROM pa0001 INTO w_name WHERE pernr = it_
zmmt170-pernr.
*                 int_final-pur_name = w_name.
*               ENDIF.
            ELSE.
              CLEAR it_eban.
              READ TABLE it_eban WITH KEY matnr       = int_final-zdesno
                                          ps_psp_pnr  = int_final-posnr.
              IF sy-subrc EQ 0.
                int_final-banfn = it_eban-banfn.
                int_final-bnfpo = it_eban-bnfpo.

*                 READ TABLE it_zmmt170 WITH KEY werks = it_eban-werks.
*                 IF sy-subrc = 0.
*                   int_final-pur_off = it_zmmt170-pernr.
*                   CLEAR: w_name.
*                   SELECT SINGLE ename FROM pa0001 INTO w_name WHERE pernr = i
t_zmmt170-pernr.
*                   int_final-pur_name = w_name.
*                 ENDIF.
              ENDIF.
            ENDIF.
          ENDIF.
          IF int_final-banfn IS NOT INITIAL.
            int_pr-banfn = int_final-banfn.
            int_pr-bnfpo = int_final-bnfpo.
            APPEND int_pr.
            CLEAR int_pr.
          ENDIF.
          MODIFY int_final TRANSPORTING banfn bnfpo." pur_off pur_name.
          CLEAR int_final.
```

```
        ENDLOOP.

****** GETTING POs FROM PRs
        IF int_pr[] IS NOT INITIAL.
          SELECT ekpo~ebeln ekpo~ebelp ekpo~banfn ekpo~bnfpo ekko~zzpurexec  IN
TO TABLE it_ekpo
            FROM ekpo
            JOIN ekko ON  ( ekpo~ebeln EQ ekko~ebeln )
                FOR ALL ENTRIES IN int_pr
                WHERE ekpo~loekz = ''
                  AND ekpo~banfn = int_pr-banfn
                  AND ekpo~bnfpo = int_pr-bnfpo.
        ENDIF.
******** GETTING POs WITHOUT PRs ( DIRECT UPLOADED POs )
        SELECT ekpo~ebeln ekpo~ebelp ekpo~matnr ekpo~bednr ekkn~ps_psp_pnr ekpo
~werks ekko~zzpurexec INTO TABLE it_ekkn
               FROM ekpo
            JOIN ekkn ON ( ekpo~ebeln EQ ekkn~ebeln AND ekpo~ebelp EQ ekkn~ebel
p )
            JOIN ekko ON  ( ekpo~ebeln EQ ekko~ebeln )
               FOR ALL ENTRIES IN int_final
               WHERE ekpo~loekz = ''
                AND  ekpo~matnr = int_final-zdesno
                AND  ekkn~loekz = ''
                AND  ekkn~ps_psp_pnr = int_final-posnr.

        SORT it_ekkn BY ebeln ebelp.

        LOOP AT int_final.  " FILLING PO DETAILS IN THE FINAL TABLE
          IF int_final-banfn IS NOT INITIAL.
            CLEAR : it_ekpo.
            READ TABLE it_ekpo WITH KEY banfn = int_final-banfn
                                        bnfpo = int_final-bnfpo.
            IF sy-subrc EQ 0.
              int_final-ebeln = it_ekpo-ebeln.
              int_final-ebelp = it_ekpo-ebelp.
              int_final-pur_off = it_ekpo-zzpurexec.
            ENDIF.
          ELSE.
            CLEAR : it_ekkn.
            READ TABLE it_ekkn WITH KEY matnr       = int_final-zdesno
                                        ps_psp_pnr  = int_final-posnr.
            IF sy-subrc EQ 0.
              int_final-ebeln = it_ekkn-ebeln.
              int_final-ebelp = it_ekkn-ebelp.
              int_final-ebelp = it_ekkn-ebelp.
              int_final-pur_off = it_ekkn-zzpurexec.
            ENDIF.
          ENDIF.

          IF int_final-ebeln IS NOT INITIAL.
            int_po-ebeln = int_final-ebeln.
            int_po-ebelp = int_final-ebelp.
```

```
              APPEND int_po . CLEAR int_po.

            ENDIF.
            CLEAR: w_submi,w_pernr.
            IF int_final-pur_off IS INITIAL.
              SELECT SINGLE submi FROM zmmt004 INTO w_submi WHERE banfn = int_fin
al-banfn AND bnfpo = int_final-bnfpo.
              IF w_submi IS NOT INITIAL.
                SELECT SINGLE gen_by FROM zmmt003 INTO w_pernr WHERE subm i = w_su
bmi.
                int_final-pur_off = w_pernr.
              ENDIF.

*                READ TABLE it_zmmt170 WITH KEY werks = int_final-werks.
*                IF sy-subrc = 0.
*                  int_final-pur_off = it_zmmt170-pernr.
*                ENDIF.
            ENDIF.

            CLEAR: w_name.
            SELECT SINGLE ename FROM pa0001 INTO w_name WHERE pernr = int_final-p
ur_off.
            int_final-pur_name = w_name.
            MODIFY int_final TRANSPORTING ebeln ebelp pur_off pur_name.
            CLEAR int_final .
          ENDLOOP.


***********  FROM PO DATA GET THE RDR DETAILS FROM MSEG
          IF int_po[] IS NOT INITIAL.
            REFRESH : it_mseg.
            SELECT  mblnr mjahr bwart menge ebeln ebelp lfbja lfbnr sjahr
                        FROM mseg
                        INTO TABLE it_mseg
                        FOR ALL ENTRIES IN int_po
                        WHERE lgort = 'DD99'
                          AND bwart IN ('101', '102','122','124')
                          AND sobkz = 'Q'
                          AND ebeln = int_po-ebeln
                          AND ebelp = int_po-ebelp.
          ENDIF.

          IF it_mseg[] IS NOT INITIAL.
            REFRESH : it_mseg102.
            LOOP AT it_mseg.
              IF it_mseg-bwart <> '101'.
                MOVE-CORRESPONDING it_mseg TO it_mseg102.
                IF it_mseg102-sjahr IS INITIAL.
                  it_mseg102-sjahr = it_mseg102-lfbja.
                ENDIF.
                APPEND it_mseg102.
                CLEAR  it_mseg102.
                DELETE it_mseg.
              ENDIF.
            ENDLOOP.
            LOOP AT it_mseg.
              READ TABLE it_mseg102 WITH KEY lfbnr = it_mseg-mblnr  sjahr = it_ms
  eg-mjahr.
              IF sy-subrc = 0.
                DELETE it_mseg.
              ENDIF.
            ENDLOOP.
          ENDIF.


  ********* UPDATING PURCHASE PENDENCY

          LOOP AT  int_final WHERE dutype = 'DD' AND totfcwono IS INITIAL AND sta
  tus NE 'SCLD'.
            IF int_final-status = 'CDCB'.
              int_final-totdesp = 1.
              int_final-purpen = ''.
            ELSE.
              CLEAR v_ddqty.
              LOOP AT it_mseg WHERE ebeln = int_final-ebeln AND ebelp = int_final
  -ebelp.
                v_ddqty = v_ddqty + it_mseg-menge.
                CLEAR it_mseg.
              ENDLOOP.

              IF int_final-menge GT v_ddqty.
                int_final-totdesp = ''.
                int_final-purpen = 1.
              ELSE.
                int_final-totdesp = 1.
                int_final-purpen = ''.
              ENDIF.
            ENDIF.
            MODIFY int_final TRANSPORTING purpen totdesp.

            CLEAR int_final.
          ENDLOOP.



        ENDIF.

  **<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<< END - 07.11.2015   >>>>>>>>>>>>>>>>>>
  >>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>***




  ************************Customer Name, Project Name etc. ********************
  **
```

```
      CLEAR: rep_dt.
      CALL FUNCTION 'CONVERT_DATE_TO_EXTERNAL'
        EXPORTING
          date_internal = sy-datum
        IMPORTING
          date_external = rep_dt.
      IF s_pspnr IS NOT INITIAL AND int_final[] IS NOT INITIAL.
*****        LOOP AT int_final.
*****          CLEAR p_pspid.
*****
*****          CALL FUNCTION 'CONVERSION_EXIT_ABPSP_OUTPUT'
*****            EXPORTING
*****              input  = int_final-pspid
*****            IMPORTING
*****              output = p_pspid.
*****
*****
*****          REPLACE ALL OCCURRENCES OF '-' IN p_pspid WITH ''.
*****          int_final-vbeln = p_pspid.
*****          MODIFY int_final TRANSPORTING vbeln.
*****        ENDLOOP.

        break 1896563.

        SELECT vbeln adrnr
          INTO TABLE it_vbpa_adrnr
        FROM vbpa
          FOR ALL ENTRIES IN int_final
        WHERE vbeln = int_final-vbeln.

        IF it_vbpa_adrnr[] IS NOT INITIAL.
          SELECT addrnumber name1
            INTO TABLE it_adrc
          FROM adrc
            FOR ALL ENTRIES IN it_vbpa_adrnr
          WHERE addrnumber = it_vbpa_adrnr-adrnr.
        ENDIF.

        SELECT posnr up
          INTO TABLE it_prhi_pname
        FROM prhi
          FOR ALL ENTRIES IN int_final
        WHERE posnr = int_final-pspid.

        IF it_prhi_pname[] IS NOT INITIAL.
          SELECT pspnr post1
            INTO TABLE it_prps_pname
          FROM prps
            FOR ALL ENTRIES IN it_prhi_pname
          WHERE pspnr = it_prhi_pname-up.
        ENDIF.

      ELSE.
```

```
        CLEAR: t_adrnr, t_name1, t_projid, t_projname.

        SELECT SINGLE adrnr INTO t_adrnr                    "Customer Name
          FROM vbpa
          WHERE vbeln = p_vbeln AND parvw = 'AG'.

        SELECT SINGLE name1 INTO t_name1
          FROM adrc
          WHERE addrnumber = t_adrnr.


        SELECT SINGLE up INTO t_projid           "Project Name
          FROM prhi
          WHERE posnr = p_pspidin.

        IF sy-subrc = 0.
          SELECT SINGLE post1 INTO t_projname
            FROM prps
            WHERE pspnr = t_projid.
        ENDIF.
      ENDIF.

*************************Customer Name, Project Name etc. ********************
**

      SORT int_final BY werks pspnr lgort.

      CLEAR: mpctot, mpcpen, ppctot, ppcpen, subtot, subpen, purtot, purpen, st
rpen, notcdc, gpissu, cdcack, cmlpen, cdcpak, totdesp, notcdc, totfcwono,cdcpq.

*     get the pgam description
      IF int_final[] IS NOT INITIAL.
        SELECT pspnr post1 FROM prps
          INTO CORRESPONDING FIELDS OF TABLE int_prpspost1
          FOR ALL ENTRIES IN int_final
          WHERE pspnr = int_final-posnr.
      ENDIF.

      SORT int_prpspost1 ASCENDING BY pspnr.


***********to get MDCC against OBD item addded by 6105483*********************
*****

      IF int_final[] IS NOT INITIAL.

        SELECT vbeln posnr qplos FROM lips
          INTO TABLE int_lips1
          FOR ALL ENTRIES IN int_final
             WHERE vbeln = int_final-deliv AND posnr = int_final-delin.

        IF int_lips1[] IS NOT INITIAL.
```

```
          SELECT prueflos zzmdcc ls_vbeln ls_posnr FROM qals
            INTO TABLE int_qals
             FOR ALL ENTRIES IN int_lips1
              WHERE prueflos = int_lips1-qplos.

        ENDIF.
        IF int_qals[] IS NOT INITIAL.
          SELECT prueflos  FROM qave
            INTO TABLE int_qave
             FOR ALL ENTRIES IN int_qals
              WHERE prueflos = int_qals-prueflos AND vcode = 'AA'.
        ENDIF.


      ENDIF.


********************************************************************************
******


      LOOP AT int_final.                               "Final data preparation

        CLEAR int_prpspost1.
        READ TABLE int_prpspost1 WITH KEY pspnr = int_final-posnr BINARY SEARCH
.
        IF sy-subrc EQ 0.
          int_final-post1 = int_prpspost1-post1.
          MODIFY int_final TRANSPORTING post1.
        ENDIF.

        READ TABLE  int_lips1 WITH KEY vbeln = int_final-deliv  posnr = int_fin
al-delin.
        IF sy-subrc = 0.
          READ TABLE int_qals WITH KEY   prueflos = int_lips1-qplos.
          IF sy-subrc EQ 0.
            int_final-zzmdcc = int_qals-zzmdcc.

            MODIFY int_final.

          ENDIF.
        ENDIF.
        temp_vbeln = int_final-vbeln.
        mpctot = mpctot + int_final-mpctot.
        mpcpen = mpcpen + int_final-mpcpen.
        ppctot = ppctot + int_final-ppctot.
        ppcpen = ppcpen + int_final-ppcpen.
        subtot = subtot + int_final-subtot.
        subpen = subpen + int_final-subpen.
        purtot = purtot + int_final-purtot.
        purpen = purpen + int_final-purpen.
        strpen = strpen + int_final-strpen.
        gpissu = gpissu + int_final-gpissu.
        cdcack = cdcack + int_final-cdcack.
        cmlpen = cmlpen + int_final-cmlpen.
        cdcpq =  cdcpq + int_final-cdcpq.
        cdcpak = cdcpak + int_final-cdcpak.
        totdesp = totdesp + int_final-totdesp.
        totfcwono = totfcwono + int_final-totfcwono.
        IF s_pspnr IS INITIAL.

          AT END OF werks.
*           SUM.
            int_count-werks = int_final-werks.
            " Logic for deriving Division
            IF int_final-werks = 'P001'.
              int_count-divis = 'TC'.
            ELSEIF int_final-werks = 'P002'.
              int_count-divis = 'EM'.
            ELSEIF int_final-werks = 'P003'.
              int_count-divis = 'SG'.
            ELSEIF int_final-werks = 'P004'.
              int_count-divis = 'FOUNDRY'.
            ELSEIF int_final-werks = 'P006'.
              int_count-divis = 'HE & F'.
            ELSEIF int_final-werks = 'P007'.
              int_count-divis = 'TOOLS'.
            ELSEIF int_final-werks = 'P010'.
              int_count-divis = 'SPARES'.
            ELSEIF int_final-werks = 'P011'.
              int_count-divis = 'OR'.
            ELSEIF int_final-werks = 'P051'.
              int_count-divis = 'BM'.
            ELSEIF int_final-werks = 'P055'.
              int_count-divis = 'GT'.
            ELSEIF int_final-werks = 'P070'.
              int_count-divis = 'PUMPS'.
            ELSEIF int_final-werks = 'P090'.
              int_count-divis = 'PED'.
            ELSEIF int_final-werks+0(4) = 'P099'.
*             INT_COUNT-DIVIS = 'CMM'.
              CONCATENATE 'CMM' int_final-werks+4(5) INTO int_count-divis.
            ENDIF.
            int_count-mpctot = mpctot.
            int_count-mpcpen = mpcpen.
            int_count-ppctot = ppctot.
            int_count-ppcpen = ppcpen.
            int_count-subtot = subtot.
            int_count-subpen = subpen.
            int_count-purtot = purtot.
            int_count-purpen = purpen.
            int_count-strpen = strpen.
            int_count-notcdc = notcdc.
            int_count-gpissu = gpissu.
            int_count-cdcack = cdcack.
            int_count-cmlpen = cmlpen.
            int_count-cdcpq = cdcpq.
            int_count-cdcpak = cdcpak.
            int_count-totdesp = totdesp.
            int_count-totfcwono = totfcwono.
            int_count-tottot = mpctot + ppctot + subtot + purtot.
            int_count-notcdc = mpcpen + ppcpen + subpen + purpen + strpen + gpi
ssu.
            int_count-totpen = int_count-notcdc + cdcpq + cdcack + cmlpen + cdc
pak.
            APPEND int_count.
            CLEAR int_count.
            CLEAR: mpctot, mpcpen, ppctot, ppcpen, subtot, subpen, purtot, purp
en, strpen, notcdc, gpissu, cdcack, cmlpen, cdcpak, totdesp, notcdc,totfcwono ,
cdcpq.
          ENDAT.
        ELSE.
          AT END OF pspnr.
*           SUM.
            int_count-werks = int_final-werks.
            int_count-pspnr = int_final-pspnr.
            int_count-pspid = int_final-pspid.

            CLEAR it_vbpa_adrnr.
            READ TABLE it_vbpa_adrnr WITH KEY vbeln = temp_vbeln.

            CLEAR it_adrc.
            READ TABLE it_adrc WITH KEY addrnumber = it_vbpa_adrnr-adrnr.
            int_count-name1 = it_adrc-name1.

            CLEAR it_prhi_pname.
            READ TABLE it_prhi_pname WITH KEY posnr = int_final-pspid.

            CLEAR it_prps_pname.
            READ TABLE it_prps_pname WITH KEY pspnr = it_prhi_pname-up.
            int_count-post1 = it_prps_pname-post1.

            " Logic for deriving Division
            IF int_final-werks = 'P001'.
              int_count-divis = 'TC'.
            ELSEIF int_final-werks = 'P002'.
              int_count-divis = 'EM'.
            ELSEIF int_final-werks = 'P003'.
              int_count-divis = 'SG'.
            ELSEIF int_final-werks = 'P004'.
              int_count-divis = 'FOUNDRY'.
            ELSEIF int_final-werks = 'P006'.
              int_count-divis = 'HE & F'.
            ELSEIF int_final-werks = 'P007'.
              int_count-divis = 'TOOLS'.
            ELSEIF int_final-werks = 'P010'.
              int_count-divis = 'SPARES'.
            ELSEIF int_final-werks = 'P011'.
              int_count-divis = 'OR'.
            ELSEIF int_final-werks = 'P051'.
              int_count-divis = 'BM'.
            ELSEIF int_final-werks = 'P055'.
              int_count-divis = 'GT'.
            ELSEIF int_final-werks = 'P070'.
              int_count-divis = 'PUMPS'.
            ELSEIF int_final-werks = 'P090'.
              int_count-divis = 'PED'.
            ELSEIF int_final-werks+0(4) = 'P099'.
*             INT_COUNT-DIVIS = 'CMM'.
              CONCATENATE 'CMM' int_final-werks+4(5) INTO int_count-divis.
            ENDIF.
            int_count-mpctot = mpctot.
            int_count-mpcpen = mpcpen.
            int_count-ppctot = ppctot.
            int_count-ppcpen = ppcpen.
            int_count-subtot = subtot.
            int_count-subpen = subpen.
            int_count-purtot = purtot.
            int_count-purpen = purpen.
            int_count-strpen = strpen.
            int_count-notcdc = notcdc.
            int_count-gpissu = gpissu.
            int_count-cdcack = cdcack.
            int_count-cmlpen = cmlpen.
            int_count-cdcpq = cdcpq.
            int_count-cdcpak = cdcpak.
            int_count-totdesp = totdesp.
            int_count-totfcwono = totfcwono.
            int_count-tottot = mpctot + ppctot + subtot + purtot.
            int_count-notcdc = mpcpen + ppcpen + subpen + purpen + strpen + gpi
ssu.
            int_count-totpen = int_count-notcdc + cdcack + cmlpen + cdcpak + cd
cpq.
            APPEND int_count.
            CLEAR int_count.
            CLEAR: mpctot, mpcpen, ppctot, ppcpen, subtot, subpen, purtot, purp
en, strpen, notcdc, gpissu, cdcack, cmlpen, cdcpak, totdesp, notcdc,totfcwono,c
dcpq.
          ENDAT.
        ENDIF.
      ENDLOOP.

      CLEAR: mpctot, mpcpen, ppctot, ppcpen, subtot, subpen, purtot, purpen, st
rpen, notcdc, gpissu, cdcack, cmlpen, cdcpak, totdesp, notcdc, tottot, totpen,t
otfcwono,cdcpq.
      LOOP AT int_count.
        mpctot = mpctot + int_count-mpctot.
        mpcpen = mpcpen + int_count-mpcpen.
        ppctot = ppctot + int_count-ppctot.
        ppcpen = ppcpen + int_count-ppcpen.
        subtot = subtot + int_count-subtot.
        subpen = subpen + int_count-subpen.
        purtot = purtot + int_count-purtot.
        purpen = purpen + int_count-purpen.
        strpen = strpen + int_count-strpen.
        notcdc = notcdc + int_count-notcdc.
        gpissu = gpissu + int_count-gpissu.
        cdcack = cdcack + int_count-cdcack.
        cmlpen = cmlpen + int_count-cmlpen.
        cdcpq = cdcpq + int_count-cdcpq.
        cdcpak = cdcpak + int_count-cdcpak.
        tottot = tottot + int_count-tottot.
        totpen = totpen + int_count-totpen.
        totdesp = totdesp + int_count-totdesp.
        totfcwono = totfcwono + int_count-totfcwono.
      ENDLOOP.
      int_count-werks = ''.
      int_count-divis = 'TOTAL :'.
      int_count-pspnr = ''.
      int_count-name1 = ''.
      int_count-post1 = ''.
      int_count-pspid = ''.
      int_count-mpctot = mpctot.
      int_count-mpcpen = mpcpen.
      int_count-ppctot = ppctot.
      int_count-ppcpen = ppcpen.
      int_count-subtot = subtot.
      int_count-subpen = subpen.
      int_count-purtot = purtot.
      int_count-purpen = purpen.
      int_count-strpen = strpen.
      int_count-notcdc = notcdc.
      int_count-gpissu = gpissu.
      int_count-cdcack = cdcack.
      int_count-cmlpen = cmlpen.
      int_count-cdcpq = cdcpq.
      int_count-cdcpak = cdcpak.
      int_count-tottot = tottot.
      int_count-totpen = totpen.
      int_count-totdesp = totdesp.
      int_count-totfcwono = totfcwono.
      APPEND int_count.
      CLEAR int_count.

    ENDIF.
  ELSE.
    IF sy-batch NE 'X'.
      MESSAGE 'No records found for given Sales Order' TYPE 'E'.
    ENDIF.

  ENDIF.

ENDFORM.                    "GET_DATA
```

```
*&---------------------------------------------------------------------*
*&      Form  BUILD_CATALOG
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
FORM build_catalog.

  CLEAR wa_fieldcat.
  wa_fieldcat-tabname = 'INT_COUNT'.
  wa_fieldcat-fieldname = 'WERKS'.
  wa_fieldcat-outputlen =  10.
*  WA_FIELDCAT-COL_POS      = 0.
  wa_fieldcat-key          = 'X'.
  wa_fieldcat-seltext_l = 'Plant'.
  APPEND wa_fieldcat TO int_fieldcat.

  CLEAR wa_fieldcat.
  wa_fieldcat-tabname = 'INT_COUNT'.
  wa_fieldcat-fieldname = 'DIVIS'.
  wa_fieldcat-outputlen =  10.
*  WA_FIELDCAT-COL_POS      = 0.
  wa_fieldcat-key          = 'X'.
*  WA_FIELDCAT-HOTSPOT = 'X'.
*  WA_FIELDCAT-SELTEXT_M   = 'S O '.
  wa_fieldcat-seltext_l = 'Division'.
  APPEND wa_fieldcat TO int_fieldcat.

  IF s_pspnr IS NOT INITIAL.
    CLEAR wa_fieldcat.
    wa_fieldcat-tabname = 'INT_COUNT'.
    wa_fieldcat-fieldname = 'PSPNR'.
    wa_fieldcat-outputlen =  12.
*  WA_FIELDCAT-COL_POS      = 0.
    wa_fieldcat-key          = 'X'.
*  WA_FIELDCAT-HOTSPOT = 'X'.
*  WA_FIELDCAT-SELTEXT_M   = 'S O '.
    wa_fieldcat-seltext_l = 'Workorder'.
    APPEND wa_fieldcat TO int_fieldcat.

    CLEAR wa_fieldcat.
    wa_fieldcat-tabname = 'INT_COUNT'.
    wa_fieldcat-fieldname = 'PSPID'.
    wa_fieldcat-outputlen =  15.
*  WA_FIELDCAT-COL_POS      = 0.
    wa_fieldcat-key          = 'X'.
*  WA_FIELDCAT-HOTSPOT = 'X'.
*  WA_FIELDCAT-SELTEXT_M   = 'S O '.
    wa_fieldcat-seltext_l = 'Sales Order'.
    APPEND wa_fieldcat TO int_fieldcat.


  ENDIF.
```

```
CLEAR wa_fieldcat.
wa_fieldcat-tabname = 'INT_COUNT'.
wa_fieldcat-fieldname = 'MPCTOT'.
wa_fieldcat-outputlen =  8.
wa_fieldcat-seltext_l = 'MPC Total'.
APPEND wa_fieldcat TO int_fieldcat.

CLEAR wa_fieldcat.
wa_fieldcat-tabname = 'INT_COUNT'.
wa_fieldcat-fieldname = 'MPCPEN'.
wa_fieldcat-outputlen =  8.
wa_fieldcat-seltext_l = 'MPC Pending'.
APPEND wa_fieldcat TO int_fieldcat.

CLEAR wa_fieldcat.
wa_fieldcat-tabname = 'INT_COUNT'.
wa_fieldcat-fieldname = 'PPCTOT'.
wa_fieldcat-outputlen =  8.
wa_fieldcat-seltext_l = 'PPC Total'.
APPEND wa_fieldcat TO int_fieldcat.

CLEAR wa_fieldcat.
wa_fieldcat-tabname = 'INT_COUNT'.
wa_fieldcat-fieldname = 'PPCPEN'.
wa_fieldcat-outputlen =  8.
wa_fieldcat-seltext_l = 'PPC Pending'.
APPEND wa_fieldcat TO int_fieldcat.

CLEAR wa_fieldcat.
wa_fieldcat-tabname = 'INT_COUNT'.
wa_fieldcat-fieldname = 'SUBTOT'.
wa_fieldcat-outputlen =  8.
wa_fieldcat-seltext_l = 'SUBC Total'.
APPEND wa_fieldcat TO int_fieldcat.

CLEAR wa_fieldcat.
wa_fieldcat-tabname = 'INT_COUNT'.
wa_fieldcat-fieldname = 'SUBPEN'.
wa_fieldcat-outputlen =  8.
wa_fieldcat-seltext_l = 'SUBC Pending'.
APPEND wa_fieldcat TO int_fieldcat.

CLEAR wa_fieldcat.
wa_fieldcat-tabname = 'INT_COUNT'.
wa_fieldcat-fieldname = 'PURTOT'.
wa_fieldcat-outputlen =  8.
wa_fieldcat-seltext_l = 'PUR Total'.
APPEND wa_fieldcat TO int_fieldcat.

CLEAR wa_fieldcat.
wa_fieldcat-tabname = 'INT_COUNT'.
wa_fieldcat-fieldname = 'PURPEN'.
wa_fieldcat-outputlen =  8.
```

```
wa_fieldcat-seltext_l = 'PUR Pending'.
APPEND wa_fieldcat TO int_fieldcat.

CLEAR wa_fieldcat.
wa_fieldcat-tabname = 'INT_COUNT'.
wa_fieldcat-fieldname = 'STRPEN'.
wa_fieldcat-outputlen =  8.
wa_fieldcat-seltext_l = 'Stores Pending'.
APPEND wa_fieldcat TO int_fieldcat.

CLEAR wa_fieldcat.
wa_fieldcat-tabname = 'INT_COUNT'.
wa_fieldcat-fieldname = 'GPISSU'.
wa_fieldcat-outputlen =  8.
wa_fieldcat-seltext_l = 'GP Issued'.
APPEND wa_fieldcat TO int_fieldcat.

CLEAR wa_fieldcat.
wa_fieldcat-tabname = 'INT_COUNT'.
wa_fieldcat-fieldname = 'NOTCDC'.
wa_fieldcat-outputlen =  8.
wa_fieldcat-seltext_l = 'Exclu CDC'.
APPEND wa_fieldcat TO int_fieldcat.

CLEAR wa_fieldcat.
wa_fieldcat-tabname = 'INT_COUNT'.
wa_fieldcat-fieldname = 'CDCACK'.
wa_fieldcat-outputlen =  8.
wa_fieldcat-seltext_l = 'Packing Pendency by CDC'.
APPEND wa_fieldcat TO int_fieldcat.

CLEAR wa_fieldcat.
wa_fieldcat-tabname = 'INT_COUNT'.
wa_fieldcat-fieldname = 'CMLPEN'.
wa_fieldcat-outputlen =  8.
wa_fieldcat-seltext_l = 'Commercial Pendency'.
APPEND wa_fieldcat TO int_fieldcat.

CLEAR wa_fieldcat.
wa_fieldcat-tabname = 'INT_COUNT'.
wa_fieldcat-fieldname = 'CDCPQ'.
wa_fieldcat-outputlen =  8.
wa_fieldcat-seltext_l = 'CDC Packed Pend Qual'.
APPEND wa_fieldcat TO int_fieldcat.

CLEAR wa_fieldcat.
wa_fieldcat-tabname = 'INT_COUNT'.
wa_fieldcat-fieldname = 'CDCPAK'.
wa_fieldcat-outputlen =  8.
wa_fieldcat-seltext_l = 'CDC Packed Qual Clrd'.
APPEND wa_fieldcat TO int_fieldcat.

CLEAR wa_fieldcat.
```

```
  wa_fieldcat-tabname = 'INT_COUNT'.
  wa_fieldcat-fieldname = 'TOTTOT'.
  wa_fieldcat-outputlen =  8.
  wa_fieldcat-seltext_l = 'Total Deliverables'.
  APPEND wa_fieldcat TO int_fieldcat.

  CLEAR wa_fieldcat.
  wa_fieldcat-tabname = 'INT_COUNT'.
  wa_fieldcat-fieldname = 'TOTPEN'.
  wa_fieldcat-outputlen =  8.
  wa_fieldcat-seltext_l = 'Total Pending'.
  APPEND wa_fieldcat TO int_fieldcat.

  CLEAR wa_fieldcat.
  wa_fieldcat-tabname = 'INT_COUNT'.
  wa_fieldcat-fieldname = 'TOTDESP'.
  wa_fieldcat-outputlen =  8.
  wa_fieldcat-seltext_l = 'Total Despatched'.
  APPEND wa_fieldcat TO int_fieldcat.

  CLEAR wa_fieldcat.
  wa_fieldcat-tabname = 'INT_COUNT'.
  wa_fieldcat-fieldname = 'TOTFCWONO'.
  wa_fieldcat-outputlen =  8.
  wa_fieldcat-seltext_l = 'Foreign Currency Items'.
  APPEND wa_fieldcat TO int_fieldcat.

  IF s_pspnr IS NOT INITIAL.
    CLEAR wa_fieldcat.
    wa_fieldcat-tabname = 'INT_COUNT'.
    wa_fieldcat-fieldname = 'NAME1'.
    wa_fieldcat-outputlen =  15.
*   WA_FIELDCAT-COL_POS     = 0.
*     wa_fieldcat-key          = 'X'.
*   WA_FIELDCAT-HOTSPOT = 'X'.
*   WA_FIELDCAT-SELTEXT_M   = 'S O '.
    wa_fieldcat-seltext_l = 'Customer Name'.
    APPEND wa_fieldcat TO int_fieldcat.

    CLEAR wa_fieldcat.
    wa_fieldcat-tabname = 'INT_COUNT'.
    wa_fieldcat-fieldname = 'POST1'.
    wa_fieldcat-outputlen =  15.
*   WA_FIELDCAT-COL_POS     = 0.
*     wa_fieldcat-key          = 'X'.
*   WA_FIELDCAT-HOTSPOT = 'X'.
*   WA_FIELDCAT-SELTEXT_M   = 'S O '.
    wa_fieldcat-seltext_l = 'Project Name'.
    APPEND wa_fieldcat TO int_fieldcat.
  ENDIF.


  CALL FUNCTION 'REUSE_ALV_EVENTS_GET'
```

```
    EXPORTING
      i_list_type     = 4
*     I_CALLBACK_USER_COMMAND = 'HOT_SPOT'
    IMPORTING
      et_events       = it_events[]
    EXCEPTIONS
      list_type_wrong = 1
      OTHERS          = 2.

  READ TABLE it_events WITH KEY name = 'TOP_OF_PAGE'.
  IF sy-subrc = 0.
    it_events-form = 'TOP_OF_PAGE'.
    MODIFY it_events INDEX sy-tabix TRANSPORTING form.
  ENDIF.

  READ TABLE it_events WITH KEY name = 'USER_COMMAND'.
  IF sy-subrc = 0.
    it_events-form = 'USER_COMMAND'.
    MODIFY it_events INDEX sy-tabix TRANSPORTING form.
  ENDIF.
ENDFORM.                    "BUILD_CATALOG

*&---------------------------------------------------------------------*
*&      Form  DISPLAY_DATA
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
FORM display_data.

  IF int_count[] IS NOT INITIAL.

    gd_repid = sy-repid.
    CALL FUNCTION 'REUSE_ALV_GRID_DISPLAY'
      EXPORTING
        i_callback_program      = gd_repid
        i_callback_top_of_page = 'TOP-OF-PAGE'   "SEE FORM
*       IS_LAYOUT               = GD_LAYOUT
*       I_CALLBACK_USER_COMMAND = 'HOT_SPOT'
*       I_BACKGROUND_ID         = 'ALV_BACKGROUND'
*       I_GRID_TITLE            = W_TITLE
        it_fieldcat            = int_fieldcat[]
        it_events              = it_events[]
        i_save                 = 'U'
        is_variant             = es_variant
      TABLES
        t_outtab               = int_count
      EXCEPTIONS
        program_error          = 1
        OTHERS                 = 2.
    IF sy-subrc <> 0.
      MESSAGE ID sy-msgid TYPE sy-msgty NUMBER sy-msgno
              WITH sy-msgv1 sy-msgv2 sy-msgv3 sy-msgv4.
    ENDIF.
  ELSE.
    MESSAGE 'NO RECORDS FOUND FOR THE INPUT' TYPE 'I'.
  ENDIF.
ENDFORM.                    "DISPLAY_DATA

*&---------------------------------------------------------------------*
*&      Form  TOP_OF_PAGE
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
FORM top_of_page.
*      ALV HEADER DECLARATIONS
  DATA: t_header      TYPE slis_t_listheader,
        wa_header     TYPE slis_listheader,
        t_line        LIKE wa_header-info,
        ld_lines      TYPE i,
        ld_linesc(10) TYPE c.
* TITLE
  wa_header-typ  = 'H'.
  wa_header-info = 'Agency wise Total and Pendency report'.
  APPEND wa_header TO t_header.
  CLEAR wa_header.

  IF s_pspnr IS INITIAL.


    wa_header-typ  = 'S'.
    wa_header-key = 'Sales Order :'.
    wa_header-info = p_vbeln.
    APPEND wa_header TO t_header.
    CLEAR wa_header.

    wa_header-typ  = 'S'.
    wa_header-key = 'Customer Name :'.
    wa_header-info = t_name1.
    APPEND wa_header TO t_header.
    CLEAR wa_header.

    wa_header-typ  = 'S'.
    wa_header-key = 'Project Name :'.
    wa_header-info = t_projname.
    APPEND wa_header TO t_header.
    CLEAR wa_header.

  ENDIF.

  wa_header-typ  = 'S'.
  wa_header-key = 'Status as on :'.
  wa_header-info = rep_dt.
  APPEND wa_header TO t_header.
  CLEAR wa_header.

  CALL FUNCTION 'REUSE_ALV_COMMENTARY_WRITE'
```

```
    EXPORTING
      it_list_commentary = t_header.

ENDFORM.                    "TOP_OF_PAGE


*&---------------------------------------------------------------------*
*&      Form  USER_COMMAND
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
*      -->UCOMM      text
*      -->SEL_FIELD  text
*----------------------------------------------------------------------*
FORM user_command USING ucomm TYPE sy-ucomm
        sel_field TYPE slis_selfield.
  break 6146252.
  IF ucomm = '&IC1'.
    colsel = sel_field-sel_tab_field.
*    MESSAGE COLSEL TYPE 'I'.
    READ TABLE int_count INDEX sel_field-tabindex.
    rowsel = int_count-werks.

    break 1896563.
    IF s_pspnr[] IS NOT INITIAL.

      CALL FUNCTION 'CONVERSION_EXIT_ABPSP_OUTPUT'
        EXPORTING
          input  = int_count-pspid
        IMPORTING
          output = p_pspid.

      REPLACE ALL OCCURRENCES OF '-' IN p_pspid WITH ''.
      CONDENSE p_pspid NO-GAPS.

      p_vbeln = p_pspid.

      t_name1 = int_count-name1.
      t_projname = int_count-post1.
    ENDIF.


*    MESSAGE ROWSEL TYPE 'I'.

    PERFORM get_details USING rowsel colsel int_count-pspnr.
    PERFORM build_details.
    PERFORM disp_details.
  ENDIF.

ENDFORM.                    "USER_COMMAND

*&---------------------------------------------------------------------*
*&      Form  DISP_DETAILS
```

```
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
*      -->I_ROW      Row which is double clicked by user
*      -->I_COL      Column which is double clicked by user
*----------------------------------------------------------------------*
FORM get_details USING i_row i_col wo.
  break 6172601.
  DATA: BEGIN OF int_resb OCCURS 0,
          rsnum TYPE resb-rsnum,
          rspos TYPE resb-rspos,
          bdmng TYPE resb-bdmng,
          meins TYPE resb-meins,
          banfn TYPE resb-banfn,
          bnfpo TYPE resb-bnfpo,
        END OF int_resb.

  DATA: BEGIN OF int_ekpo OCCURS 0,
          ebeln      TYPE ekpo-ebeln,
          ebelp      TYPE ekpo-ebelp,
          matnr      TYPE ekpo-matnr,
          menge      TYPE ekpo-menge,
          meins      TYPE ekpo-meins,
          banfn      TYPE ekpo-banfn,
          bnfpo      TYPE ekpo-bnfpo,
          ps_psp_pnr TYPE ekkn-ps_psp_pnr,
        END OF int_ekpo.

  DATA: BEGIN OF int_ebkn OCCURS 0,
          banfn      TYPE ebkn-banfn,
          bnfpo      TYPE ebkn-bnfpo,
          erdat      TYPE ebkn-erdat,
          ps_psp_pnr TYPE ebkn-ps_psp_pnr,
        END OF int_ebkn.

  DATA: BEGIN OF int_eban OCCURS 0,
          banfn      TYPE eban-banfn,
          bnfpo      TYPE eban-bnfpo,
          matnr      TYPE eban-matnr,
          menge      TYPE eban-menge,
          frgdt      TYPE eban-frgdt,
          erdat      TYPE ebkn-erdat,
          bednr      TYPE eban-bednr,
          ps_psp_pnr TYPE ebkn-ps_psp_pnr,
        END OF int_eban,

        int_eban_cd LIKE TABLE OF int_eban WITH HEADER LINE.

  DATA: BEGIN OF int_ekkn OCCURS 0,
          ebeln      TYPE ekkn-ebeln,
          ebelp      TYPE ekkn-ebelp,
          ps_psp_pnr TYPE ekkn-ps_psp_pnr,
        END OF int_ekkn.
```

```
DATA: BEGIN OF int_ekko OCCURS 0,
        ebeln    TYPE ekko-ebeln,
        lifnr    TYPE ekko-lifnr,
        bedat    TYPE ekko-bedat,
        eq_eindt TYPE ekko-eq_eindt,
        banfn    TYPE ekpo-banfn,
        bnfpo    TYPE ekpo-bnfpo,
        ebelp    TYPE ekpo-ebelp,
      END OF int_ekko,
      int_ekko_cd LIKE STANDARD TABLE OF int_ekko WITH HEADER LINE.

DATA: BEGIN OF int_eket OCCURS 0,
        ebeln TYPE eket-ebeln,
        ebelp TYPE eket-ebelp,
        eindt TYPE eket-eindt,
      END OF int_eket,
      int_eket_cd LIKE STANDARD TABLE OF int_eket WITH HEADER LINE.

DATA: BEGIN OF int_lfa1 OCCURS 0,
        lifnr TYPE lfa1-lifnr,
        name1 TYPE lfa1-name1,
      END OF int_lfa1,
      int_lfa1_cd LIKE STANDARD TABLE OF int_lfa1 WITH HEADER LINE.

DATA: BEGIN OF int_mkpf OCCURS 0,
        mblnr TYPE mkpf-mblnr,
        mjahr TYPE mkpf-mjahr,
        bldat TYPE mkpf-bldat,
      END OF int_mkpf.

TYPES:BEGIN OF ty_mseg,
        mblnr      TYPE mseg-mblnr,
        mjahr      TYPE mseg-mjahr,
        zeile      TYPE mseg-zeile,
        bwart      TYPE mseg-bwart,
        matnr      TYPE mseg-matnr,
        werks      TYPE mseg-werks,
        lgort      TYPE mseg-lgort,
        charg      TYPE mseg-charg,
        lifnr      TYPE mseg-lifnr,
        dmbtr      TYPE mseg-dmbtr,
        menge      TYPE mseg-menge,
        meins      TYPE mseg-meins,
        ebeln      TYPE mseg-ebeln,
        ebelp      TYPE mseg-ebelp,
        lfbja      TYPE mseg-lfbja,  " ADDED ON 06.11.2015
        lfbnr      TYPE mseg-lfbnr,
        lfpos      TYPE mseg-lfpos,
        sjahr      TYPE mseg-sjahr,
        ps_psp_pnr TYPE mseg-ps_psp_pnr,
        bldat      TYPE mkpf-bldat,
        ibdno      TYPE likp-vbeln,
        zzrlr      TYPE likp-zzrlr,
        zzrld      TYPE likp-zzrld,
        zztnm      TYPE likp-zztnm,
      END OF ty_mseg.

DATA: int_mseg    TYPE STANDARD TABLE OF ty_mseg WITH HEADER LINE,
      int_mseg102 TYPE STANDARD TABLE OF ty_mseg WITH HEADER LINE.

DATA: BEGIN OF int_vbfa_det OCCURS 0,
        vbelv TYPE vbfa-vbelv,
        vbeln TYPE vbfa-vbeln,
      END OF int_vbfa_det.

DATA: BEGIN OF int_likp_det OCCURS 0,
        vbeln TYPE likp-vbeln,
        zzrlr TYPE likp-zzrlr,
        zzrld TYPE likp-zzrld,
        vbelv TYPE vbfa-vbeln,
        zztnm TYPE likp-zztnm,
      END OF int_likp_det.

REFRESH int_details.
CLEAR int_details.
CLEAR d_divis.
break 1896563.
IF i_row IS NOT INITIAL.
  CASE i_col.
    WHEN 'INT_COUNT-MPCTOT' .
      LOOP AT int_final WHERE mpctot = 1 AND werks = i_row.
        MOVE-CORRESPONDING int_final TO int_details.
        APPEND int_details.
        CLEAR int_details.
      ENDLOOP.
    WHEN 'INT_COUNT-MPCPEN'.
      LOOP AT int_final WHERE mpcpen = 1 AND werks = i_row..
        MOVE-CORRESPONDING int_final TO int_details.
        APPEND int_details.
        CLEAR int_details.
      ENDLOOP.
    WHEN 'INT_COUNT-PPCTOT'.
      LOOP AT int_final WHERE ppctot = 1 AND werks = i_row.
        MOVE-CORRESPONDING int_final TO int_details.
        APPEND int_details.
        CLEAR int_details.
      ENDLOOP.
    WHEN 'INT_COUNT-PPCPEN'.
      LOOP AT int_final WHERE ppcpen = 1 AND werks = i_row.
        MOVE-CORRESPONDING int_final TO int_details.
        APPEND int_details.
        CLEAR int_details.
      ENDLOOP.
    WHEN 'INT_COUNT-SUBTOT'.
      LOOP AT int_final WHERE subtot = 1 AND werks = i_row.
        MOVE-CORRESPONDING int_final TO int_details.
        APPEND int_details.
        CLEAR int_details.
      ENDLOOP.
    WHEN 'INT_COUNT-SUBPEN'.
      LOOP AT int_final WHERE subpen = 1 AND werks = i_row .
        MOVE-CORRESPONDING int_final TO int_details.
        APPEND int_details.
        CLEAR int_details.
      ENDLOOP.
    WHEN 'INT_COUNT-PURTOT'.
      LOOP AT int_final WHERE purtot = 1 AND werks = i_row.
        MOVE-CORRESPONDING int_final TO int_details.
        APPEND int_details.
        CLEAR int_details.
      ENDLOOP.
    WHEN 'INT_COUNT-PURPEN'.
      LOOP AT int_final WHERE purpen = 1 AND werks = i_row.
        MOVE-CORRESPONDING int_final TO int_details.
        APPEND int_details.
        CLEAR int_details.
      ENDLOOP.
    WHEN 'INT_COUNT-STRPEN'.
      LOOP AT int_final WHERE strpen = 1 AND werks = i_row .
        MOVE-CORRESPONDING int_final TO int_details.
        APPEND int_details.
        CLEAR int_details.
      ENDLOOP.
    WHEN 'INT_COUNT-GPISSU'.
      LOOP AT int_final WHERE gpissu = 1 AND werks = i_row .
        MOVE-CORRESPONDING int_final TO int_details.
        APPEND int_details.
        CLEAR int_details.
      ENDLOOP.
    WHEN 'INT_COUNT-CDCACK'.
      LOOP AT int_final WHERE cdcack = 1 AND werks = i_row .
        MOVE-CORRESPONDING int_final TO int_details.
        APPEND int_details.
        CLEAR int_details.
      ENDLOOP.
    WHEN 'INT_COUNT-CMLPEN'.
      LOOP AT int_final WHERE cmlpen = 1 AND werks = i_row.
        MOVE-CORRESPONDING int_final TO int_details.
        APPEND int_details.
        CLEAR int_details.
      ENDLOOP.
    WHEN 'INT_COUNT-CDCPQ'.
      LOOP AT int_final WHERE cdcpq = 1 AND werks = i_row .
        MOVE-CORRESPONDING int_final TO int_details.
        APPEND int_details.
        CLEAR int_details.
      ENDLOOP.
    WHEN 'INT_COUNT-CDCPAK'.
```

```
        LOOP AT int_final WHERE cdcpak = 1 AND werks = i_row .
          MOVE-CORRESPONDING int_final TO int_details.
          APPEND int_details.
          CLEAR int_details.
        ENDLOOP.

      WHEN 'INT_COUNT-TOTDESP'.
        LOOP AT int_final WHERE totdesp = 1 AND werks = i_row.
          MOVE-CORRESPONDING int_final TO int_details.
          APPEND int_details.
          CLEAR int_details.
        ENDLOOP.
      WHEN 'INT_COUNT-TOTTOT'.
        LOOP AT int_final WHERE ( mpctot = 1 OR ppctot = 1 OR subtot = 1 OR pur
tot = 1 ) AND werks = i_row.
          MOVE-CORRESPONDING int_final TO int_details.
          APPEND int_details.
          CLEAR int_details.
        ENDLOOP.
      WHEN 'INT_COUNT-NOTCDC'.
        LOOP AT int_final WHERE ( mpcpen = 1 OR ppcpen = 1 OR subpen = 1 OR pur
pen = 1
                            OR strpen = 1 OR gpissu = 1 ) AND werks = i_row.
          MOVE-CORRESPONDING int_final TO int_details.
          APPEND int_details.
          CLEAR int_details.
        ENDLOOP.
      WHEN 'INT_COUNT-TOTPEN'.
        LOOP AT int_final WHERE ( mpcpen = 1 OR ppcpen = 1 OR subpen = 1 OR pur
pen = 1 OR strpen = 1
          OR gpissu = 1 OR cdcack = 1 OR cmlpen = 1 OR cdcpak = 1 OR cdcpq = 1
) AND werks = i_row.
          MOVE-CORRESPONDING int_final TO int_details.
          APPEND int_details.
          CLEAR int_details.
        ENDLOOP.
      WHEN 'INT_COUNT-TOTFCWONO'.
        LOOP AT int_final WHERE totfcwono = 1 AND werks EQ i_row.
          MOVE-CORRESPONDING int_final TO int_details.
          APPEND int_details.
          CLEAR int_details.
        ENDLOOP.
        PERFORM display_fc_details USING i_row.
    ENDCASE.
  ELSE.
    CASE i_col.
      WHEN 'INT_COUNT-MPCTOT'.
        LOOP AT int_final WHERE mpctot = 1.
          MOVE-CORRESPONDING int_final TO int_details.
          APPEND int_details.
          CLEAR int_details.
        ENDLOOP.
      WHEN 'INT_COUNT-MPCPEN'.
        LOOP AT int_final WHERE mpcpen = 1.
          MOVE-CORRESPONDING int_final TO int_details.
          APPEND int_details.
          CLEAR int_details.
        ENDLOOP.
      WHEN 'INT_COUNT-PPCTOT'.
        LOOP AT int_final WHERE ppctot = 1.
          MOVE-CORRESPONDING int_final TO int_details.
          APPEND int_details.
          CLEAR int_details.
        ENDLOOP.
      WHEN 'INT_COUNT-PPCPEN'.
        LOOP AT int_final WHERE ppcpen = 1.
          MOVE-CORRESPONDING int_final TO int_details.
          APPEND int_details.
          CLEAR int_details.
        ENDLOOP.
      WHEN 'INT_COUNT-SUBTOT'.
        LOOP AT int_final WHERE subtot = 1.
          MOVE-CORRESPONDING int_final TO int_details.
          APPEND int_details.
          CLEAR int_details.
        ENDLOOP.
      WHEN 'INT_COUNT-SUBPEN'.
        LOOP AT int_final WHERE subpen = 1.
          MOVE-CORRESPONDING int_final TO int_details.
          APPEND int_details.
          CLEAR int_details.
        ENDLOOP.
      WHEN 'INT_COUNT-PURTOT'.
        LOOP AT int_final WHERE purtot = 1.
          MOVE-CORRESPONDING int_final TO int_details.
          APPEND int_details.
          CLEAR int_details.
        ENDLOOP.
      WHEN 'INT_COUNT-PURPEN'.
        LOOP AT int_final WHERE purpen = 1.
          MOVE-CORRESPONDING int_final TO int_details.
          APPEND int_details.
          CLEAR int_details.
        ENDLOOP.
      WHEN 'INT_COUNT-STRPEN'.
        LOOP AT int_final WHERE strpen = 1.
          MOVE-CORRESPONDING int_final TO int_details.
          APPEND int_details.
          CLEAR int_details.
        ENDLOOP.
      WHEN 'INT_COUNT-GPISSU'.
        LOOP AT int_final WHERE gpissu = 1.
          MOVE-CORRESPONDING int_final TO int_details.
          APPEND int_details.
          CLEAR int_details.
        ENDLOOP.
```

```
      WHEN 'INT_COUNT-CDCACK'.
        LOOP AT int_final WHERE cdcack = 1.
          MOVE-CORRESPONDING int_final TO int_details.
          APPEND int_details.
          CLEAR int_details.
        ENDLOOP.
      WHEN 'INT_COUNT-CMLPEN'.
        LOOP AT int_final WHERE cmlpen = 1.
          MOVE-CORRESPONDING int_final TO int_details.
          APPEND int_details.
          CLEAR int_details.
        ENDLOOP.
      WHEN 'INT_COUNT-CDCPQ'.
        LOOP AT int_final WHERE cdcpq = 1.
          MOVE-CORRESPONDING int_final TO int_details.
          APPEND int_details.
          CLEAR int_details.
        ENDLOOP.
      WHEN 'INT_COUNT-CDCPAK'.
        LOOP AT int_final WHERE cdcpak = 1.
          MOVE-CORRESPONDING int_final TO int_details.
          APPEND int_details.
          CLEAR int_details.
        ENDLOOP.
      WHEN 'INT_COUNT-TOTDESP'.
        LOOP AT int_final WHERE totdesp = 1.
          MOVE-CORRESPONDING int_final TO int_details.
          APPEND int_details.
          CLEAR int_details.
        ENDLOOP.
      WHEN 'INT_COUNT-TOTTOT'.
        LOOP AT int_final.
          MOVE-CORRESPONDING int_final TO int_details.
          APPEND int_details.
          CLEAR int_details.
        ENDLOOP.
      WHEN 'INT_COUNT-NOTCDC'.
        LOOP AT int_final WHERE mpcpen = 1 OR ppcpen = 1 OR subpen = 1 OR purpe
n = 1 OR strpen = 1 OR gpissu = 1.
          MOVE-CORRESPONDING int_final TO int_details.
          APPEND int_details.
          CLEAR int_details.
        ENDLOOP.
      WHEN 'INT_COUNT-TOTPEN'.
        LOOP AT int_final WHERE mpcpen = 1 OR ppcpen = 1 OR subpen = 1 OR purpe
n = 1
                             OR strpen = 1 OR gpissu = 1 OR cdcack = 1 OR cmlpe
n = 1 OR cdcpak = 1 OR cdcpq = 1 .
          MOVE-CORRESPONDING int_final TO int_details.
          APPEND int_details.
          CLEAR int_details.
        ENDLOOP.
      WHEN 'INT_COUNT-TOTFCWONO'.
```

```
        LOOP AT int_final WHERE totfcwono = 1.
          MOVE-CORRESPONDING int_final TO int_details.
          APPEND int_details.
          CLEAR int_details.
        ENDLOOP.
        PERFORM display_fc_details USING ''.
    ENDCASE.
  ENDIF.

  IF wo IS NOT INITIAL.
    DELETE int_details WHERE pspnr NE wo.
  ENDIF.



  IF int_details[] IS NOT INITIAL.
    IF i_col = 'INT_COUNT-TOTDESP' OR i_col = 'INT_COUNT-TOTFCWONO'.
      SELECT tknum tpnum vbeln
        FROM vttp INTO TABLE int_vttp
        FOR ALL ENTRIES IN int_details
        WHERE vbeln = int_details-deliv.
      IF int_vttp[] IS NOT INITIAL.
        SELECT tknum signi exti1 tdlnr sdabw datbg
          FROM vttk INTO TABLE int_vttk
          FOR ALL ENTRIES IN int_vttp
          WHERE tknum = int_vttp-tknum
          AND sttbg = 'X'.
      ENDIF.

      LOOP AT int_details.
        READ TABLE int_vttp WITH KEY vbeln = int_details-deliv.
        IF sy-subrc = 0.
          MOVE int_vttp-tknum TO int_details-tknum.
          MODIFY int_details.
        ENDIF.
        READ TABLE int_vttk WITH KEY tknum = int_details-tknum.
        IF sy-subrc = 0.
          MOVE int_vttk-signi TO int_details-signi.
          MOVE int_vttk-exti1 TO int_details-exti1.
          MOVE int_vttk-tdlnr TO int_details-tdlnr.
          MOVE int_vttk-sdabw TO int_details-sdabw.
          MOVE int_vttk-datbg TO int_details-datbg.
          MODIFY int_details.
        ENDIF.
      ENDLOOP.
    ENDIF.

    IF ( i_col = 'INT_COUNT-CDCPAK' OR i_col = 'INT_COUNT-CDCPQ' OR i_col = 'IN
T_COUNT-TOTDESP' OR i_col = 'INT_COUNT-TOTPEN' OR i_col = 'INT_COUNT-TOTFCWONO'
 ).
      SELECT SINGLE zzpcd INTO zzpcd FROM vbak WHERE vbeln = p_vbeln.
      SELECT vbeln bldat bolnr podat zzdid ernam
        INTO TABLE int_likp
        FROM likp
        FOR ALL ENTRIES IN int_details
        WHERE vbeln = int_details-deliv.
      LOOP AT int_details.
        IF int_details-status = 'SCLD'.
          int_details-remarks = 'Short Closed'.
        ENDIF.

        READ TABLE int_likp WITH KEY vbeln = int_details-deliv.
        IF sy-subrc = 0.
          MOVE int_likp-deldt TO int_details-deldt.
          MOVE int_likp-bolnr TO int_details-bolnr.
          MOVE int_likp-podat TO int_details-podat.
          MOVE int_likp-zzdid TO int_details-zzdid.
          MOVE int_likp-ernam TO int_details-cdc_off.

          CLEAR: w_name, w_accnt.
          IF int_likp-ernam+0(1) = 'C'.
            SELECT SINGLE accnt FROM usr02 INTO w_accnt WHERE bname = int_likp-
ernam.
            IF sy-subrc = 0.
              SELECT SINGLE ename FROM pa0001 INTO w_name WHERE pernr = w_accnt
.
            ENDIF.
          ELSE.
            SELECT SINGLE ename FROM pa0001 INTO w_name WHERE pernr = int_likp-
ernam.
          ENDIF.
          int_details-cdc_name = w_name.
          IF zzpcd IS NOT INITIAL OR int_details-zzdid IS NOT INITIAL.
            CONCATENATE zzpcd int_details-zzdid INTO int_details-pkgno.
          ENDIF.
        ENDIF.
        IF int_details-deliv IS NOT INITIAL.
          int_details-ack_txt = 'Packed'.
        ELSEIF int_details-sales_indi = 'X'.
          int_details-ack_txt = 'Acknowledged by CDC and ready for Packing'.
        ELSEIF int_details-goods_indi = 'X'.
          int_details-ack_txt = 'Acknowledged by CDC'.
        ENDIF.
        MODIFY int_details.
      ENDLOOP.
    ENDIF.

    IF ( i_col = 'INT_COUNT-PURPEN' OR i_col = 'INT_COUNT-PURTOT' OR i_col = 'I
NT_COUNT-SUBPEN' OR i_col = 'INT_COUNT-SUBTOT').
      REFRESH: int_resb, int_ekpo, int_ebkn, int_eban, int_ekkn, int_ekko, int_
eket, int_lfa1, int_mkpf, int_mseg, int_mseg102, int_vbfa_det, int_likp_det, in
t_rdr_det.
      CLEAR  : int_resb, int_ekpo, int_ebkn, int_eban, int_ekkn, int_ekko, int_
eket, int_lfa1, int_mkpf, int_mseg, int_mseg102, int_vbfa_det, int_likp_det, in
t_rdr_det.
```

```
*****************************DD PR PO Data*******************************
      SELECT rsnum rspos aufnr
        INTO TABLE it_zmmt001_temp
      FROM zmmt001
        FOR ALL ENTRIES IN int_details
      WHERE rsnum = int_details-rsnum
        AND rspos = int_details-rspos
        AND bwart = '281'
        AND xloek = ''.

      LOOP AT it_zmmt001_temp.
        e_network-network = it_zmmt001_temp-aufnr.
        APPEND e_network.
      ENDLOOP.

      SORT e_network BY network ASCENDING.
      DELETE ADJACENT DUPLICATES FROM e_network.

**      CALL FUNCTION 'BAPI_NETWORK_GETINFO'
**        TABLES
**          i_network_list = e_network
**          e_component    = e_component.

      LOOP AT e_network.        " Added by Siva 15.12.2020

        CALL FUNCTION 'BAPI_NETWORK_COMP_GETDETAIL'
          EXPORTING
            number              = e_network-network
*           MAX_ROWS             = 0
* IMPORTING
*           RETURN               =
          TABLES
*           I_ACTIVITY_RANGE     =
*           I_COMPONENTS_ID      =
            e_components_detail = e_component1.


        APPEND LINES OF e_component1 TO e_component.

        CLEAR e_network.
        REFRESH : e_component1 , e_component1[].
      ENDLOOP.

      LOOP AT e_component.
        READ TABLE it_zmmt001_temp WITH KEY rsnum = e_component-reserv_no rspos
 = e_component-res_item.
        IF sy-subrc NE 0.
          DELETE e_component.
        ENDIF.
      ENDLOOP.
      SELECT banfn bnfpo matnr menge frgdt badat bednr
             INTO TABLE int_eban  FROM eban
             FOR ALL ENTRIES IN e_component
             WHERE banfn = e_component-preq_no
             AND bnfpo = e_component-preq_item
             AND loekz = ''.

      IF int_eban[] IS NOT INITIAL.
        SELECT b~ebeln b~lifnr b~bedat b~eq_eindt a~banfn a~bnfpo a~ebelp
          INTO TABLE int_ekko
        FROM ekpo AS a
          INNER JOIN ekko AS b ON ( b~ebeln = a~ebeln )
          FOR ALL ENTRIES IN int_eban
        WHERE a~banfn = int_eban-banfn
          AND a~bnfpo = int_eban-bnfpo
          AND a~loekz = ''.
      ENDIF.
      IF int_ekko[] IS NOT INITIAL.
        SELECT lifnr name1
          FROM lfa1 INTO TABLE int_lfa1          "Vendor Name
          FOR ALL ENTRIES IN int_ekko
          WHERE lifnr = int_ekko-lifnr.

        SELECT ebeln ebelp eindt
          FROM eket INTO TABLE int_eket          "Delivery Date
          FOR ALL ENTRIES IN int_ekko
          WHERE ebeln = int_ekko-ebeln
          AND ebelp = int_ekko-ebelp.
        SORT int_eket BY ebeln ebelp eindt.
      ENDIF.

********************************************************************************
*** ADDED ON 16.11.2011 BY 6058515
      REFRESH : int_eban_cd , int_details_cd, int_ekko_cd , int_lfa1_cd ,int_ek
et_cd.

      int_details_cd[] = int_details[].
      DELETE int_details_cd WHERE dutype NE 'CD'.
      IF int_details_cd[] IS NOT INITIAL.
******  1. PASS WBS ELEMENT AND MATERIAL COMBINATION TO EBAN, EBKN JOIN TABLE
TO GET PR INFORMATION FOR CD ITEMS
        SELECT a~banfn a~bnfpo a~matnr a~menge a~frgdt a~badat a~bednr b~ps_psp
_pnr FROM eban AS a INNER JOIN ebkn AS b ON ( a~banfn EQ b~banfn AND a~bnfpo EQ
 b~bnfpo )
                INTO TABLE int_eban_cd
                FOR ALL ENTRIES IN int_details_cd
                WHERE a~matnr = int_details_cd-zdesno
                AND   a~loekz = ''
                AND   b~ps_psp_pnr = int_details_cd-posnr.

***** 2. GET PO INFO BY PASSING PR AND PR ITEM
        IF int_eban_cd[] IS NOT INITIAL.
```

```
          SELECT b~ebeln b~lifnr b~bedat b~eq_eindt a~banfn a~bnfpo a~ebelp
                   INTO TABLE int_ekko_cd
                 FROM ekpo AS a
                   INNER JOIN ekko AS b ON ( b~ebeln = a~ebeln )
                   FOR ALL ENTRIES IN int_eban_cd
                 WHERE a~banfn = int_eban_cd-banfn
                   AND a~bnfpo = int_eban_cd-bnfpo
                   AND a~loekz = ''.

          IF int_ekko_cd[] IS NOT INITIAL.
            SELECT lifnr name1
              FROM lfa1 INTO TABLE int_lfa1_cd          "Vendor Name
              FOR ALL ENTRIES IN int_ekko_cd
              WHERE lifnr = int_ekko_cd-lifnr.

            SELECT ebeln ebelp eindt
              FROM eket INTO TABLE int_eket_cd          "Delivery Date
              FOR ALL ENTRIES IN int_ekko_cd
              WHERE ebeln = int_ekko_cd-ebeln
              AND ebelp = int_ekko_cd-ebelp.
            SORT int_eket_cd BY ebeln ebelp eindt ASCENDING.
          ENDIF.


        ENDIF.
      ENDIF.

*********************************************************************************
********
      LOOP AT int_details.
        IF int_details-dutype EQ 'CD'.    " PO, PR INFORMATION LOGIC FOR CD ITE
MS

          CLEAR int_eban_cd.
          READ TABLE int_eban_cd WITH KEY matnr = int_details-zdesno  ps_psp_pn
r = int_details-posnr .
          int_details-banfn = int_eban_cd-banfn.
          int_details-bnfpo = int_eban_cd-bnfpo.
          int_details-prdat = int_eban_cd-erdat.
          int_details-frgdt = int_eban_cd-frgdt.

          CLEAR int_ekko_cd.
          READ TABLE int_ekko_cd WITH KEY banfn = int_eban_cd-banfn bnfpo = int
_eban_cd-bnfpo.
          int_details-ebeln = int_ekko_cd-ebeln.
          int_details-ebelp = int_ekko_cd-ebelp.
          int_details-bedat = int_ekko_cd-bedat.
          int_details-lifnr = int_ekko_cd-lifnr.

          CLEAR int_lfa1_cd.
          READ TABLE int_lfa1_cd WITH KEY lifnr = int_ekko_cd-lifnr.
          int_details-name1 = int_lfa1_cd-name1.
          CLEAR int_eket_cd.
          READ TABLE int_eket_cd WITH KEY ebeln = int_ekko_cd-ebeln ebelp = int
_ekko_cd-ebelp.
          int_details-eindt = int_eket_cd-eindt.
**********************<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<         END OF LOGIC 16
.11.2011
        ELSE.
          CLEAR e_component.
          READ TABLE e_component WITH KEY reserv_no = int_details-rsnum  res_it
em = int_details-rspos.

          CLEAR int_eban.
          READ TABLE int_eban WITH KEY banfn = e_component-preq_no bnfpo = e_co
mponent-preq_item.
          int_details-banfn = int_eban-banfn.
          int_details-bnfpo = int_eban-bnfpo.
          int_details-prdat = int_eban-erdat.
          int_details-frgdt = int_eban-frgdt.

          CLEAR int_ekko.
          READ TABLE int_ekko WITH KEY banfn = int_eban-banfn bnfpo = int_eban-
bnfpo.
          int_details-ebeln = int_ekko-ebeln.
          int_details-ebelp = int_ekko-ebelp.
          int_details-bedat = int_ekko-bedat.
          int_details-lifnr = int_ekko-lifnr.

          CLEAR int_lfa1.
          READ TABLE int_lfa1 WITH KEY lifnr = int_ekko-lifnr.
          int_details-name1 = int_lfa1-name1.

          CLEAR int_eket.
          READ TABLE int_eket WITH KEY ebeln = int_ekko-ebeln ebelp = int_ekko-
ebelp.
          int_details-eindt = int_eket-eindt.
        ENDIF.
        MODIFY int_details TRANSPORTING banfn bnfpo prdat frgdt ebeln bedat lif
nr name1 eindt ebelp.

      ENDLOOP.

******************For Uploaded POs********************
      CLEAR: int_ekko, int_ekko[], int_ekkn, int_ekkn[],int_ekpo, int_ekpo[],in
t_ekko,int_ekko[],int_lfa1,int_lfa1[],int_eket,int_eket[].
      SELECT ebeln ebelp ps_psp_pnr
        FROM ekkn INTO TABLE int_ekkn                      "PO Items WBS
        FOR ALL ENTRIES IN int_details
        WHERE ps_psp_pnr = int_details-posnr
        AND loekz = ''.
      IF int_ekkn[] IS NOT INITIAL.
        REFRESH int_ekpo.
        SELECT ebeln ebelp matnr menge meins banfn bnfpo
          FROM ekpo INTO TABLE int_ekpo                   "PO Items MATNR
          FOR ALL ENTRIES IN int_ekkn
          WHERE ebeln = int_ekkn-ebeln
            AND ebelp = int_ekkn-ebelp
            AND lgort EQ 'DD99'
            AND loekz = ''.
        LOOP AT int_ekpo.
          READ TABLE int_ekkn WITH KEY ebeln = int_ekpo-ebeln ebelp = int_ekpo-
ebelp.
          IF sy-subrc = 0.
            MOVE int_ekkn-ps_psp_pnr TO int_ekpo-ps_psp_pnr.
            MODIFY int_ekpo TRANSPORTING ps_psp_pnr.
          ENDIF.
        ENDLOOP.

        SORT int_ekpo BY ebeln ebelp.

        LOOP AT int_details WHERE banfn IS INITIAL.
          READ TABLE int_ekpo WITH KEY matnr = int_details-zdesno ps_psp_pnr =
int_details-posnr.
          IF sy-subrc = 0.
            MOVE int_ekpo-ebeln TO int_details-ebeln.
            MOVE int_ekpo-ebelp TO int_details-ebelp.
            MOVE int_ekpo-menge TO int_details-pomenge.
            MOVE int_ekpo-meins TO int_details-pomeins.
            MODIFY int_details TRANSPORTING ebeln ebelp pomenge pomeins.
          ENDIF.
        ENDLOOP.
      ENDIF.

      IF int_details[] IS NOT INITIAL.
        SELECT ebeln lifnr bedat eq_eindt
          FROM ekko INTO TABLE int_ekko           "PO Header
          FOR ALL ENTRIES IN int_details
          WHERE ebeln = int_details-ebeln.
      ENDIF.
      IF int_ekko[] IS NOT INITIAL.
        SELECT lifnr name1
          FROM lfa1 INTO TABLE int_lfa1           "Vendor Name
          FOR ALL ENTRIES IN int_ekko
          WHERE lifnr = int_ekko-lifnr.

        SELECT ebeln ebelp eindt
          FROM eket INTO TABLE int_eket           "Delivery Date
          FOR ALL ENTRIES IN int_ekko
          WHERE ebeln = int_ekko-ebeln
          AND ebelp = int_ekko-ebelp.
        SORT int_eket BY ebeln ebelp eindt.
      ENDIF.

      LOOP AT int_details.
        READ TABLE int_ekko WITH KEY ebeln = int_details-ebeln.
        IF sy-subrc = 0.
          MOVE int_ekko-bedat TO int_details-bedat.
          MOVE int_ekko-lifnr TO int_details-lifnr.
          MODIFY int_details TRANSPORTING bedat lifnr.
        ENDIF.
        READ TABLE int_lfa1 WITH KEY lifnr = int_details-lifnr.
        IF sy-subrc = 0.
          MOVE int_lfa1-name1 TO int_details-name1.
          MODIFY int_details TRANSPORTING name1.
        ENDIF.
        READ TABLE int_eket WITH KEY ebeln = int_details-ebeln ebelp = int_deta
ils-ebelp.
        IF sy-subrc = 0.
          MOVE int_eket-eindt TO int_details-eindt.
          MODIFY int_details TRANSPORTING eindt.
        ENDIF.
      ENDLOOP.
********************Uploaded POs**********************


      IF int_details[] IS NOT INITIAL.

        IF ( i_col = 'INT_COUNT-PURPEN' OR i_col = 'INT_COUNT-PURTOT' ).

          int_det_tem[] = int_details[].
          DELETE int_det_tem WHERE ebeln = ''.

          IF int_det_tem[] IS NOT INITIAL.

            SELECT mblnr mjahr zeile bwart matnr werks lgort charg lifnr dmbtr
menge meins ebeln ebelp lfbja lfbnr lfpos sjahr ps_psp_pnr
              FROM mseg INTO CORRESPONDING FIELDS OF TABLE int_mseg
              FOR ALL ENTRIES IN int_det_tem
              WHERE ebeln = int_det_tem-ebeln AND ebelp = int_det_tem-ebelp
              AND lgort = 'DD99' AND sobkz = 'Q'
              AND bwart IN ('101', '102','122','124').
          ENDIF.

          IF int_mseg[] IS NOT INITIAL.
            LOOP AT int_mseg.
              IF int_mseg-bwart <> '101'.
                MOVE-CORRESPONDING int_mseg TO int_mseg102.
                IF int_mseg102-sjahr IS INITIAL. " IF CONDITION ADDED ON 06.11.
2015
                  int_mseg102-sjahr = int_mseg102-lfbja.
                ENDIF.
                APPEND int_mseg102.
                CLEAR  int_mseg102.
                DELETE int_mseg.
              ENDIF.
            ENDLOOP.
            LOOP AT int_mseg.
              READ TABLE int_mseg102 WITH KEY lfbnr = int_mseg-mblnr  sjahr = i
nt_mseg-mjahr. "lfpos = int_mseg-zeile
              IF sy-subrc = 0.
```

```
                DELETE int_mseg.
              ENDIF.
            ENDLOOP.
          ENDIF.
        ELSEIF ( i_col = 'INT_COUNT-SUBPEN' OR i_col = 'INT_COUNT-SUBTOT' ).

          IF int_mseg[] IS NOT INITIAL.
            LOOP AT int_mseg.
              IF int_mseg-bwart = '102' OR int_mseg-bwart = '104'.
                MOVE-CORRESPONDING int_mseg TO int_mseg102.
                APPEND int_mseg102.
                CLEAR  int_mseg102.
                DELETE int_mseg.
              ENDIF.
            ENDLOOP.
            LOOP AT int_mseg.
              READ TABLE int_mseg102 WITH KEY lfbnr = int_mseg-mblnr lfpos = in
t_mseg-zeile sjahr = int_mseg-mjahr.
              IF sy-subrc = 0.
                DELETE int_mseg.
              ENDIF.
            ENDLOOP.
          ENDIF.
        ENDIF.

      ENDIF.
      IF int_mseg[] IS NOT INITIAL.
        SELECT mblnr mjahr bldat
          FROM mkpf INTO CORRESPONDING FIELDS OF TABLE int_mkpf
          FOR ALL ENTRIES IN int_mseg
          WHERE mblnr = int_mseg-mblnr AND mjahr = int_mseg-mjahr.
      ENDIF.
      LOOP AT int_mseg.
        READ TABLE int_mkpf WITH KEY mblnr = int_mseg-mblnr mjahr = int_mseg-mj
ahr.
        IF sy-subrc = 0.
          MOVE int_mkpf-bldat TO int_mseg-bldat.
          MODIFY int_mseg TRANSPORTING bldat.
        ENDIF.
      ENDLOOP.
      IF int_mseg[] IS NOT INITIAL.
        SELECT vbelv vbeln
          FROM vbfa INTO CORRESPONDING FIELDS OF TABLE int_vbfa_det
          FOR ALL ENTRIES IN int_mseg
          WHERE vbeln = int_mseg-mblnr AND vbtyp_v = '7'.
      ENDIF.
      IF int_vbfa_det[] IS NOT INITIAL.
        SELECT vbeln zzrlr zzrld zztnm
          FROM likp INTO CORRESPONDING FIELDS OF TABLE int_likp_det
          FOR ALL ENTRIES IN int_vbfa_det
          WHERE vbeln = int_vbfa_det-vbelv.
        LOOP AT int_likp_det.
          READ TABLE int_vbfa_det WITH KEY vbelv = int_likp_det-vbeln.
          IF sy-subrc = 0.
            int_likp_det-vbelv = int_vbfa_det-vbeln.
            MODIFY int_likp_det TRANSPORTING vbelv.
          ENDIF.
        ENDLOOP.
      ENDIF.

      LOOP AT int_mseg.
        READ TABLE int_likp_det WITH KEY vbelv = int_mseg-mblnr.
        IF sy-subrc = 0.
          int_mseg-ibdno = int_likp_det-vbeln.
          int_mseg-zzrlr = int_likp_det-zzrlr.
          int_mseg-zzrld = int_likp_det-zzrld.
          int_mseg-zztnm = int_likp_det-zztnm.
          MODIFY int_mseg TRANSPORTING ibdno zzrlr zzrld zztnm.
        ENDIF.
      ENDLOOP.

      break 1897799.

      CLEAR: int_rdr_det, int_rdr_det[].
      LOOP AT int_mseg.
        READ TABLE int_details WITH KEY ebeln = int_mseg-ebeln ebelp = int_mseg
-ebelp.
        IF sy-subrc = 0.
          MOVE-CORRESPONDING int_details TO int_rdr_det.
          MOVE int_mseg-mblnr TO int_rdr_det-rdrno.
          MOVE int_mseg-mjahr TO int_rdr_det-rdryr.
          MOVE int_mseg-dmbtr TO int_rdr_det-rdrvl.
          MOVE int_mseg-menge TO int_rdr_det-rdrqt.
          MOVE int_mseg-bldat TO int_rdr_det-rdrdt.
          MOVE int_mseg-ibdno TO int_rdr_det-ibdno.
          MOVE int_mseg-zzrlr TO int_rdr_det-rlrno.
          MOVE int_mseg-zzrld TO int_rdr_det-rlrdt.
          MOVE int_mseg-zztnm TO int_rdr_det-zztnm.
          APPEND int_rdr_det.
          CLEAR int_rdr_det.
        ENDIF.
      ENDLOOP.

      LOOP AT int_details.
        READ TABLE int_rdr_det WITH KEY projn = int_details-projn pspid = int_d
etails-pspid pspnr = int_details-pspnr posnr = int_details-posnr zdesno = int_d
etails-zdesno partno = int_details-partno des_serial = int_details-des_serial.
        IF sy-subrc <> 0.
          MOVE-CORRESPONDING int_details TO int_rdr_det.
          APPEND int_rdr_det.
          CLEAR int_rdr_det.
        ENDIF.
      ENDLOOP.

      CLEAR: int_details,int_details[].
      break 1897799.
      int_details[] = int_rdr_det[].
    ENDIF.

    IF ( i_col = 'INT_COUNT-MPCTOT' OR i_col = 'INT_COUNT-MPCPEN' ) AND int_det
ails[] IS NOT INITIAL.

      SELECT rsnum rspos aufnr
        INTO TABLE it_zmmt001_temp
      FROM zmmt001
        FOR ALL ENTRIES IN int_details
      WHERE rsnum = int_details-rsnum
        AND rspos = int_details-rspos
        AND bwart = '281'
        AND xloek = ''.

      LOOP AT it_zmmt001_temp.
        e_network-network = it_zmmt001_temp-aufnr.
        APPEND e_network.
      ENDLOOP.

      SORT e_network BY network ASCENDING.
      DELETE ADJACENT DUPLICATES FROM e_network.
**
**      CALL FUNCTION 'BAPI_NETWORK_GETINFO'
**        TABLES
**          i_network_list = e_network
**          e_component    = e_component.


      LOOP AT e_network.        " Added by Siva 15.12.2020

        CALL FUNCTION 'BAPI_NETWORK_COMP_GETDETAIL'
          EXPORTING
            number              = e_network-network
*           MAX_ROWS              = 0
* IMPORTING
*           RETURN                =
          TABLES
*           I_ACTIVITY_RANGE      =
*           I_COMPONENTS_ID       =
            e_components_detail = e_component1.


        APPEND LINES OF e_component1 TO e_component.

        CLEAR e_network.
        REFRESH : e_component1 , e_component1[].
      ENDLOOP.


      LOOP AT e_component.
        READ TABLE it_zmmt001_temp WITH KEY rsnum = e_component-reserv_no rspos
 = e_component-res_item.
```

```
        IF sy-subrc NE 0.
          DELETE e_component.
        ENDIF.
      ENDLOOP.


      SELECT banfn bnfpo matnr menge frgdt badat bednr
        INTO TABLE int_eban                     "PR Items
      FROM eban
      FOR ALL ENTRIES IN e_component
      WHERE banfn = e_component-preq_no
        AND bnfpo = e_component-preq_item
        AND loekz = ''.

      IF int_eban[] IS NOT INITIAL.
        SELECT b~ebeln b~lifnr b~bedat b~eq_eindt a~banfn a~bnfpo
          INTO TABLE int_ekko
        FROM ekpo AS a
          INNER JOIN ekko AS b ON ( b~ebeln = a~ebeln )
          FOR ALL ENTRIES IN int_eban
        WHERE a~banfn = int_eban-banfn
          AND a~bnfpo = int_eban-bnfpo
          AND a~loekz = ''.
      ENDIF.
      IF int_ekko[] IS NOT INITIAL.
        SELECT lifnr name1
          FROM lfa1 INTO TABLE int_lfa1          "Vendor Name
          FOR ALL ENTRIES IN int_ekko
          WHERE lifnr = int_ekko-lifnr.

        SELECT ebeln ebelp eindt
          FROM eket INTO TABLE int_eket          "Delivery Date
          FOR ALL ENTRIES IN int_ekko
          WHERE ebeln = int_ekko-ebeln
          AND ebelp = int_ekko-ebelp.
        SORT int_eket BY ebeln ebelp eindt.
      ENDIF.

      LOOP AT int_details.
        CLEAR e_component.
        READ TABLE e_component WITH KEY reserv_no = int_details-rsnum  res_item
  = int_details-rspos.

        CLEAR int_eban.
        READ TABLE int_eban WITH KEY banfn = e_component-preq_no bnfpo = e_comp
onent-preq_item.
        int_details-banfn = int_eban-banfn.
        int_details-bnfpo = int_eban-bnfpo.
        int_details-prdat = int_eban-erdat.
        int_details-frgdt = int_eban-frgdt.

        CLEAR int_ekko.
        READ TABLE int_ekko WITH KEY banfn = int_eban-banfn bnfpo = int_eban-bn
```

```
fpo.
        int_details-ebeln = int_ekko-ebeln.
        int_details-ebelp = int_ekko-ebelp.
        int_details-bedat = int_ekko-bedat.
        int_details-lifnr = int_ekko-lifnr.

        CLEAR int_lfa1.
        READ TABLE int_lfa1 WITH KEY lifnr = int_ekko-lifnr.
        int_details-name1 = int_lfa1-name1.

        CLEAR int_eket.
        READ TABLE int_eket WITH KEY ebeln = int_ekko-ebeln ebelp = int_ekko-eb
elp.
        int_details-eindt = int_eket-eindt.

        MODIFY int_details TRANSPORTING banfn bnfpo prdat frgdt ebeln bedat lif
nr name1 eindt.

      ENDLOOP.


******************For Uploaded POs********************
      CLEAR: int_ekko, int_ekko[], int_ekkn, int_ekkn[],int_ekpo, int_ekpo[],in
t_ekko,int_ekko[],int_lfa1,int_lfa1[],int_eket,int_eket[].
      SELECT ebeln ebelp ps_psp_pnr
        FROM ekkn INTO TABLE int_ekkn                      "PO Items WBS
        FOR ALL ENTRIES IN int_details
        WHERE ps_psp_pnr = int_details-posnr
        AND loekz = ''.
      IF int_ekkn[] IS NOT INITIAL.
        REFRESH int_ekpo.
        SELECT ebeln ebelp matnr menge meins banfn bnfpo
          FROM ekpo INTO TABLE int_ekpo                   "PO Items MATNR
          FOR ALL ENTRIES IN int_ekkn
          WHERE ebeln = int_ekkn-ebeln
            AND ebelp = int_ekkn-ebelp
            AND lgort NE 'DD99'
            AND loekz = ''.
        LOOP AT int_ekpo.
          READ TABLE int_ekkn WITH KEY ebeln = int_ekpo-ebeln ebelp = int_ekpo-
ebelp.
          IF sy-subrc = 0.
            MOVE int_ekkn-ps_psp_pnr TO int_ekpo-ps_psp_pnr.
            MODIFY int_ekpo TRANSPORTING ps_psp_pnr.
          ENDIF.
        ENDLOOP.

        LOOP AT int_details WHERE banfn IS INITIAL.
          READ TABLE int_ekpo WITH KEY matnr = int_details-zdesno ps_psp_pnr =
int_details-posnr.
          IF sy-subrc = 0.
            MOVE int_ekpo-ebeln TO int_details-ebeln.
            MOVE int_ekpo-ebelp TO int_details-ebelp.
            MOVE int_ekpo-menge TO int_details-pomenge.
            MOVE int_ekpo-meins TO int_details-pomeins.
            MODIFY int_details TRANSPORTING ebeln ebelp pomenge pomeins.
          ENDIF.
        ENDLOOP.
      ENDIF.

      IF int_details[] IS NOT INITIAL.
        SELECT ebeln lifnr bedat eq_eindt
          FROM ekko INTO TABLE int_ekko          "PO Header
          FOR ALL ENTRIES IN int_details
          WHERE ebeln = int_details-ebeln.
      ENDIF.
      IF int_ekko[] IS NOT INITIAL.
        SELECT lifnr name1
          FROM lfa1 INTO TABLE int_lfa1          "Vendor Name
          FOR ALL ENTRIES IN int_ekko
          WHERE lifnr = int_ekko-lifnr.

        SELECT ebeln ebelp eindt
          FROM eket INTO TABLE int_eket          "Delivery Date
          FOR ALL ENTRIES IN int_ekko
          WHERE ebeln = int_ekko-ebeln
          AND ebelp = int_ekko-ebelp.
        SORT int_eket BY ebeln ebelp eindt.
      ENDIF.

      LOOP AT int_details.
        READ TABLE int_ekko WITH KEY ebeln = int_details-ebeln.
        IF sy-subrc = 0.
          MOVE int_ekko-bedat TO int_details-bedat.
          MOVE int_ekko-lifnr TO int_details-lifnr.
          MODIFY int_details TRANSPORTING bedat lifnr.
        ENDIF.
        READ TABLE int_lfa1 WITH KEY lifnr = int_details-lifnr.
        IF sy-subrc = 0.
          MOVE int_lfa1-name1 TO int_details-name1.
          MODIFY int_details TRANSPORTING name1.
        ENDIF.
        READ TABLE int_eket WITH KEY ebeln = int_details-ebeln  ebelp = int_ekk
o-ebelp.
        IF sy-subrc = 0.
          MOVE int_eket-eindt TO int_details-eindt.
          MODIFY int_details TRANSPORTING eindt.
        ENDIF.
      ENDLOOP.
********************Uploaded POs*********************

    ENDIF.
    IF ( i_col = 'INT_COUNT-PPCTOT' OR i_col = 'INT_COUNT-PPCPEN' ) AND int_det
ails[] IS NOT INITIAL.
      CLEAR int_mspr[].
      SELECT matnr werks lgort charg pspnr prlab
```

```
        FROM mspr INTO TABLE int_mspr
        FOR ALL ENTRIES IN int_details
        WHERE matnr = int_details-zdesno AND pspnr = int_details-pspnr .
      LOOP AT int_details.
        CLEAR int_mspr.
        READ TABLE int_mspr WITH KEY matnr = int_details-zdesno pspnr = int_det
ails-pspnr.
        IF int_mspr-lgort+0(2) EQ 'IS'.
          int_details-remarks = 'ICC Generated'.
          MODIFY int_details.
        ENDIF.
      ENDLOOP.
*         AND pspnr = int_ecm-pspnr
*         AND lgort = 'DD99' AND sobkz = 'Q' AND prlab GT int_ecm-menge.

    ENDIF.
    SORT int_details BY werks lgort pspnr partno.
    CLEAR sl.

    CALL FUNCTION 'DD_DOMA_GET'
      EXPORTING
        domain_name = 'ZSTATUS_DES'
*       GET_STATE    = 'M   '
        langu        = sy-langu
*       PRID         = 0
*       WITHTEXT     = 'X'
*       IMPORTING
*       DD01V_WA_A   =
*       DD01V_WA_N   =
*       GOT_STATE    =
      TABLES
        dd07v_tab_a = int_desc
        dd07v_tab_n = int_desc1
*       EXCEPTIONS
*       ILLEGAL_VALUE        = 1
*       OP_FAILURE   = 2
*       OTHERS       = 3
      .
    IF sy-subrc <> 0.
* Implement suitable error handling here
    ENDIF.

    LOOP AT int_details.
      sl = sl + 1.
      int_details-slno = sl.
      IF int_details-werks = 'P001'.
        int_details-divis = 'TC'.
      ELSEIF int_details-werks = 'P002'.
        int_details-divis = 'EM'.
      ELSEIF int_details-werks = 'P003'.
        int_details-divis = 'SG'.
      ELSEIF int_details-werks = 'P004'.
        int_details-divis = 'FOUNDRY'.
      ELSEIF int_details-werks = 'P006'.
        int_details-divis = 'HE & F'.
      ELSEIF int_details-werks = 'P007'.
        int_details-divis = 'TOOLS'.
      ELSEIF int_details-werks = 'P010'.
        int_details-divis = 'SPARES'.
      ELSEIF int_details-werks = 'P011'.
        int_details-divis = 'OR'.
      ELSEIF int_details-werks = 'P051'.
        int_details-divis = 'BM'.
      ELSEIF int_details-werks = 'P055'.
        int_details-divis = 'GT'.
      ELSEIF int_details-werks = 'P070'.
        int_details-divis = 'PUMPS'.
      ELSEIF int_details-werks = 'P090'.
        int_details-divis = 'PED'.
      ELSEIF int_details-werks+0(4) = 'P099'.
        CONCATENATE 'CMM' int_details-werks+4(5) INTO int_details-divis.
      ENDIF.

      CLEAR int_desc.
      READ TABLE int_desc WITH KEY domvalue_l = int_details-status.
      IF sy-subrc EQ 0.
        int_details-status_desc = int_desc-ddtext.
      ENDIF.

      MODIFY int_details TRANSPORTING divis slno status_desc.
    ENDLOOP.
  ENDIF.
  DATA:BEGIN OF int_eket_delv OCCURS 0,
         ebeln TYPE eket-ebeln,
         ebelp TYPE eket-ebelp,
         eindt TYPE eket-eindt,
       END OF int_eket_delv.
  SELECT ebeln ebelp eindt
    FROM eket
    INTO TABLE int_eket_delv
    FOR ALL ENTRIES IN int_details
    WHERE ebeln = int_details-ebeln
    AND ebelp = int_details-ebelp.
  SORT int_eket_delv BY ebeln ebelp eindt ASCENDING.
  LOOP AT int_details.
    CLEAR int_eket_delv.
    READ TABLE int_eket_delv WITH KEY ebeln = int_details-ebeln ebelp = int_det
ails-ebelp.
    int_details-eindt = int_eket_delv-eindt.
    MODIFY int_details.
  ENDLOOP.
**  Logic to get l2 & site req date
  break 6146252.
  DATA:int_zsdt135 TYPE TABLE OF zsdt135 WITH HEADER LINE.
  DATA: BEGIN OF int_pgma OCCURS 0,
          pgma    TYPE zecmt006-posnr,
          desno   TYPE zecmt006-zdesno,
          partno  TYPE zecmt006-partno,
          pgma5th TYPE zecmt006-posnr,
        END OF int_pgma.
  LOOP AT int_details.
    int_pgma-pgma = int_details-posnr.
    CALL FUNCTION 'ZPP_GET_5THLEVEL_PGMA'
      EXPORTING
        i_pspnr = int_pgma-pgma
      IMPORTING
        e_pspnr = int_pgma-pgma5th.
    .

    int_pgma-partno = int_details-partno.
    int_pgma-desno = int_details-zdesno.
    APPEND int_pgma.
  ENDLOOP.
  IF int_pgma[] IS NOT INITIAL.

    DATA:BEGIN OF int_prps_l2 OCCURS 0,
           pspnr          TYPE prps-pspnr,
           usr08          TYPE prps-usr08,
           zzsite_reqdate TYPE prps-zzsite_reqdate,
         END OF int_prps_l2.
    SELECT pspnr usr08 zzsite_reqdate
      FROM prps
      INTO TABLE int_prps_l2
      FOR ALL ENTRIES IN int_pgma
      WHERE pspnr = int_pgma-pgma5th.
    LOOP AT int_details.
      CLEAR int_pgma.
      READ TABLE int_pgma WITH KEY  pgma = int_details-posnr.
      CLEAR int_prps.
      READ TABLE int_prps_l2 WITH KEY pspnr = int_pgma-pgma5th.
      int_details-l2_date = int_prps_l2-usr08.
      int_details-site_reqdate = int_prps_l2-zzsite_reqdate.
      MODIFY int_details TRANSPORTING l2_date site_reqdate.
    ENDLOOP.

    SELECT *
      FROM zsdt135
      INTO TABLE int_zsdt135
      FOR ALL ENTRIES IN int_pgma
      WHERE posnr = int_pgma-pgma5th
      AND zdesno = int_pgma-desno
      AND partno = int_pgma-partno.
    IF sy-subrc EQ 0.
      SORT int_zsdt135 BY projn pspid pspnr posnr werks matnr zdesno partno des
_serial updated_date DESCENDING updated_time DESCENDING.
      DELETE ADJACENT DUPLICATES FROM int_zsdt135 COMPARING projn pspid pspnr p
osnr werks matnr zdesno partno des_serial .
      LOOP AT int_details.
*        CLEAR int_pgma.
*         READ TABLE int_pgma WITH KEY pgma = int_details-posnr.
        CLEAR int_zsdt135.
*         READ TABLE int_zsdt135 WITH KEY posnr = int_pgma-pgma5th zdesno = int
_pgma-desno partno = int_pgma-partno.
        READ TABLE int_zsdt135 WITH KEY posnr = int_details-posnr zdesno = int_
details-zdesno partno = int_details-partno.
        IF sy-subrc = 0.
          int_details-l2_date = int_zsdt135-l2date.
          int_details-site_reqdate = int_zsdt135-site_reqdate.
          MODIFY int_details TRANSPORTING l2_date site_reqdate.
        ENDIF.
      ENDLOOP.

    ENDIF.

  ENDIF.
ENDFORM.                    "DISP_DETAILS


*&---------------------------------------------------------------------*
*&      Form  BUILD_DETAILS
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
FORM build_details.

  REFRESH  int_detcat.

  CLEAR wa_detcat.
  wa_detcat-tabname = 'INT_DETAILS'.
  wa_detcat-fieldname = 'SLNO'.
  wa_detcat-outputlen =  5.
*  WA_DETCAT-COL_POS      = 0.
  wa_detcat-key          = 'X'.
  wa_detcat-seltext_l = 'SlNo'.
  APPEND wa_detcat TO int_detcat.

  CLEAR wa_detcat.
  wa_detcat-tabname = 'INT_DETAILS'.
  wa_detcat-fieldname = 'WERKS'.
  wa_detcat-outputlen =  10.
*  WA_DETCAT-COL_POS      = 0.
  wa_detcat-key          = 'X'.
  wa_detcat-seltext_l = 'Plant'.
  APPEND wa_detcat TO int_detcat.

  CLEAR wa_detcat.
  wa_detcat-tabname = 'INT_DETAILS'.
  wa_detcat-fieldname = 'DIVIS'.
  wa_detcat-outputlen =  10.
*  WA_DETCAT-COL_POS      = 0.
  wa_detcat-key          = 'X'.
*  WA_DETCAT-HOTSPOT = 'X'.
*  WA_DETCAT-SELTEXT_M  = 'S O '.
  wa_detcat-seltext_l = 'Division'.
  APPEND wa_detcat TO int_detcat.

  CLEAR wa_detcat.
  wa_detcat-tabname = 'INT_DETAILS'.
  wa_detcat-fieldname = 'LGORT'.
  wa_detcat-outputlen =  6.
*  WA_DETCAT-COL_POS      = 0.
  wa_detcat-key          = 'X'.
*  WA_DETCAT-HOTSPOT = 'X'.
*  WA_DETCAT-SELTEXT_M  = 'S O '.
  wa_detcat-seltext_l = 'Store'.
  APPEND wa_detcat TO int_detcat.

  CLEAR wa_detcat.
  wa_detcat-tabname = 'INT_DETAILS'.
  wa_detcat-fieldname = 'PSPID'.
  wa_detcat-outputlen =  15.
*  WA_DETCAT-HOTSPOT = 'X'.
  wa_detcat-seltext_l = 'Salesorder'.
  wa_detcat-edit_mask = '==ABPSP'.
  APPEND wa_detcat TO int_detcat.

  CLEAR wa_detcat.
  wa_detcat-tabname = 'INT_DETAILS'.
  wa_detcat-fieldname = 'PSPNR'.
  wa_detcat-outputlen =  13.
*  WA_DETCAT-HOTSPOT = 'X'.
  wa_detcat-seltext_l = 'Workorder'.
  wa_detcat-edit_mask = '==ABPSP'.
  APPEND wa_detcat TO int_detcat.

  CLEAR wa_detcat.
  wa_detcat-tabname = 'INT_DETAILS'.
  wa_detcat-fieldname = 'POSNR'.
  wa_detcat-outputlen =  18.
*  WA_DETCAT-HOTSPOT = 'X'.
  wa_detcat-seltext_l = 'PGMA WBS Element'.
  wa_detcat-edit_mask = '==ABPSP'.
  APPEND wa_detcat TO int_detcat.

  CLEAR wa_detcat.
  wa_detcat-tabname = 'INT_DETAILS'.
  wa_detcat-fieldname = 'POST1'.
  wa_detcat-outputlen =  35.
*  WA_DETCAT-HOTSPOT = 'X'.
  wa_detcat-seltext_l = 'PGMA Description'.
  APPEND wa_detcat TO int_detcat.

  CLEAR wa_detcat.
  wa_detcat-tabname = 'INT_DETAILS'.
  wa_detcat-fieldname = 'ZDESNO'.
```

```
wa_detcat-outputlen =  19.
wa_detcat-seltext_l = 'Desno'.
APPEND wa_detcat TO int_detcat.

CLEAR wa_detcat.
wa_detcat-tabname = 'INT_DETAILS'.
wa_detcat-fieldname = 'MAKTX'.
wa_detcat-outputlen =  41.
wa_detcat-seltext_l = 'Material Description'.
APPEND wa_detcat TO int_detcat.

CLEAR wa_detcat.
wa_detcat-tabname = 'INT_DETAILS'.
wa_detcat-fieldname = 'DUTYPE'.
wa_detcat-outputlen =  4.
wa_detcat-seltext_l = 'DU'.
APPEND wa_detcat TO int_detcat.

CLEAR wa_detcat.
wa_detcat-tabname = 'INT_DETAILS'.
wa_detcat-fieldname = 'MENGE'.
wa_detcat-outputlen =  10.
wa_detcat-seltext_l = 'Qty'.
APPEND wa_detcat TO int_detcat.

CLEAR wa_detcat.
wa_detcat-tabname = 'INT_DETAILS'.
wa_detcat-fieldname = 'MEINS'.
wa_detcat-outputlen =  4.
wa_detcat-seltext_l = 'UOM'.
APPEND wa_detcat TO int_detcat.

CLEAR wa_detcat.
wa_detcat-tabname = 'INT_DETAILS'.
wa_detcat-fieldname = 'ZZNETWT'.
wa_detcat-outputlen =  12.
wa_detcat-seltext_l = 'Total Weight'.
APPEND wa_detcat TO int_detcat.

CLEAR wa_detcat.
wa_detcat-tabname = 'INT_DETAILS'.
wa_detcat-fieldname = 'PARTNO'.
wa_detcat-outputlen =  9.
wa_detcat-seltext_l = 'Part No'.
APPEND wa_detcat TO int_detcat.
CLEAR wa_detcat.
IF  colsel <> 'INT_COUNT-PPCTOT'.
  CLEAR wa_detcat.
  wa_detcat-tabname = 'INT_DETAILS'.
  wa_detcat-fieldname = 'ITEM'.
  wa_detcat-outputlen =  6.
  wa_detcat-seltext_l = 'Sales Order Item'.
  APPEND wa_detcat TO int_detcat.
  ENDIF.
  IF ( colsel <> 'INT_COUNT-PPCTOT' AND colsel <> 'INT_COUNT-PPCPEN' AND colsel
 <> 'INT_COUNT-TOTDESP' AND colsel NE 'INT_COUNT-TOTFCWONO' ) .
    CLEAR wa_detcat.
    wa_detcat-tabname = 'INT_DETAILS'.
    wa_detcat-fieldname = 'RSNUM'.
    wa_detcat-outputlen =  11.
    wa_detcat-seltext_l = 'Reservation No'.
    APPEND wa_detcat TO int_detcat.

    CLEAR wa_detcat.
    wa_detcat-tabname = 'INT_DETAILS'.
    wa_detcat-fieldname = 'RSPOS'.
    wa_detcat-outputlen =  6.
    wa_detcat-seltext_l = 'Res Item'.
    APPEND wa_detcat TO int_detcat.


    CLEAR wa_detcat.
    wa_detcat-tabname = 'INT_DETAILS'.
    wa_detcat-fieldname = 'ZSTATU1'.
    wa_detcat-outputlen =  4.
    wa_detcat-seltext_l = 'Status'.
    APPEND wa_detcat TO int_detcat.

    CLEAR wa_detcat.
    wa_detcat-tabname = 'INT_DETAILS'.
    wa_detcat-fieldname = 'KZEAR'.
    wa_detcat-outputlen =  2.
    wa_detcat-seltext_l = 'FI Ind'.
    APPEND wa_detcat TO int_detcat.

    IF colsel = 'INT_COUNT-NOTCDC' .
      CLEAR wa_detcat.
      wa_detcat-tabname = 'INT_DETAILS'.
      wa_detcat-fieldname = 'ZGPNO'.
      wa_detcat-outputlen =  11.
      wa_detcat-seltext_l = 'GP No'.
      APPEND wa_detcat TO int_detcat.

      CLEAR wa_detcat.
      wa_detcat-tabname = 'INT_DETAILS'.
      wa_detcat-fieldname = 'ZGPDATE'.
      wa_detcat-outputlen =  10.
      wa_detcat-seltext_l = 'GP Dt'.
      APPEND wa_detcat TO int_detcat.
    ENDIF.
  ENDIF.

  IF colsel = 'INT_COUNT-TOTTOT' OR colsel = 'INT_COUNT-TOTPEN' OR colsel = 'IN
T_COUNT-TOTDESP'. "added by 6194419 on 07042021
    CLEAR wa_detcat.
    wa_detcat-tabname = 'INT_DETAILS'.
    wa_detcat-fieldname = 'ZGPNO'.
    wa_detcat-outputlen =  11.
    wa_detcat-seltext_l = 'GP No'.
    APPEND wa_detcat TO int_detcat.

    CLEAR wa_detcat.
    wa_detcat-tabname = 'INT_DETAILS'.
    wa_detcat-fieldname = 'ZGPDATE'.
    wa_detcat-outputlen =  10.
    wa_detcat-seltext_l = 'GP Dt'.
    APPEND wa_detcat TO int_detcat.
  ENDIF.

  IF colsel = 'INT_COUNT-TOTPEN' .
    CLEAR wa_detcat.
    wa_detcat-tabname = 'INT_DETAILS'.
    wa_detcat-fieldname = 'ACK_TXT'.
    wa_detcat-outputlen =  4.
    wa_detcat-seltext_l = 'CDC Status'.
    APPEND wa_detcat TO int_detcat.

    CLEAR wa_detcat.
    wa_detcat-tabname = 'INT_DETAILS'.
    wa_detcat-fieldname = 'ITEM'.
    wa_detcat-outputlen =  6.
    wa_detcat-seltext_l = 'Sales Order Item'.
    APPEND wa_detcat TO int_detcat.

    CLEAR wa_detcat.
    wa_detcat-tabname = 'INT_DETAILS'.
    wa_detcat-fieldname = 'ZLINE_CREAT_DATE'.
    wa_detcat-outputlen =  10.
    wa_detcat-seltext_l = 'Item Create Dt'.
    APPEND wa_detcat TO int_detcat.
  ENDIF.

  IF ( colsel = 'INT_COUNT-GPISSU' OR colsel = 'INT_COUNT-MPCTOT' OR colsel = '
INT_COUNT-SUBTOT' ) .
    CLEAR wa_detcat.
    wa_detcat-tabname = 'INT_DETAILS'.
    wa_detcat-fieldname = 'ZPICKSLPNO'.
    wa_detcat-outputlen =  11.
    wa_detcat-seltext_l = 'Picklist No'.
    APPEND wa_detcat TO int_detcat.

    CLEAR wa_detcat.
    wa_detcat-tabname = 'INT_DETAILS'.
    wa_detcat-fieldname = 'ZPLBLDAT'.
    wa_detcat-outputlen =  10.
    wa_detcat-seltext_l = 'Picklist Dt'.
    APPEND wa_detcat TO int_detcat.

    CLEAR wa_detcat.
    wa_detcat-tabname = 'INT_DETAILS'.
    wa_detcat-fieldname = 'ZPICKSLPNO1'.
    wa_detcat-outputlen =  11.
    wa_detcat-seltext_l = 'Pickslip No'.
    APPEND wa_detcat TO int_detcat.

    CLEAR wa_detcat.
    wa_detcat-tabname = 'INT_DETAILS'.
    wa_detcat-fieldname = 'ZPSBLDAT'.
    wa_detcat-outputlen =  10.
    wa_detcat-seltext_l = 'Pickslip Dt'.
    APPEND wa_detcat TO int_detcat.

    CLEAR wa_detcat.
    wa_detcat-tabname = 'INT_DETAILS'.
    wa_detcat-fieldname = 'ZGPNO'.
    wa_detcat-outputlen =  11.
    wa_detcat-seltext_l = 'GP No'.
    APPEND wa_detcat TO int_detcat.

    CLEAR wa_detcat.
    wa_detcat-tabname = 'INT_DETAILS'.
    wa_detcat-fieldname = 'ZGPDATE'.
    wa_detcat-outputlen =  10.
    wa_detcat-seltext_l = 'GP Dt'.
    APPEND wa_detcat TO int_detcat.
  ENDIF.

  IF colsel = 'INT_COUNT-STRPEN'.
    CLEAR wa_detcat.
    wa_detcat-tabname = 'INT_DETAILS'.
    wa_detcat-fieldname = 'ZPICKSLPNO'.
    wa_detcat-outputlen =  11.
    wa_detcat-seltext_l = 'Picklist No'.
    APPEND wa_detcat TO int_detcat.

    CLEAR wa_detcat.
    wa_detcat-tabname = 'INT_DETAILS'.
    wa_detcat-fieldname = 'ZPLBLDAT'.
    wa_detcat-outputlen =  10.
    wa_detcat-seltext_l = 'Picklist Dt'.
    APPEND wa_detcat TO int_detcat.

    CLEAR wa_detcat.
    wa_detcat-tabname = 'INT_DETAILS'.
    wa_detcat-fieldname = 'ZPICKSLPNO1'.
    wa_detcat-outputlen =  11.
    wa_detcat-seltext_l = 'Pickslip No'.
    APPEND wa_detcat TO int_detcat.

    CLEAR wa_detcat.
    wa_detcat-tabname = 'INT_DETAILS'.
    wa_detcat-fieldname = 'ZPSBLDAT'.
    wa_detcat-outputlen =  10.
    wa_detcat-seltext_l = 'Pickslip Dt'.
    APPEND wa_detcat TO int_detcat.
  ENDIF.

  IF colsel = 'INT_COUNT-CDCACK'.
    CLEAR wa_detcat.
    wa_detcat-tabname = 'INT_DETAILS'.
    wa_detcat-fieldname = 'ZPICKSLPNO'.
    wa_detcat-outputlen =  11.
    wa_detcat-seltext_l = 'Picklist No'.
    APPEND wa_detcat TO int_detcat.

    CLEAR wa_detcat.
    wa_detcat-tabname = 'INT_DETAILS'.
    wa_detcat-fieldname = 'ZPLBLDAT'.
    wa_detcat-outputlen =  10.
    wa_detcat-seltext_l = 'Picklist Dt'.
    APPEND wa_detcat TO int_detcat.

    CLEAR wa_detcat.
    wa_detcat-tabname = 'INT_DETAILS'.
    wa_detcat-fieldname = 'ZGPNO'.
    wa_detcat-outputlen =  11.
    wa_detcat-seltext_l = 'GP No'.
    APPEND wa_detcat TO int_detcat.

    CLEAR wa_detcat.
    wa_detcat-tabname = 'INT_DETAILS'.
    wa_detcat-fieldname = 'ZGPDATE'.
    wa_detcat-outputlen =  10.
    wa_detcat-seltext_l = 'GP Dt'.
    APPEND wa_detcat TO int_detcat.


    CLEAR wa_detcat.
    wa_detcat-tabname = 'INT_DETAILS'.
    wa_detcat-fieldname = 'ZLINE_CREAT_DATE'.
    wa_detcat-outputlen =  10.
    wa_detcat-seltext_l = 'SO Copy Dt'.
    APPEND wa_detcat TO int_detcat.
  ENDIF.

  IF ( colsel = 'INT_COUNT-CDCPAK' OR colsel = 'INT_COUNT-CDCPQ' OR colsel = 'I
NT_COUNT-TOTPEN' OR colsel = 'INT_COUNT-TOTDESP' OR colsel = 'INT_COUNT-TOTFCWO
NO'
    OR colsel = 'INT_COUNT-TOTTOT' ) ."TOTTOT added by 6194419 on 07042021
    CLEAR wa_detcat.
    wa_detcat-tabname = 'INT_DETAILS'.
    wa_detcat-fieldname = 'DELIV'.
    wa_detcat-outputlen =  15.
    wa_detcat-seltext_l = 'OBD No'.
    wa_detcat-edit_mask = '==ALPHA'.
```

```
APPEND wa_detcat TO int_detcat.

CLEAR wa_detcat.
wa_detcat-tabname = 'INT_DETAILS'.
wa_detcat-fieldname = 'DELDT'.
wa_detcat-outputlen =  10.
wa_detcat-seltext_l = 'OBD Dt'.
APPEND wa_detcat TO int_detcat.

CLEAR wa_detcat.
wa_detcat-tabname = 'INT_DETAILS'.
wa_detcat-fieldname = 'CDC_QUALITY_STATUS'.
wa_detcat-outputlen =  3.
wa_detcat-seltext_l = 'Quality Cleared Status'.
APPEND wa_detcat TO int_detcat.


CLEAR wa_detcat.
wa_detcat-tabname = 'INT_DETAILS'.
wa_detcat-fieldname = 'ZZMDCC'.
wa_detcat-outputlen =  50.
wa_detcat-seltext_l = 'MDCC No.'.
APPEND wa_detcat TO int_detcat.

IF colsel NE 'INT_COUNT-TOTFCWONO'.
  CLEAR wa_detcat.
  wa_detcat-tabname = 'INT_DETAILS'.
  wa_detcat-fieldname = 'PKGNO'.
  wa_detcat-outputlen =  30.
  wa_detcat-seltext_l = 'Export Package No'.
  APPEND wa_detcat TO int_detcat.

  CLEAR wa_detcat.
  wa_detcat-tabname = 'INT_DETAILS'.
  wa_detcat-fieldname = 'BOLNR'.
  wa_detcat-outputlen =  30.
  wa_detcat-seltext_l = 'Bill of Lading'.
  APPEND wa_detcat TO int_detcat.

  CLEAR wa_detcat.
  wa_detcat-tabname = 'INT_DETAILS'.
  wa_detcat-fieldname = 'PODAT'.
  wa_detcat-outputlen =  10.
  wa_detcat-seltext_l = 'B/L Date'.
  APPEND wa_detcat TO int_detcat.


  CLEAR wa_detcat.
  wa_detcat-tabname = 'INT_DETAILS'.
  wa_detcat-fieldname = 'CDC_OFF'.
  wa_detcat-outputlen =  10.
  wa_detcat-seltext_l = 'CDC Official'.
  APPEND wa_detcat TO int_detcat.
```

```
      CLEAR wa_detcat.
      wa_detcat-tabname = 'INT_DETAILS'.
      wa_detcat-fieldname = 'CDC_NAME'.
      wa_detcat-outputlen =  10.
      wa_detcat-seltext_l = 'CDC Off Name'.
      APPEND wa_detcat TO int_detcat.

    ENDIF.
  ENDIF.
  IF colsel = 'INT_COUNT-MPCTOT' OR colsel = 'INT_COUNT-PPCPEN'.
    CLEAR wa_detcat.
    wa_detcat-tabname = 'INT_DETAILS'.
    wa_detcat-fieldname = 'REMARKS'.
    wa_detcat-outputlen =  10.
    wa_detcat-seltext_l = 'Remarks'.
    APPEND wa_detcat TO int_detcat.
  ENDIF.
  IF colsel = 'INT_COUNT-TOTDESP' OR colsel = 'INT_COUNT-TOTFCWONO'.
    CLEAR wa_detcat.
    wa_detcat-tabname = 'INT_DETAILS'.
    wa_detcat-fieldname = 'TKNUM'.
    wa_detcat-outputlen =  11.
    wa_detcat-seltext_l = 'Shipment No'.
    wa_detcat-edit_mask = '==ALPHA'.
    APPEND wa_detcat TO int_detcat.

    CLEAR wa_detcat.
    wa_detcat-tabname = 'INT_DETAILS'.
    wa_detcat-fieldname = 'DATBG'.
    wa_detcat-outputlen =  10.
    wa_detcat-seltext_l = 'Shipment Dt'.
    APPEND wa_detcat TO int_detcat.

    CLEAR wa_detcat.
    wa_detcat-tabname = 'INT_DETAILS'.
    wa_detcat-fieldname = 'EXTI1'.
    wa_detcat-outputlen =  11.
    wa_detcat-seltext_l = 'LR No'.
    APPEND wa_detcat TO int_detcat.

    CLEAR wa_detcat.
    wa_detcat-tabname = 'INT_DETAILS'.
    wa_detcat-fieldname = 'DATBG'.
    wa_detcat-outputlen =  10.
    wa_detcat-seltext_l = 'LR Dt'.
    APPEND wa_detcat TO int_detcat.

    CLEAR wa_detcat.
    wa_detcat-tabname = 'INT_DETAILS'.
    wa_detcat-fieldname = 'SIGNI'.
    wa_detcat-outputlen =  13.
    wa_detcat-seltext_l = 'Vehicle No'.
    APPEND wa_detcat TO int_detcat.

    CLEAR wa_detcat.
    wa_detcat-tabname = 'INT_DETAILS'.
    wa_detcat-fieldname = 'TDLNR'.
    wa_detcat-outputlen =  6.
    wa_detcat-seltext_l = 'Trns'.
    APPEND wa_detcat TO int_detcat.

    CLEAR wa_detcat.
    wa_detcat-tabname = 'INT_DETAILS'.
    wa_detcat-fieldname = 'REMARKS'.
    wa_detcat-outputlen =  10.
    wa_detcat-seltext_l = 'Remarks'.
    APPEND wa_detcat TO int_detcat.

  ENDIF.

  IF ( colsel = 'INT_COUNT-PURPEN' OR colsel = 'INT_COUNT-PURTOT' OR colsel = '
INT_COUNT-SUBPEN' OR colsel = 'INT_COUNT-SUBTOT' OR colsel = 'INT_COUNT-MPCPEN'
 OR colsel = 'INT_COUNT-MPCTOT' ).
    CLEAR wa_detcat.
    wa_detcat-tabname = 'INT_DETAILS'.
    wa_detcat-fieldname = 'BANFN'.
    wa_detcat-outputlen =  10.
    wa_detcat-seltext_l = 'PR No'.
    APPEND wa_detcat TO int_detcat.

*     CLEAR WA_DETCAT.
*     WA_DETCAT-TABNAME = 'INT_DETAILS'.
*     WA_DETCAT-FIELDNAME = 'PRDAT'.
*     WA_DETCAT-OUTPUTLEN =  10.
*     WA_DETCAT-SELTEXT_L = 'PR Dt'.
*     APPEND WA_DETCAT TO INT_DETCAT.

    CLEAR wa_detcat.
    wa_detcat-tabname = 'INT_DETAILS'.
    wa_detcat-fieldname = 'FRGDT'.
    wa_detcat-outputlen =  10.
    wa_detcat-seltext_l = 'PR Rel Dt'.
    APPEND wa_detcat TO int_detcat.

    CLEAR wa_detcat.
    wa_detcat-tabname = 'INT_DETAILS'.
    wa_detcat-fieldname = 'EBELN'.
    wa_detcat-outputlen =  10.
    wa_detcat-seltext_l = 'PO No'.
    APPEND wa_detcat TO int_detcat.

    CLEAR wa_detcat.
    wa_detcat-tabname = 'INT_DETAILS'.
    wa_detcat-fieldname = 'EBELP'.
    wa_detcat-outputlen =  10.
    wa_detcat-seltext_l = 'PO Item No'.
    APPEND wa_detcat TO int_detcat.

    CLEAR wa_detcat.
    wa_detcat-tabname = 'INT_DETAILS'.
    wa_detcat-fieldname = 'BEDAT'.
    wa_detcat-outputlen =  10.
    wa_detcat-seltext_l = 'PO Dt'.
    APPEND wa_detcat TO int_detcat.

    CLEAR wa_detcat.
    wa_detcat-tabname = 'INT_DETAILS'.
    wa_detcat-fieldname = 'NAME1'.
    wa_detcat-outputlen =  40.
    wa_detcat-seltext_l = 'Vendor Name'.
    APPEND wa_detcat TO int_detcat.

    CLEAR wa_detcat.
    wa_detcat-tabname = 'INT_DETAILS'.
    wa_detcat-fieldname = 'EINDT'.
    wa_detcat-outputlen =  10.
    wa_detcat-seltext_l = 'PO Deliv Dt'.
    APPEND wa_detcat TO int_detcat.

*     CLEAR WA_DETCAT.
*     WA_DETCAT-TABNAME = 'INT_DETAILS'.
*     WA_DETCAT-FIELDNAME = 'DATBG'.
*     WA_DETCAT-OUTPUTLEN =  10.
*     WA_DETCAT-SELTEXT_L = 'LR Dt'.
*     APPEND WA_DETCAT TO INT_DETCAT.

    CLEAR wa_detcat.
    wa_detcat-tabname = 'INT_DETAILS'.
    wa_detcat-fieldname = 'PUR_OFF'.
    wa_detcat-outputlen =  10.
    wa_detcat-seltext_l = 'Purchase Officer'.
    APPEND wa_detcat TO int_detcat.

    CLEAR wa_detcat.
    wa_detcat-tabname = 'INT_DETAILS'.
    wa_detcat-fieldname = 'PUR_NAME'.
    wa_detcat-outputlen =  10.
    wa_detcat-seltext_l = 'Pur Officer'.
    APPEND wa_detcat TO int_detcat.

  ENDIF.

  IF ( colsel = 'INT_COUNT-PURPEN' OR colsel = 'INT_COUNT-PURTOT' ).

    CLEAR wa_detcat.
    wa_detcat-tabname = 'INT_DETAILS'.
    wa_detcat-fieldname = 'RDRNO'.
    wa_detcat-outputlen =  10.
    wa_detcat-seltext_l = 'RDR No'.
    APPEND wa_detcat TO int_detcat.

    CLEAR wa_detcat.
    wa_detcat-tabname = 'INT_DETAILS'.
    wa_detcat-fieldname = 'RDRDT'.
    wa_detcat-outputlen =  10.
    wa_detcat-seltext_l = 'RDR Dt'.
    APPEND wa_detcat TO int_detcat.

    CLEAR wa_detcat.
    wa_detcat-tabname = 'INT_DETAILS'.
    wa_detcat-fieldname = 'RDRQT'.
    wa_detcat-outputlen =  10.
    wa_detcat-seltext_l = 'RDR Qty'.
    wa_detcat-no_zero = 'X'.
    APPEND wa_detcat TO int_detcat.

    CLEAR wa_detcat.
    wa_detcat-tabname = 'INT_DETAILS'.
    wa_detcat-fieldname = 'RLRNO'.
    wa_detcat-outputlen =  10.
    wa_detcat-seltext_l = 'LR No'.
    APPEND wa_detcat TO int_detcat.

    CLEAR wa_detcat.
    wa_detcat-tabname = 'INT_DETAILS'.
    wa_detcat-fieldname = 'RLRDT'.
    wa_detcat-outputlen =  10.
    wa_detcat-seltext_l = 'LR Dt'.
    APPEND wa_detcat TO int_detcat.

    CLEAR wa_detcat.
    wa_detcat-tabname = 'INT_DETAILS'.
    wa_detcat-fieldname = 'ZZTNM'.
    wa_detcat-outputlen =  10.
    wa_detcat-seltext_l = 'Trans'.
    APPEND wa_detcat TO int_detcat.



  ENDIF.

  CLEAR wa_detcat.
  wa_detcat-tabname = 'INT_DETAILS'.
  wa_detcat-fieldname = 'PKGNO_PGMA'.
  wa_detcat-outputlen =  30.
  wa_detcat-seltext_l = 'Package No'.
  APPEND wa_detcat TO int_detcat.

  CLEAR wa_detcat.
  wa_detcat-tabname = 'INT_DETAILS'.
```

```
  wa_detcat-fieldname = 'STATUS_DESC'.
  wa_detcat-outputlen =  25.
  wa_detcat-seltext_l = 'Delv Status'.
  APPEND wa_detcat TO int_detcat.

  CLEAR wa_detcat.
  wa_detcat-tabname = 'INT_DETAILS'.
  wa_detcat-fieldname = 'ANDAT'.
  wa_detcat-outputlen =  10.
  wa_detcat-seltext_l = 'DES CREATED ON'.
  APPEND wa_detcat TO int_detcat.

  IF sy-tcode = 'ZSDAGENCYPEND_DATE'.
    CLEAR wa_detcat.
    wa_detcat-tabname = 'INT_DETAILS'.
    wa_detcat-fieldname = 'PLAN_DESP_DATE'.
    wa_detcat-outputlen =  15.
    wa_detcat-seltext_l = 'Plan Desp Date'.
    wa_detcat-edit = 'X'.
    wa_detcat-datatype = 'DATS'.
*   wa_detcat-f4availabl = 'X'.
    wa_detcat-ref_tabname = 'SYST'.
    wa_detcat-ref_fieldname = 'DATUM'.
    APPEND wa_detcat TO int_detcat.
  ENDIF.

  CLEAR wa_detcat.
  wa_detcat-tabname = 'INT_DETAILS'.
  wa_detcat-fieldname = 'L2_DATE'.
  wa_detcat-outputlen =  10.
  wa_detcat-seltext_l = 'L2 Date'.
  wa_detcat-no_zero = 'X'.
  APPEND wa_detcat TO int_detcat.

  CLEAR wa_detcat.
  wa_detcat-tabname = 'INT_DETAILS'.
  wa_detcat-fieldname = 'SITE_REQDATE'.
  wa_detcat-outputlen =  10.
  wa_detcat-seltext_l = 'Site Req Dt'.
  wa_detcat-no_zero = 'X'.
  APPEND wa_detcat TO int_detcat.


  CALL FUNCTION 'REUSE_ALV_EVENTS_GET'
    EXPORTING
      i_list_type     = 4
*     I_CALLBACK_USER_COMMAND = 'HOT_SPOT'
    IMPORTING
      et_events       = it_events[]
    EXCEPTIONS
      list_type_wrong = 1
      OTHERS          = 2.
  READ TABLE it_events WITH KEY name = 'TOP_OF_PAGE'.
  IF sy-subrc = 0.
    it_events-form = 'TOP_OF_PAGE1'.
    MODIFY it_events INDEX sy-tabix TRANSPORTING form.
  ENDIF.

*  READ TABLE IT_EVENTS WITH KEY NAME = 'USER_COMMAND'.
*  IF SY-SUBRC = 0.
*    IT_EVENTS-FORM = 'USER_COMMAND'.
*    MODIFY IT_EVENTS INDEX SY-TABIX TRANSPORTING FORM.
*  ENDIF.

ENDFORM.                    "BUILD_DETAILS

*&---------------------------------------------------------------------*
*&      Form  DISP_DETAILS
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
FORM disp_details.
*  PERFORM GET_STATUS.
*  SORT INT_COUNT BY VBELN PSPNR PARTNO ZDESNO.
*  PERFORM GET_FIN_CONVERSIONS.
*  w_Title = 'hi'.
  IF int_details[] IS NOT INITIAL.

    gd_repid = sy-repid.
    CALL FUNCTION 'REUSE_ALV_GRID_DISPLAY'
      EXPORTING
        i_callback_program       = gd_repid
        i_callback_top_of_page   = 'TOP-OF-PAGE'  "SEE FORM
*        IS_LAYOUT                = GD_LAYOUT
        i_callback_pf_status_set = 'PFSTATUS'
        i_callback_user_command  = 'USER_COMMAND_MAT'
        i_background_id          = 'ALV_BACKGROUND'
*        I_GRID_TITLE             = W_TITLE
        it_fieldcat              = int_detcat[]
        it_events                = it_events[]
        i_save                   = 'U'
        is_variant               = gx_variant
      TABLES
        t_outtab                 = int_details
      EXCEPTIONS
        program_error            = 1
        OTHERS                   = 2.
    IF sy-subrc <> 0.
      MESSAGE ID sy-msgid TYPE sy-msgty NUMBER sy-msgno
              WITH sy-msgv1 sy-msgv2 sy-msgv3 sy-msgv4.
    ENDIF.
  ELSE.
    MESSAGE 'No Details are available' TYPE 'I'.
  ENDIF.
ENDFORM.                    "DISP_DETAILS
```

```
*&---------------------------------------------------------------------*
*&      Form  TOP_OF_PAGE
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
FORM top_of_page1.
*      ALV HEADER DECLARATIONS
  DATA: d_header      TYPE slis_t_listheader,
        da_header     TYPE slis_listheader,
        t_line        LIKE da_header-info,
        ld_lines      TYPE i,
        ld_linesc(10) TYPE c,
        det_head(60)  TYPE c.

  break 1896563.
** TITLE
  CLEAR det_head.
  IF colsel = 'INT_COUNT-MPCTOT'.
    det_head = 'MPC Total'.
  ELSEIF colsel = 'INT_COUNT-MPCPEN'.
    det_head = 'MPC Pendency'.
  ELSEIF colsel = 'INT_COUNT-PPCTOT'.
    det_head = 'PPC Total'.
  ELSEIF colsel = 'INT_COUNT-PPCPEN'.
    det_head = 'PPC Pendency'.
  ELSEIF colsel = 'INT_COUNT-SUBTOT'.
    det_head = 'SUBC Total'.
  ELSEIF colsel = 'INT_COUNT-SUBPEN'.
    det_head = 'SUBC Pendency'.
  ELSEIF colsel = 'INT_COUNT-PURTOT'.
    det_head = 'Purchase Total'.
  ELSEIF colsel = 'INT_COUNT-PURPEN'.
    det_head = 'Purchase Pendency'.
  ELSEIF colsel = 'INT_COUNT-STRPEN'.
    det_head = 'Stores Pendency'.
  ELSEIF colsel = 'INT_COUNT-NOTCDC'.
    det_head = 'Pendency Excluding CDC'.
  ELSEIF colsel = 'INT_COUNT-GPISSU'.
    det_head = 'Gatepass Issued (Not Acknowledged)'.
  ELSEIF colsel = 'INT_COUNT-CDCACK'.
    det_head = 'Materials Pending for Packing by CDC'.
  ELSEIF colsel = 'INT_COUNT-CMLPEN'.
    det_head = 'Pendency with Commercial'.
  ELSEIF colsel = 'INT_COUNT-CDCPQ'.
    det_head = 'CDC Packed Pend Quality'.
  ELSEIF colsel = 'INT_COUNT-CDCPAK'.
    det_head = 'CDC Packed Qual Cleared'.
  ELSEIF colsel = 'INT_COUNT-TOTTOT'.
    det_head = 'Total deliverables'.
  ELSEIF colsel = 'INT_COUNT-TOTPEN'.
    det_head = 'Total Pendency'.
  ELSEIF colsel = 'INT_COUNT-TOTDESP'.
```

```
    det_head = 'Total Despatches'.
  ELSEIF colsel = 'INT_COUNT-TOTFCWONO'.
    det_head = 'Foreign Currency Items'.
  ENDIF.
  CONCATENATE det_head ' for ' INTO det_head.
  IF rowsel = 'P001'.
    CONCATENATE det_head ' TC Division' INTO det_head.
  ELSEIF rowsel = 'P002'.
    CONCATENATE det_head ' EM Division' INTO det_head.
  ELSEIF rowsel = 'P003'.
    CONCATENATE det_head ' SG Division' INTO det_head.
  ELSEIF rowsel = 'P004'.
    CONCATENATE det_head ' Foundry Division' INTO det_head.
  ELSEIF rowsel = 'P006'.
    CONCATENATE det_head ' HE & F Division' INTO det_head.
  ELSEIF rowsel = 'P007'.
    CONCATENATE det_head ' Tools Division' INTO det_head.
  ELSEIF rowsel = 'P010'.
    CONCATENATE det_head ' Spares Division' INTO det_head.
  ELSEIF rowsel = 'P011'.
    CONCATENATE det_head ' OR Division' INTO det_head.
  ELSEIF rowsel = 'P051'.
    CONCATENATE det_head ' BM Division' INTO det_head.
  ELSEIF rowsel = 'P055'.
    CONCATENATE det_head ' GT Division' INTO det_head.
  ELSEIF rowsel = 'P070'.
    CONCATENATE det_head ' PUMPS Division' INTO det_head.
  ELSEIF rowsel = 'P090'.
    CONCATENATE det_head ' PED Division' INTO det_head.
  ELSEIF rowsel+0(4) = 'P099'.
    CONCATENATE det_head ' CMM' rowsel+4(5) ' Division' INTO det_head.
  ELSE.
    CONCATENATE det_head ' All Divisions' INTO det_head.
  ENDIF.

  break 1896563.

  CLEAR da_header.
  REFRESH d_header.

  da_header-typ  = 'H'.
  da_header-info = det_head.
  APPEND da_header TO d_header.
  CLEAR da_header.

  da_header-typ  = 'S'.
  da_header-key = 'Sales Order :'.
  da_header-info = p_vbeln.
  APPEND da_header TO d_header.
  CLEAR da_header.

  da_header-typ  = 'S'.
  da_header-key = 'Customer Name :'.
  da_header-info = t_name1.
  APPEND da_header TO d_header.
  CLEAR da_header.

  da_header-typ  = 'S'.
  da_header-key = 'Project Name :'.
  da_header-info = t_projname.
  APPEND da_header TO d_header.
  CLEAR da_header.

  da_header-typ  = 'S'.
  da_header-key = 'Status as on :'.
  da_header-info = rep_dt.
  APPEND da_header TO d_header.
  CLEAR da_header.

  CALL FUNCTION 'REUSE_ALV_COMMENTARY_WRITE'
    EXPORTING
      it_list_commentary = d_header.

ENDFORM.                    "TOP_OF_PAGE1
*&---------------------------------------------------------------------*
*&      Form  DISPLAY_FC_DETAILS
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
*  -->  p1        text
*  <--  p2        text
*----------------------------------------------------------------------*
FORM display_fc_details USING i_werks .
  CLEAR int_fcdetails.
  DATA: int_details_with_ptno    LIKE TABLE OF int_details WITH HEADER LINE,
        int_details_without_ptno LIKE TABLE OF int_details WITH HEADER LINE.
  break 1896563.

  int_details_with_ptno[] = int_details[].
  DELETE int_details_with_ptno WHERE partno EQ ''.
  int_details_without_ptno[] = int_details[].
  DELETE int_details_without_ptno WHERE partno NE ''.

  IF int_details_with_ptno[] IS NOT INITIAL.
    SELECT projn pspid pspnr posnr zdesno matnr werks partno des_serial meins m
aktx menge zznetwt total_price dutype status sales_indi goods_indi item
      INTO TABLE int_fcdetails
    FROM zecmt006
      FOR ALL ENTRIES IN int_details_with_ptno
    WHERE pspnr = int_details_with_ptno-fcwono
      AND zdesno = int_details_with_ptno-zdesno
      AND partno = int_details_with_ptno-partno.
  ENDIF.
  IF int_details_without_ptno[] IS NOT INITIAL.
    SELECT projn pspid pspnr posnr zdesno matnr werks partno des_serial meins m
aktx menge zznetwt total_price dutype status sales_indi goods_indi item
```

```
      APPENDING TABLE int_fcdetails
    FROM zecmt006
      FOR ALL ENTRIES IN int_details_without_ptno
    WHERE pspnr = int_details_without_ptno-fcwono
      AND zdesno = int_details_without_ptno-zdesno.
  ENDIF.

  IF int_details_with_ptno[] IS NOT INITIAL.
    SELECT rsnum rspos zstatu1 bwart xloek kzear zwono matnr pspel werks lgort
aufnr sortf zpickslpno zplbldat zpickslpno1 zpsbldat zgpno zgpdate
    FROM zmmt001
      INTO TABLE int_zmmt001
      FOR ALL ENTRIES IN int_details_with_ptno
    WHERE pspel = int_details_with_ptno-posnr
      AND matnr = int_details_with_ptno-zdesno
      AND sortf = int_details_with_ptno-partno
      AND xloek EQ ''
      AND bwart = '281'.
  ENDIF.
  IF int_details_without_ptno[] IS NOT INITIAL.
    SELECT rsnum rspos zstatu1 bwart xloek kzear zwono matnr pspel werks lgort
aufnr sortf zpickslpno zplbldat zpickslpno1 zpsbldat zgpno zgpdate
    FROM zmmt001
      APPENDING TABLE int_zmmt001
      FOR ALL ENTRIES IN int_details_without_ptno
    WHERE pspel = int_details_without_ptno-posnr
      AND matnr = int_details_without_ptno-zdesno
      AND xloek EQ ''
      AND bwart = '281'.
  ENDIF.

  LOOP AT int_fcdetails.
    CLEAR p_pspid.
    CALL FUNCTION 'CONVERSION_EXIT_ABPSP_OUTPUT'
      EXPORTING
        input  = int_fcdetails-pspid
      IMPORTING
        output = p_pspid.

    REPLACE ALL OCCURRENCES OF '-' IN p_pspid WITH ''.
    CONDENSE p_pspid NO-GAPS.
    int_fcdetails-sono = p_pspid.

    CLEAR int_details.
    READ TABLE int_details WITH KEY zdesno = int_fcdetails-zdesno fcwono = int_
fcdetails-pspnr.

    CLEAR int_zmmt001.
    READ TABLE int_zmmt001 WITH KEY pspel = int_details-posnr matnr = int_detai
ls-zdesno.
    IF sy-subrc EQ 0.
      int_fcdetails-werks = int_zmmt001-werks.
      int_fcdetails-lgort = int_zmmt001-lgort.
    ENDIF.

    IF int_fcdetails-werks = 'P099'.
      CONCATENATE int_fcdetails-werks '-' int_fcdetails-lgort INTO int_fcdetail
s-werks.
    ENDIF.

    sl = sl + 1.
    int_fcdetails-slno = sl.
    IF int_fcdetails-werks = 'P001'.
      int_fcdetails-divis = 'TC'.
    ELSEIF int_fcdetails-werks = 'P002'.
      int_fcdetails-divis = 'EM'.
    ELSEIF int_fcdetails-werks = 'P003'.
      int_fcdetails-divis = 'SG'.
    ELSEIF int_fcdetails-werks = 'P004'.
      int_fcdetails-divis = 'FOUNDRY'.
    ELSEIF int_fcdetails-werks = 'P006'.
      int_fcdetails-divis = 'HE & F'.
    ELSEIF int_fcdetails-werks = 'P007'.
      int_fcdetails-divis = 'TOOLS'.
    ELSEIF int_fcdetails-werks = 'P010'.
      int_fcdetails-divis = 'SPARES'.
    ELSEIF int_fcdetails-werks = 'P011'.
      int_fcdetails-divis = 'OR'.
    ELSEIF int_fcdetails-werks = 'P051'.
      int_fcdetails-divis = 'BM'.
    ELSEIF int_fcdetails-werks = 'P055'.
      int_fcdetails-divis = 'GT'.
    ELSEIF int_fcdetails-werks = 'P070'.
      int_fcdetails-divis = 'PUMPS'.
    ELSEIF int_fcdetails-werks = 'P090'.
      int_fcdetails-divis = 'PED'.
    ELSEIF int_fcdetails-werks+0(4) = 'P099'.
      CONCATENATE 'CMM' int_fcdetails-werks+4(5) INTO int_fcdetails-divis.
    ENDIF.
    MODIFY int_fcdetails.
  ENDLOOP.

  IF i_werks IS NOT INITIAL.
    DELETE int_fcdetails WHERE werks NE i_werks.
  ENDIF.

  IF int_fcdetails[] IS NOT INITIAL.
    SELECT vbeln posnn vbelv posnv
      INTO TABLE int_vbfa
    FROM vbfa
      FOR ALL ENTRIES IN int_fcdetails
    WHERE vbelv = int_fcdetails-sono
      AND posnv = int_fcdetails-item
      AND vbtyp_n = 'J'.
  ENDIF.
  IF int_vbfa[] IS NOT INITIAL.
```

```
    SELECT vbeln bldat
      INTO TABLE int_likp
    FROM likp
      FOR ALL ENTRIES IN int_vbfa
    WHERE vbeln = int_vbfa-vbeln.

    SELECT tknum tpnum vbeln
      INTO TABLE int_vttp
    FROM vttp
      FOR ALL ENTRIES IN int_vbfa
    WHERE vbeln = int_vbfa-vbeln.
  ENDIF.
  IF int_vttp[] IS NOT INITIAL.
    SELECT tknum signi exti1 tdlnr sdabw datbg
      INTO TABLE int_vttk
    FROM vttk
      FOR ALL ENTRIES IN int_vttp
    WHERE tknum = int_vttp-tknum.
  ENDIF.
  CLEAR: int_details, int_details[].
  IF int_fcdetails[] IS NOT INITIAL.
    CLEAR: int_makt, int_makt[].
    SELECT matnr maktx
      INTO TABLE int_makt
    FROM makt
      FOR ALL ENTRIES IN int_fcdetails
    WHERE matnr = int_fcdetails-zdesno.
  ENDIF.

  LOOP AT int_fcdetails.
    IF int_fcdetails-matnr NE int_fcdetails-zdesno.
      int_fcdetails-menge = 1.
      int_fcdetails-meins = 'EA'.
      CLEAR int_makt.
      READ TABLE int_makt WITH KEY matnr = int_fcdetails-zdesno.
      int_fcdetails-maktx = int_makt-maktx.
      int_fcdetails-zznetwt = 0.
      int_fcdetails-partno = ''.
      LOOP AT int_fcdetails INTO wa_fcdetails WHERE zdesno = int_fcdetails-zdes
no..
        int_fcdetails-zznetwt = int_fcdetails-zznetwt + wa_fcdetails-zznetwt.
        DELETE int_fcdetails.
      ENDLOOP.

    ENDIF.
    CLEAR int_vbfa.
    READ TABLE int_vbfa WITH KEY vbelv = int_fcdetails-sono posnv = int_fcdetai
ls-item.

    CLEAR int_likp.
    READ TABLE int_likp WITH KEY vbeln = int_vbfa-vbeln.
    int_fcdetails-deliv = int_likp-vbeln.
    int_fcdetails-deldt = int_likp-deldt.
    CLEAR int_vttp.
    READ TABLE int_vttp WITH KEY vbeln = int_vbfa-vbeln.

    CLEAR int_vttk.
    READ TABLE int_vttk WITH KEY tknum = int_vttp-tknum.
    int_fcdetails-tknum = int_vttk-tknum.
    int_fcdetails-datbg = int_vttk-datbg.
    int_fcdetails-exti1 = int_vttk-exti1.

    CLEAR int_zmmt001.
    READ TABLE int_zmmt001 WITH KEY pspel = int_fcdetails-posnr matnr = int_fcd
etails-zdesno sortf = int_fcdetails-partno.

    MOVE-CORRESPONDING int_fcdetails TO int_details.
    APPEND int_details.
    MODIFY int_fcdetails.
  ENDLOOP.

ENDFORM.                    " DISPLAY_FC_DETAILS
*&---------------------------------------------------------------------*
*&      Form  user_command_mat
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
*      -->UCOMM      text
*      -->SEL_FIELD  text
*----------------------------------------------------------------------*
FORM user_command_mat USING ucomm TYPE sy-ucomm
        sel_field TYPE slis_selfield.
  break 6146252.
  CASE ucomm.

    WHEN '&IC1'.

      IF sel_field-fieldname EQ 'ZDESNO'.
        SET PARAMETER ID 'MAT' FIELD sel_field-value.
        CALL TRANSACTION 'MMBE' AND SKIP FIRST SCREEN.
      ENDIF.

    WHEN '&DATA_SAVE'.
      break 6058515.

* to reflect the data changed into internal table
      DATA : ref_grid TYPE REF TO cl_gui_alv_grid. "new
      IF ref_grid IS INITIAL.
        CALL FUNCTION 'GET_GLOBALS_FROM_SLVC_FULLSCR'
          IMPORTING
            e_grid = ref_grid.
      ENDIF.

      IF NOT ref_grid IS INITIAL.
        CALL METHOD ref_grid->check_changed_data.
      ENDIF.


      LOOP AT int_details WHERE plan_desp_date NE '00000000'.

        IF int_details-zdesno+0(1) = 'Z'.
          UPDATE zecmt006 SET plan_desp_date = int_details-plan_desp_date
                WHERE projn = int_details-projn AND
                pspid = int_details-pspid AND
                pspnr = int_details-pspnr AND
                posnr = int_details-posnr AND
                werks = int_details-werks AND
**        matnr = int_details-matnr AND
                zdesno = int_details-zdesno AND
**          partno = int_details-partno AND
                des_serial = int_details-des_serial.
        ELSE.
          UPDATE zecmt006 SET plan_desp_date = int_details-plan_desp_date
          WHERE projn = int_details-projn AND
          pspid = int_details-pspid AND
          pspnr = int_details-pspnr AND
          posnr = int_details-posnr AND
          werks = int_details-werks AND
**        matnr = int_details-matnr AND
          zdesno = int_details-zdesno AND
          partno = int_details-partno AND
          des_serial = int_details-des_serial.
        ENDIF.
      ENDLOOP.
  ENDCASE.
ENDFORM.                    "user_command_mat
*&---------------------------------------------------------------------*
*&      Form  ADD_RECORD
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
*      -->P_WA_WERKS  text
*----------------------------------------------------------------------*
FORM add_record  USING    wa_werks.

  int_zsdt077-psphi = int_prps-psphi.
  int_zsdt077-werks = wa_werks.
  int_zsdt077-agency = 'MPC'.
  int_zsdt077-agency_pend = int_count-mpcpen.
  int_zsdt077-agency_tot = int_count-mpctot.
  int_zsdt077-erdat = sy-datum.
  APPEND int_zsdt077.
  CLEAR:int_zsdt077-agency_tot,int_zsdt077-agency_pend.
  int_zsdt077-agency = 'PPC'.
  int_zsdt077-agency_pend = int_count-ppcpen.
  int_zsdt077-agency_tot = int_count-ppctot.
  int_zsdt077-erdat = sy-datum.
  APPEND int_zsdt077.
  CLEAR:int_zsdt077-agency_tot,int_zsdt077-agency_pend.
  int_zsdt077-agency = 'SUBC'.
  int_zsdt077-agency_pend =  int_count-subpen.
  int_zsdt077-agency_tot = int_count-subtot.
  int_zsdt077-erdat = sy-datum.
  APPEND int_zsdt077.
  CLEAR:int_zsdt077-agency_tot,int_zsdt077-agency_pend.
  int_zsdt077-agency = 'PUR'.
  int_zsdt077-agency_pend =  int_count-purpen.
  int_zsdt077-agency_tot =  int_count-purtot.
  int_zsdt077-erdat = sy-datum.
  APPEND int_zsdt077.
  CLEAR:int_zsdt077-agency_tot,int_zsdt077-agency_pend.
  int_zsdt077-agency = 'STORE'.
  int_zsdt077-agency_pend =  int_count-strpen + int_count-gpissu.
  int_zsdt077-erdat = sy-datum.
  APPEND int_zsdt077.
  CLEAR:int_zsdt077-agency_tot,int_zsdt077-agency_pend.
*           int_zsdt077-agency_tot = int_count-mpctot.
  int_zsdt077-agency = 'CDC'.
  int_zsdt077-agency_pend =  int_count-cdcack + int_count-cmlpen + int_count-cd
cpak + int_count-cdcpq.
  int_zsdt077-erdat = sy-datum.
  APPEND int_zsdt077.
  CLEAR:int_zsdt077-agency_tot,int_zsdt077-agency_pend.
*           int_zsdt077-agency_tot = int_count-mpctot.
  int_zsdt077-agency = 'DES'.
  int_zsdt077-agency_tot =  int_count-totdesp.
  int_zsdt077-erdat = sy-datum.
  APPEND int_zsdt077.
  CLEAR:int_zsdt077.
*           int_zsdt077-agency_tot = int_count-mpctot.
ENDFORM.
*&---------------------------------------------------------------------*
*&      Form  INCR_COUNT
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
*  -->  p1        text
*  <--  p2        text
*----------------------------------------------------------------------*
FORM incr_count .
  READ TABLE int_zsdt077 WITH KEY psphi = int_prps-psphi werks = int_count-werk
s agency = 'MPC'.
  IF sy-subrc EQ 0.
    int_zsdt077-agency_pend = int_zsdt077-agency_pend + int_count-mpcpen.
    int_zsdt077-agency_tot = int_zsdt077-agency_tot + int_count-mpctot.
    MODIFY int_zsdt077 INDEX sy-tabix.
    CLEAR:int_zsdt077.
  ENDIF.

  READ TABLE int_zsdt077 WITH KEY psphi = int_prps-psphi werks = int_count-werk
s agency = 'PPC'.
```

```
  IF sy-subrc EQ 0.
    int_zsdt077-agency_pend = int_zsdt077-agency_pend + int_count-ppcpen.
    int_zsdt077-agency_tot = int_zsdt077-agency_tot + int_count-ppctot.
    MODIFY int_zsdt077 INDEX sy-tabix.
    CLEAR:int_zsdt077.
  ENDIF.

  READ TABLE int_zsdt077 WITH KEY psphi = int_prps-psphi werks = int_count-werk
s agency = 'SUBC'.
  IF sy-subrc EQ 0.
    int_zsdt077-agency_pend = int_zsdt077-agency_pend + int_count-subpen.
    int_zsdt077-agency_tot = int_zsdt077-agency_tot + int_count-subtot.
    MODIFY int_zsdt077 INDEX sy-tabix.
    CLEAR:int_zsdt077.
  ENDIF.

  READ TABLE int_zsdt077 WITH KEY psphi = int_prps-psphi werks = int_count-werk
s agency = 'PUR'.
  IF sy-subrc EQ 0.
    int_zsdt077-agency_pend = int_zsdt077-agency_pend + int_count-purpen.
    int_zsdt077-agency_tot = int_zsdt077-agency_tot + int_count-purtot.
    MODIFY int_zsdt077 INDEX sy-tabix.
    CLEAR:int_zsdt077.
  ENDIF.

  READ TABLE int_zsdt077 WITH KEY psphi = int_prps-psphi werks = int_count-werk
s agency = 'STORE'.
  IF sy-subrc EQ 0.
    int_zsdt077-agency_pend = int_zsdt077-agency_pend + int_count-strpen + int_
count-gpissu.
    MODIFY int_zsdt077 INDEX sy-tabix.
    CLEAR:int_zsdt077.
  ENDIF.

  READ TABLE int_zsdt077 WITH KEY psphi = int_prps-psphi werks = int_count-werk
s agency = 'CDC'.
  IF sy-subrc EQ 0.
    int_zsdt077-agency_pend = int_zsdt077-agency_pend + int_count-cdcack + int_
count-cmlpen + int_count-cdcpak + int_count-cdcpq.
    MODIFY int_zsdt077 INDEX sy-tabix.
    CLEAR:int_zsdt077.
  ENDIF.

  READ TABLE int_zsdt077 WITH KEY psphi = int_prps-psphi werks = int_count-werk
s agency = 'DES'.
  IF sy-subrc EQ 0.
    int_zsdt077-agency_tot = int_zsdt077-agency_tot + int_count-totdesp.
    MODIFY int_zsdt077 INDEX sy-tabix.
    CLEAR:int_zsdt077.
  ENDIF.
ENDFORM.
*&---------------------------------------------------------------------*
*&      Form  pfstatus
```

```
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
*      -->UT_EXTAB   text
*----------------------------------------------------------------------*
FORM pfstatus USING ut_extab TYPE slis_t_extab.
  SET PF-STATUS 'STANDARD_FULLSCREEN' OF PROGRAM 'SAPLKKBL'.
ENDFORM.                    "pfstatus
```