Step by Step Procedure for Reconciliation of Sales Data in SAP BW Environment with ECC Environment

Applies to:

SAP ECC 6.00 and SAP BW 7.0 releases. For more information, visit the Business Intelligence homepage.

Summary

This paper gives a detail understanding of the steps to follow custom ABAP Programs creation for Sales reconciliation. This document provides a step by step guide to create custom ABAP programs on ECC environment and also discusses how these programs can help in reconciling sales data on SAP BW environment.

Author: Vamsi Sirish Siddabattuni

Company: Deloitte Consulting LLP

Created on: 06 November 2009

Author Bio

Vamsi Sirish Siddabattuni is an SAP Certified Netweaver 2004s BI Solution Consultant and SAP Certified ABAP Workbench consultant. He has more than 6 yrs of experience working on SAP BW projects and has implemented SAP BW systems based on SAP BW 3.0b, 3.5 and 7.0 versions.

Table of Contents

Introduction

Often SAP BI Projects implementing Sales and Distribution module face the challenge of reconciling the data loaded into SAP BI from ECC as there are no standard reports in ECC which can help in easy reconciliation of Sales, Delivery and Billing data loaded.

This document is focused on creating custom ABAP Reports on ECC environment to validate data loaded onto SAP BW and perform reconciliation process easily.

Scenario

The scenario discusses about loading the Line Item Level data for Sales Header and Item, Delivery Header and Item, Billing Header and Item and Billing Conditions into SAP BI environment to DSOs and validating the data loaded into those DSOs with custom ABAP Reports built on ECC to see if the data loads are getting the data successfully.

It primarily concentrates on creation of ABAP programs within each sub-module of Sales and Distribution on ECC and steps on how generic BEx queries built on SAP BI environment on top of the Line Item Level DSOs for Sales Line Item , Delivery Line Item, Billing Line Item and Billing Conditions DSOs are used to reconcile the data. The scenario doesn't discuss about the creation of Line Item Level DSOs and the design of those DSOs and BEx queries built on top of the DSOs.

Step By Step Procedure

Step by Step Procedure discusses in four sections about the Sales Order Item, Delivery Item, Billing Item and Billing Conditions reconciliation of data on SAP BI with ECC environment.

Sales Order Reconciliation :

1. Create an ABAP Program on ECC environment (Ex : ZBI_SALES_ANALYSIS). Refer to appendix for Program Code. This program retrieves the data from Sales Order Header and Sales Order Item Tables and shows the data in an ALV output with the selections entered in the selection screen. Selection screen for the ABAP Program (ZBI_SALES_ANALYSIS) is in the screenshot below. Sample output of the program execution is show below.
2. You can also schedule a background job for this ABAP Program to execute at the same time when the nightly job run for BI Extraction happens. This way the spool output of the ABAP Program can be downloaded onto a flat file and can be compared with the BI Report we are going to discuss in the next point.

Execute the ABAP Program 'ZBI_SALES_ANALYSIS' for example in SE38. This prompts the following screen.

Enter the selections needed and schedule the background job which runs on a daily basis.

Enter the spool parameters in the below screen

Enter the frequency

Background job will be scheduled after entering the selections and pressing the check button.

Spool output of the request can be printed from SM37 on a daily basis to keep the documents for records.

3. On the SAP BW environment connected to the ECC environment create a BEx Query on top of DSO which stores Sales Order Header and Item Information on SAP BI Environment. Sample query definition is shown in the below screenshot.

Execute the query with the same selections you defined for ABAP Program on ECC to check the data for reconciliation. Sample output of the BEx query is shown below. Output of the BEx query can be used for reconciliation with the ABAP Spool output on a daily basis to check and see if the data loaded in to BW is correct or not.

Delivery Item Reconciliation :

1. Create an ABAP Program on ECC environment (Ex : ZBI_DELIVERY_ANALYSIS). Refer to appendix for Program Code. This program retrieves the data from Delivery Header and Delivery Item Tables and shows the data in an ALV output with the selections entered in the selection screen. Selection screen for the ABAP Program (ZBI_DELIVERY_ANALYSIS) is in the screenshot below. Sample output of the program execution is show below.
2. You can also schedule a background job for this ABAP Program by following the steps mentioned in Step by step procedure for Sales Analysis Reconciliation for background job scheduling.
3. On the SAP BW environment connected to the ECC environment create a BEx Query on top of DSO which stores Delivery Header and Item Information on SAP BI Environment. Sample query definition is shown in the below screenshot.

Execute the query with the same selections you defined for ABAP Program on ECC to check the data for reconciliation. Sample output of the BEx query is shown below. Output of the BEx query can be used for reconciliation with the ABAP Spool output on a daily basis to check and see if the data loaded in to BW is correct or not.

Billing Item Reconciliation :

1. Create an ABAP Program on ECC environment (Ex : ZBI_BILLING_ANALYSIS). Refer to appendix for Program Code. This program retrieves the data from Billing Header and Billing Item Tables and shows the data in an ALV output with the selections entered in the selection screen.
   Selection screen for the ABAP Program (ZBI_BILLING_ANALYSIS) is in the screenshot below.
   Sample output of the program execution is show below.
2. You can also schedule a background job for this ABAP Program by following the steps mentioned in Step by step procedure for Sales Analysis Reconciliation for background job scheduling.
3. On the SAP BW environment connected to the ECC environment create a BEx Query on top of DSO which stores Billing Header and Item Information on SAP BI Environment. Sample query definition is shown in the below screenshot.

Execute the query with the same selections you defined for ABAP Program on ECC to check the data for reconciliation. Sample output of the BEx query is shown below. Output of the BEx query can be used for reconciliation with the ABAP Spool output on a daily basis to check and see if the data loaded in to BW is correct or not.

Billing Conditions Reconciliation :

1. Create an ABAP Program on ECC environment (Ex : ZBI_BILLING_COND_ANALYSIS). Refer to appendix for Program Code. This program retrieves the data from underlying Billing Conditions Tables and shows the data in an ALV output with the selections entered in the selection screen. Selection screen for the ABAP Program (ZBI_BILLING_COND_ANALYSIS) is in the screenshot below.
   Sample output of the program execution is show below.
2. You can also schedule a background job for this ABAP Program by following the steps mentioned in Step by step procedure for Sales Analysis Reconciliation for background job scheduling.
3. On the SAP BW environment connected to the ECC environment create a BEx Query on top of DSO which stores Delivery Header and Item Information on SAP BI Environment. Sample query definition is shown in the below screenshot.

Execute the query with the same selections you defined for ABAP Program on ECC to check the data for reconciliation. Sample output of the BEx query is shown below. Output of the BEx query can be used for reconciliation with the ABAP Spool output on a daily basis to check and see if the data loaded in to BW is correct or not.

The Code

Sales Analysis Program Code

```
*&---------------------------------------------------------------------*
*& Report  ZBI_SALES_ANALYSIS
*&
*&---------------------------------------------------------------------*
*&
*&
*&---------------------------------------------------------------------*
REPORT  ZBI_SALES_ANALYSIS.
TABLES : VBAK,VBAP,VBEP.
START-OF-SELECTION.
  SELECT-OPTIONS SLS_DOC FOR VBAK-VBELN.
  SELECT-OPTIONS CRTD_ON FOR SY-DATUM.
  SELECT-OPTIONS REQD_DT FOR SY-DATUM.
  SELECT-OPTIONS DOC_TYPE FOR VBAK-AUART.
  SELECT-OPTIONS SAL_ORG FOR VBAK-VKORG.
  SELECT-OPTIONS DIS_CHAN FOR VBAK-VTWEG.
  SELECT-OPTIONS DIVISION FOR VBAK-SPART.
  SELECT-OPTIONS MATERIAL FOR VBAP-MATNR.
  SELECT-OPTIONS SOLD_TO FOR VBAK-KUNNR.
  SELECT-OPTIONS PLANT FOR VBAP-WERKS.
END-OF-SELECTION.
  TYPES : BEGIN OF STR_SALES,
          VBELN TYPE VBAK-VBELN,
          POSNR TYPE VBAP-POSNR,
          ERDAT TYPE VBAK-ERDAT,
          VBTYP TYPE VBAK-VBTYP,
          AUART TYPE VBAK-AUART,
          AUGRU TYPE VBAK-AUGRU,
          VKORG TYPE VBAK-VKORG,
          VTWEG TYPE VBAK-VTWEG,
          SPART TYPE VBAK-SPART,
          VDATU TYPE VBAK-VDATU,
          VSBED TYPE VBAK-VSBED,
          BSTNK TYPE VBAK-BSTNK,
          KUNNR TYPE VBAK-KUNNR,
          MATNR TYPE VBAP-MATNR,
          ABGRU TYPE VBAP-ABGRU,
          ZMENG TYPE VBAP-ZMENG,
          ZIEME TYPE VBAP-ZIEME,
          NETWR TYPE VBAP-NETWR,
          KWMENG TYPE VBAP-KWMENG,
          VRKME TYPE VBAP-VRKME,
          VGBEL TYPE VBAP-VGBEL,
          VGPOS TYPE VBAP-VGPOS,
          NETPR TYPE VBAP-NETPR,
          SHKZG TYPE VBAP-SHKZG,
          WAVWR TYPE VBAP-WAVWR,
          KZWI1 TYPE VBAP-KZWI1,
          KZWI2 TYPE VBAP-KZWI2,
          KZWI3 TYPE VBAP-KZWI3,
          KZWI4 TYPE VBAP-KZWI4,
          KZWI5 TYPE VBAP-KZWI5,
```

```
        KZWI6 TYPE VBAP-KZWI6,
        WERKS TYPE VBAP-WERKS,
        WADAT TYPE VBEP-WADAT,
        END OF STR_SALES.
TYPES : BEGIN OF STR_VBAK,
        VBELN TYPE VBAK-VBELN,
        ERDAT TYPE VBAK-ERDAT,
        VBTYP TYPE VBAK-VBTYP,
        AUART TYPE VBAK-AUART,
        AUGRU TYPE VBAK-AUGRU,
        VKORG TYPE VBAK-VKORG,
        VTWEG TYPE VBAK-VTWEG,
        SPART TYPE VBAK-SPART,
        VDATU TYPE VBAK-VDATU,
        VSBED TYPE VBAK-VSBED,
        BSTNK TYPE VBAK-BSTNK,
        KUNNR TYPE VBAK-KUNNR,
        END OF STR_VBAK.
TYPES : BEGIN OF STR_VBAP,
        VBELN TYPE VBAP-VBELN,
        POSNR TYPE VBAP-POSNR,
        MATNR TYPE VBAP-MATNR,
        ABGRU TYPE VBAP-ABGRU,
        ZMENG TYPE VBAP-ZMENG,
        ZIEME TYPE VBAP-ZIEME,
        NETWR TYPE VBAP-NETWR,
        KWMENG TYPE VBAP-KWMENG,
        VRKME TYPE VBAP-VRKME,
        VGBEL TYPE VBAP-VGBEL,
        VGPOS TYPE VBAP-VGPOS,
        NETPR TYPE VBAP-NETPR,
        SHKZG TYPE VBAP-SHKZG,
        WAVWR TYPE VBAP-WAVWR,
        KZWI1 TYPE VBAP-KZWI1,
        KZWI2 TYPE VBAP-KZWI2,
        KZWI3 TYPE VBAP-KZWI3,
        KZWI4 TYPE VBAP-KZWI4,
        KZWI5 TYPE VBAP-KZWI5,
        KZWI6 TYPE VBAP-KZWI6,
        WERKS TYPE VBAP-WERKS,
        END OF STR_VBAP.
TYPES : BEGIN OF STR_VBEP,
        VBELN TYPE VBAP-VBELN,
        POSNR TYPE VBAP-POSNR,
        ETENR TYPE VBEP-ETENR,
        WADAT TYPE VBEP-WADAT,
        END OF STR_VBEP.
DATA : IT_VBAK TYPE TABLE OF STR_VBAK,
       IT_VBAP TYPE TABLE OF STR_VBAP,
       IT_VBEP TYPE TABLE OF STR_VBEP,
       WA_VBAK TYPE STR_VBAK,
       WA_VBAP TYPE STR_VBAP,
       WA_VBEP TYPE STR_VBEP,
       IT_OUTPUT TYPE TABLE OF STR_SALES,
       WA_OUTPUT TYPE STR_SALES.
type-pools : slis .
```

```
*  data: fieldcatalog type slis_t_fieldcat_alv with header line,
*        gd_tab_group type slis_t_sp_group_alv,
*        gd_layout    type slis_layout_alv,
*        gd_repid     like sy-repid,
*        gt_events     type slis_t_event.
  DATA:   gr_table      TYPE REF TO cl_salv_table,
          gr_functions  TYPE REF TO cl_salv_functions,      "#EC *
          gr_columns    TYPE REF TO cl_salv_columns,        "#EC *
          gr_column     TYPE REF TO cl_salv_column,         "#EC *
          gr_sorts      TYPE REF TO cl_salv_sorts,
          gr_agg        TYPE REF TO cl_salv_aggregations.
  SELECT * FROM VBAK INTO CORRESPONDING FIELDS OF TABLE
  IT_VBAK WHERE VBELN IN SLS_DOC AND
  ERDAT IN CRTD_ON AND VDATU IN REQD_DT AND
  AUART IN DOC_TYPE AND VKORG IN SAL_ORG AND
  VTWEG IN DIS_CHAN AND SPART IN DIVISION AND
  KUNNR IN SOLD_TO.
  SELECT * FROM VBAP INTO CORRESPONDING FIELDS OF TABLE IT_VBAP
  FOR ALL ENTRIES IN IT_VBAK WHERE
  VBELN = IT_VBAK-VBELN AND MATNR IN MATERIAL AND WERKS IN PLANT.
  SELECT * FROM VBEP INTO CORRESPONDING FIELDS OF TABLE IT_VBEP
  FOR ALL ENTRIES IN IT_VBAP WHERE
  VBELN = IT_VBAP-VBELN AND POSNR = IT_VBAP-POSNR.
  SORT IT_VBEP BY VBELN POSNR ETENR DESCENDING.
  LOOP AT IT_VBAP INTO WA_VBAP.
    WA_OUTPUT-VBELN = WA_VBAP-VBELN.
    WA_OUTPUT-POSNR = WA_VBAP-POSNR.
    WA_OUTPUT-MATNR = WA_VBAP-MATNR.
    WA_OUTPUT-ABGRU = WA_VBAP-ABGRU.
    WA_OUTPUT-ZMENG = WA_VBAP-ZMENG.
    WA_OUTPUT-ZIEME = WA_VBAP-ZIEME.
    WA_OUTPUT-NETWR = WA_VBAP-NETWR.
    WA_OUTPUT-KWMENG = WA_VBAP-KWMENG.
    WA_OUTPUT-VRKME = WA_VBAP-VRKME.
    WA_OUTPUT-VGBEL = WA_VBAP-VGBEL.
    WA_OUTPUT-VGPOS = WA_VBAP-VGPOS.
    WA_OUTPUT-NETPR = WA_VBAP-NETPR.
    WA_OUTPUT-SHKZG = WA_VBAP-SHKZG.
    WA_OUTPUT-WAVWR = WA_VBAP-WAVWR.
    WA_OUTPUT-KZWI1 = WA_VBAP-KZWI1.
    WA_OUTPUT-KZWI2 = WA_VBAP-KZWI2.
    WA_OUTPUT-KZWI3 = WA_VBAP-KZWI3.
    WA_OUTPUT-KZWI4 = WA_VBAP-KZWI4.
    WA_OUTPUT-KZWI5 = WA_VBAP-KZWI5.
    WA_OUTPUT-KZWI6 = WA_VBAP-KZWI6.
    WA_OUTPUT-WERKS = WA_VBAP-WERKS.
    APPEND WA_OUTPUT TO IT_OUTPUT.
    CLEAR : WA_OUTPUT, WA_VBAP.
  ENDLOOP.
  LOOP AT IT_OUTPUT INTO WA_OUTPUT.
    READ TABLE IT_VBAK INTO WA_VBAK
    WITH KEY VBELN = WA_OUTPUT-VBELN.
    IF SY-SUBRC EQ 0.
      WA_OUTPUT-ERDAT = WA_VBAK-ERDAT.
      WA_OUTPUT-VBTYP = WA_VBAK-VBTYP.
      WA_OUTPUT-AUART = WA_VBAK-AUART.
```

```
      WA_OUTPUT-AUGRU = WA_VBAK-AUGRU.
      WA_OUTPUT-VKORG = WA_VBAK-VKORG.
      WA_OUTPUT-VTWEG = WA_VBAK-VTWEG.
      WA_OUTPUT-SPART = WA_VBAK-SPART.
      WA_OUTPUT-VDATU = WA_VBAK-VDATU.
      WA_OUTPUT-VSBED = WA_VBAK-VSBED.
      WA_OUTPUT-BSTNK = WA_VBAK-BSTNK.
      WA_OUTPUT-KUNNR = WA_VBAK-KUNNR.
    ENDIF.
    READ TABLE IT_VBEP INTO WA_VBEP
    WITH KEY VBELN = WA_OUTPUT-VBELN POSNR = WA_OUTPUT-POSNR.
    IF SY-SUBRC EQ 0.
      WA_OUTPUT-WADAT = WA_VBEP-WADAT.
    ENDIF.
    MODIFY IT_OUTPUT FROM WA_OUTPUT.
    CLEAR : WA_OUTPUT, WA_VBAK, WA_VBEP.
  ENDLOOP.
  DATA: ls_info     TYPE slis_listheader,
          lv_stext    TYPE scrtext_s,                                  " Short Text
          lv_mtext    TYPE scrtext_m,                                  " Medium
Text
          lv_ltext    TYPE scrtext_l,                                  " Long Text
          lv_colname  TYPE lvc_fname,                                  " Column
Name
          lv_position TYPE i VALUE 1,                                  " Position
          ls_line TYPE slis_listheader,                                "
          lv_char(15) TYPE c.
* Class declarations
  DATA: ls_rows        TYPE REF TO cl_salv_form_layout_grid,
        ls_row         TYPE REF TO cl_salv_form_layout_flow.
* Create Object Reference
  CREATE OBJECT ls_rows.
  TRY.
      cl_salv_table=>factory( IMPORTING r_salv_table = gr_table CHANGING t_table =
it_output ). "#EC
    CATCH cx_salv_msg.                                          "#EC
    CATCH cx_salv_not_found.                                    "#EC
  ENDTRY.
  gr_columns = gr_table->get_columns( ).
  gr_columns->set_optimize( abap_true ).
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'VBELN'.
  lv_position = 1.
  lv_stext = 'Sales Order'(s28).
  lv_mtext = 'Sales Order'(s28).
  lv_ltext = 'Sales Order'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'POSNR'.
  lv_position = lv_position + 1.
  lv_stext = 'Sales Item'(s28).
  lv_mtext = 'Sales Item'(s28).
  lv_ltext = 'Sales Item'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'ERDAT'.
```

```
  lv_position = lv_position + 1.
  lv_stext = 'Create Date'(s28).
  lv_mtext = 'Create Date'(s28).
  lv_ltext = 'Create Date'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'VBTYP'.
  lv_position = lv_position + 1.
  lv_stext = 'Doc Categ'(s28).
  lv_mtext = 'Doc Categ'(s28).
  lv_ltext = 'Doc Categ'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'AUART'.
  lv_position = lv_position + 1.
  lv_stext = 'Doc Type'(s28).
  lv_mtext = 'Doc Type'(s28).
  lv_ltext = 'Doc Type'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'AUGRU'.
  lv_position = lv_position + 1.
  lv_stext = 'Ord Reason'(s28).
  lv_mtext = 'Ord Reason'(s28).
  lv_ltext = 'Ord Reason'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'VKORG'.
  lv_position = lv_position + 1.
  lv_stext = 'Sales Org'(s28).
  lv_mtext = 'Sales Org'(s28).
  lv_ltext = 'Sales Org'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'VTWEG'.
  lv_position = lv_position + 1.
  lv_stext = 'Dist Chan'(s28).
  lv_mtext = 'Dist Chan'(s28).
  lv_ltext = 'Dist Chan'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'SPART'.
  lv_position = lv_position + 1.
  lv_stext = 'Division'(s28).
  lv_mtext = 'Division'(s28).
  lv_ltext = 'Division'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'VDATU'.
  lv_position = lv_position + 1.
  lv_stext = 'Req Del Date'(s28).
  lv_mtext = 'Req Del Date'(s28).
  lv_ltext = 'Req Del Date'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'VSBED'.
```

```
  lv_position = lv_position + 1.
  lv_stext = 'Ship Cond'(s28).
  lv_mtext = 'Ship Cond'(s28).
  lv_ltext = 'Ship Cond'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'BSTNK'.
  lv_position = lv_position + 1.
  lv_stext = 'Cust Pur No'(s28).
  lv_mtext = 'Cust Pur No'(s28).
  lv_ltext = 'Cust Pur No'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'KUNNR'.
  lv_position = lv_position + 1.
  lv_stext = 'Sold To'(s28).
  lv_mtext = 'Sold To'(s28).
  lv_ltext = 'Sold To'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'MATNR'.
  lv_position = lv_position + 1.
  lv_stext = 'Material'(s28).
  lv_mtext = 'Material'(s28).
  lv_ltext = 'Material'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'ABGRU'.
  lv_position = lv_position + 1.
  lv_stext = 'Reason for Rej'(s28).
  lv_mtext = 'Reason for Rej'(s28).
  lv_ltext = 'Reason for Rej'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'ZMENG'.
  lv_position = lv_position + 1.
  lv_stext = 'Target Qty'(s28).
  lv_mtext = 'Target Qty'(s28).
  lv_ltext = 'Target Qty'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'ZIEME'.
  lv_position = lv_position + 1.
  lv_stext = 'Target Qty UOM'(s28).
  lv_mtext = 'Target Qty UOM'(s28).
  lv_ltext = 'Target Qty UOM'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'NETWR'.
  lv_position = lv_position + 1.
  lv_stext = 'Net Value'(s28).
  lv_mtext = 'Net Value'(s28).
  lv_ltext = 'Net Value'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'KWMENG'.
```

```
  lv_position = lv_position + 1.
  lv_stext = 'Sal Qty'(s28).
  lv_mtext = 'Sal Qty'(s28).
  lv_ltext = 'Sal Qty'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'VRKME'.
  lv_position = lv_position + 1.
  lv_stext = 'Sal Qty UOM'(s28).
  lv_mtext = 'Sal Qty UOM'(s28).
  lv_ltext = 'Sal Qty UOM'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'VGBEL'.
  lv_position = lv_position + 1.
  lv_stext = 'Refer Doc'(s28).
  lv_mtext = 'Refer Doc'(s28).
  lv_ltext = 'Refer Doc'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'VGPOS'.
  lv_position = lv_position + 1.
  lv_stext = 'Refer Item'(s28).
  lv_mtext = 'Refer Item'(s28).
  lv_ltext = 'Refer Item'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'NETPR'.
  lv_position = lv_position + 1.
  lv_stext = 'Net Price'(s28).
  lv_mtext = 'Net Price'(s28).
  lv_ltext = 'Net Price'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'SHKZG'.
  lv_position = lv_position + 1.
  lv_stext = 'Returns'(s28).
  lv_mtext = 'Returns'(s28).
  lv_ltext = 'Returns'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'WAVWR'.
  lv_position = lv_position + 1.
  lv_stext = 'Cost'(s28).
  lv_mtext = 'Cost'(s28).
  lv_ltext = 'Cost'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'WADAT'.
  lv_position = lv_position + 1.
  lv_stext = 'Goods Issue Dt'(s28).
  lv_mtext = 'Goods Issue Dt'(s28).
  lv_ltext = 'Goods Issue Dt'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'WERKS'.
```

```
  lv_position = lv_position + 1.
  lv_stext = 'Plant'(s28).
  lv_mtext = 'Plant'(s28).
  lv_ltext = 'Plant'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'KZWI1'.
  lv_position = lv_position + 1.
  lv_stext = 'Sub Total 1'(s28).
  lv_mtext = 'Sub Total 1'(s28).
  lv_ltext = 'Sub Total 1'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'KZWI2'.
  lv_position = lv_position + 1.
  lv_stext = 'Sub Total 2'(s28).
  lv_mtext = 'Sub Total 2'(s28).
  lv_ltext = 'Sub Total 2'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'KZWI3'.
  lv_position = lv_position + 1.
  lv_stext = 'Sub Total 3'(s28).
  lv_mtext = 'Sub Total 3'(s28).
  lv_ltext = 'Sub Total 3'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'KZWI4'.
  lv_position = lv_position + 1.
  lv_stext = 'Sub Total 4'(s28).
  lv_mtext = 'Sub Total 4'(s28).
  lv_ltext = 'Sub Total 4'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'KZWI5'.
  lv_position = lv_position + 1.
  lv_stext = 'Sub Total 5'(s28).
  lv_mtext = 'Sub Total 5'(s28).
  lv_ltext = 'Sub Total 5'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'KZWI6'.
  lv_position = lv_position + 1.
  lv_stext = 'Sub Total 6'(s28).
  lv_mtext = 'Sub Total 6'(s28).
  lv_ltext = 'Sub Total 6'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Display the Header information.
  CALL METHOD gr_table->set_top_of_list
    EXPORTING
      value = ls_rows.
* Getting the All columns names
  gr_columns = gr_table->get_columns( ).
  gr_functions = gr_table->get_functions( ).
  gr_functions->set_all( abap_true ).
* To addd the sort functionality
```

```
  gr_sorts = gr_table->get_sorts( ).
*  gr_sorts->add_sort( columnname = 'VBELN'(s27) subtotal = abap_true ).
*  gr_sorts->add_sort( columnname = 'POSNR'(s30) subtotal = abap_true ).
  gr_agg = gr_table->get_aggregations( ).
  gr_agg->add_aggregation( 'ZMENG'(s66) ).
  gr_agg->add_aggregation( 'NETWR'(s63) ).
  gr_agg->add_aggregation( 'NETPR'(s46) ).
  gr_agg->add_aggregation( 'WAVWR'(s55) ).
  gr_agg->add_aggregation( 'KZWI1'(s57) ).
  gr_agg->add_aggregation( 'KZWI2'(s57) ).
  gr_agg->add_aggregation( 'KZWI3'(s57) ).
  gr_agg->add_aggregation( 'KZWI4'(s57) ).
  gr_agg->add_aggregation( 'KZWI5'(s57) ).
  gr_agg->add_aggregation( 'KZWI6'(s57) ).
*Display output table.
  gr_table->display( ).
*&---------------------------------------------------------------------*
*&      Form  F_SETCOLUMN_NAME
*&---------------------------------------------------------------------*
* Display the column according to the report output
*----------------------------------------------------------------------*
FORM f_setcolumn_name  USING    pu_colname  TYPE lvc_fname               " Field Name
                                pu_position TYPE i                      " Position
                                pu_lv_stext TYPE scrtext_s              " Short Text
                                pu_lv_mtext TYPE scrtext_m              " Medium
Text
                                pu_lv_ltext TYPE scrtext_l.             " Short Text
  TRY.
*     get the required column details to GR_COLUMN.
      gr_column = gr_columns->get_column( pu_colname ).
*     Short Text for column.
      gr_column->set_short_text( pu_lv_stext ).
*     Medium Text for column.
      gr_column->set_medium_text( pu_lv_mtext ).
*     Long Text for column.
      gr_column->set_long_text( pu_lv_ltext ).
*     Setting Column name and position.
      gr_columns->set_column_position(
                           columnname = pu_colname
                           position   = pu_position ).
    CATCH cx_salv_not_found.                                "#EC
  ENDTRY.
* clearing the variable used in the perform .
  CLEAR: pu_colname,pu_lv_stext,pu_lv_mtext,pu_lv_ltext.
ENDFORM.                    " F_SETCOLUMN_NAME
```

Delivery Analysis Program Code

```
*&---------------------------------------------------------------------*
*& Report  ZBI_DELIVERY_ANALYSIS
*&
*&---------------------------------------------------------------------*
*&
*&
*&---------------------------------------------------------------------*
REPORT  ZBI_DELIVERY_ANALYSIS.
TABLES : LIKP,LIPS.
START-OF-SELECTION.
  SELECT-OPTIONS DEL_DOC FOR LIKP-VBELN.
  SELECT-OPTIONS CRTD_ON FOR SY-DATUM.
  SELECT-OPTIONS PLNGDSDT FOR SY-DATUM.
  SELECT-OPTIONS ACTGDSDT FOR SY-DATUM.
  SELECT-OPTIONS DEL_TYPE FOR LIKP-LFART.
  SELECT-OPTIONS ITEM_CAT FOR LIPS-PSTYV.
  SELECT-OPTIONS SAL_ORG FOR LIKP-VKORG.
  SELECT-OPTIONS PLANT FOR LIPS-WERKS.
  SELECT-OPTIONS DIS_CHAN FOR LIPS-VTWEG.
  SELECT-OPTIONS DIVISION FOR LIPS-SPART.
  SELECT-OPTIONS MATERIAL FOR LIPS-MATNR.
  SELECT-OPTIONS SOLD_TO FOR LIKP-KUNNR.
  SELECT-OPTIONS SHIP_TO FOR LIKP-KUNAG.
END-OF-SELECTION.
  TYPES : BEGIN OF STR_DEL,
          VBELN TYPE LIKP-VBELN,
          POSNR TYPE LIPS-POSNR,
          ERDAT TYPE LIKP-ERDAT,
          VKORG TYPE LIKP-VKORG,
          LFART TYPE LIKP-LFART,
          WADAT TYPE LIKP-WADAT,
          LFDAT TYPE LIKP-LFDAT,
          VBTYP TYPE LIKP-VBTYP,
          VSBED TYPE LIKP-VSBED,
          KUNNR TYPE LIKP-KUNNR,
          KUNAG TYPE LIKP-KUNAG,
          WADAT_IST TYPE LIKP-WADAT_IST,
          PSTYV TYPE LIPS-PSTYV,
          MATNR TYPE LIPS-MATNR,
          WERKS TYPE LIPS-WERKS,
          LGORT TYPE LIPS-LGORT,
          KDMAT TYPE LIPS-KDMAT,
          LFIMG TYPE LIPS-LFIMG,
          VRKME TYPE LIPS-VRKME,
          NTGEW TYPE LIPS-NTGEW,
          BRGEW TYPE LIPS-BRGEW,
          GEWEI TYPE LIPS-GEWEI,
          LGMNG TYPE LIPS-LGMNG,
          VBTYV TYPE LIPS-VBTYV,
          VGBEL TYPE LIPS-VGBEL,
          VGPOS TYPE LIPS-VGPOS,
          POSAR TYPE LIPS-POSAR,
          VTWEG TYPE LIPS-VTWEG,
          SPART TYPE LIPS-SPART,
```

```
        SHKZG TYPE LIPS-SHKZG,
        NETPR TYPE LIPS-NETPR,
        NETWR TYPE LIPS-NETWR,
        WAVWR TYPE LIPS-WAVWR,
        KZWI1 TYPE LIPS-KZWI1,
        KZWI2 TYPE LIPS-KZWI2,
        KZWI3 TYPE LIPS-KZWI3,
        KZWI4 TYPE LIPS-KZWI4,
        KZWI5 TYPE LIPS-KZWI5,
        KZWI6 TYPE LIPS-KZWI6,
        END OF STR_DEL.
TYPES : BEGIN OF STR_LIKP,
        VBELN TYPE LIKP-VBELN,
        ERDAT TYPE LIKP-ERDAT,
        VKORG TYPE LIKP-VKORG,
        LFART TYPE LIKP-LFART,
        WADAT TYPE LIKP-WADAT,
        LFDAT TYPE LIKP-LFDAT,
        VBTYP TYPE LIKP-VBTYP,
        VSBED TYPE LIKP-VSBED,
        KUNNR TYPE LIKP-KUNNR,
        KUNAG TYPE LIKP-KUNAG,
        WADAT_IST TYPE LIKP-WADAT_IST,
        END OF STR_LIKP.
TYPES : BEGIN OF STR_LIPS,
        VBELN TYPE LIPS-VBELN,
        POSNR TYPE LIPS-POSNR,
        PSTYV TYPE LIPS-PSTYV,
        MATNR TYPE LIPS-MATNR,
        WERKS TYPE LIPS-WERKS,
        LGORT TYPE LIPS-LGORT,
        KDMAT TYPE LIPS-KDMAT,
        LFIMG TYPE LIPS-LFIMG,
        VRKME TYPE LIPS-VRKME,
        NTGEW TYPE LIPS-NTGEW,
        BRGEW TYPE LIPS-BRGEW,
        GEWEI TYPE LIPS-GEWEI,
        LGMNG TYPE LIPS-LGMNG,
        VBTYV TYPE LIPS-VBTYV,
        VGBEL TYPE LIPS-VGBEL,
        VGPOS TYPE LIPS-VGPOS,
        POSAR TYPE LIPS-POSAR,
        VTWEG TYPE LIPS-VTWEG,
        SPART TYPE LIPS-SPART,
        SHKZG TYPE LIPS-SHKZG,
        NETPR TYPE LIPS-NETPR,
        NETWR TYPE LIPS-NETWR,
        WAVWR TYPE LIPS-WAVWR,
        KZWI1 TYPE LIPS-KZWI1,
        KZWI2 TYPE LIPS-KZWI2,
        KZWI3 TYPE LIPS-KZWI3,
        KZWI4 TYPE LIPS-KZWI4,
        KZWI5 TYPE LIPS-KZWI5,
        KZWI6 TYPE LIPS-KZWI6,
        END OF STR_LIPS.
DATA : IT_LIKP TYPE TABLE OF STR_LIKP,
```

```
        IT_LIPS TYPE TABLE OF STR_LIPS,
        WA_LIKP TYPE STR_LIKP,
        WA_LIPS TYPE STR_LIPS,
        IT_OUTPUT TYPE TABLE OF STR_DEL,
        WA_OUTPUT TYPE STR_DEL.
  type-pools : slis .
*data: fieldcatalog type slis_t_fieldcat_alv with header line,
*        gd_tab_group type slis_t_sp_group_alv,
*        gd_layout    type slis_layout_alv,
*        gd_repid     like sy-repid.
  DATA:   gr_table      TYPE REF TO cl_salv_table,
          gr_functions  TYPE REF TO cl_salv_functions,      "#EC *
          gr_columns    TYPE REF TO cl_salv_columns,        "#EC *
          gr_column     TYPE REF TO cl_salv_column,         "#EC *
          gr_sorts      TYPE REF TO cl_salv_sorts,
          gr_agg        TYPE REF TO cl_salv_aggregations.
  SELECT * FROM LIKP INTO CORRESPONDING FIELDS OF TABLE
  IT_LIKP WHERE VBELN IN DEL_DOC AND
  ERDAT IN CRTD_ON AND WADAT IN PLNGDSDT
  AND WADAT_IST IN ACTGDSDT AND LFART IN DEL_TYPE AND
  VKORG IN SAL_ORG AND WERKS IN PLANT AND
  KUNNR IN SOLD_TO AND KUNAG IN SHIP_TO.
  SELECT * FROM LIPS INTO CORRESPONDING FIELDS OF TABLE IT_LIPS
  FOR ALL ENTRIES IN IT_LIKP WHERE
  VBELN = IT_LIKP-VBELN AND MATNR IN MATERIAL AND WERKS IN PLANT AND
  PSTYV IN ITEM_CAT AND VTWEG IN DIS_CHAN AND SPART IN DIVISION.
  SORT IT_LIPS BY VBELN POSNR.
  LOOP AT IT_LIPS INTO WA_LIPS.
    WA_OUTPUT-VBELN = WA_LIPS-VBELN.
    WA_OUTPUT-POSNR = WA_LIPS-POSNR.
    WA_OUTPUT-PSTYV = WA_LIPS-PSTYV.
    WA_OUTPUT-MATNR = WA_LIPS-MATNR.
    WA_OUTPUT-WERKS = WA_LIPS-WERKS.
    WA_OUTPUT-LGORT = WA_LIPS-LGORT.
    WA_OUTPUT-KDMAT = WA_LIPS-KDMAT.
    WA_OUTPUT-LFIMG = WA_LIPS-LFIMG.
    WA_OUTPUT-VRKME = WA_LIPS-VRKME.
    WA_OUTPUT-NTGEW = WA_LIPS-NTGEW.
    WA_OUTPUT-BRGEW = WA_LIPS-BRGEW.
    WA_OUTPUT-GEWEI = WA_LIPS-GEWEI.
    WA_OUTPUT-LGMNG = WA_LIPS-LGMNG.
    WA_OUTPUT-VBTYV = WA_LIPS-VBTYV.
    WA_OUTPUT-VGBEL = WA_LIPS-VGBEL.
    WA_OUTPUT-VGPOS = WA_LIPS-VGPOS.
    WA_OUTPUT-POSAR = WA_LIPS-POSAR.
    WA_OUTPUT-VTWEG = WA_LIPS-VTWEG.
    WA_OUTPUT-SPART = WA_LIPS-SPART.
    WA_OUTPUT-SHKZG = WA_LIPS-SHKZG.
    WA_OUTPUT-NETPR = WA_LIPS-NETPR.
    WA_OUTPUT-NETWR = WA_LIPS-NETWR.
    WA_OUTPUT-WAVWR = WA_LIPS-WAVWR.
    WA_OUTPUT-KZWI1 = WA_LIPS-KZWI1.
    WA_OUTPUT-KZWI2 = WA_LIPS-KZWI2.
    WA_OUTPUT-KZWI3 = WA_LIPS-KZWI3.
    WA_OUTPUT-KZWI4 = WA_LIPS-KZWI4.
    WA_OUTPUT-KZWI5 = WA_LIPS-KZWI5.
```

```
    WA_OUTPUT-KZWI6 = WA_LIPS-KZWI6.
    APPEND WA_OUTPUT TO IT_OUTPUT.
    CLEAR : WA_OUTPUT, WA_LIPS.
  ENDLOOP.
  LOOP AT IT_OUTPUT INTO WA_OUTPUT.
    READ TABLE IT_LIKP INTO WA_LIKP
    WITH KEY VBELN = WA_OUTPUT-VBELN.
    IF SY-SUBRC EQ 0.
      WA_OUTPUT-ERDAT = WA_LIKP-ERDAT.
      WA_OUTPUT-VKORG = WA_LIKP-VKORG.
      WA_OUTPUT-LFART = WA_LIKP-LFART.
      WA_OUTPUT-WADAT = WA_LIKP-WADAT.
      WA_OUTPUT-LFDAT = WA_LIKP-LFDAT.
      WA_OUTPUT-VBTYP = WA_LIKP-VBTYP.
      WA_OUTPUT-VSBED = WA_LIKP-VSBED.
      WA_OUTPUT-KUNNR = WA_LIKP-KUNNR.
      WA_OUTPUT-KUNAG = WA_LIKP-KUNAG.
      WA_OUTPUT-WADAT_IST = WA_LIKP-WADAT_IST.
    ENDIF.
    MODIFY IT_OUTPUT FROM WA_OUTPUT.
    CLEAR : WA_OUTPUT, WA_LIKP.
  ENDLOOP.
  DATA: ls_info     TYPE slis_listheader,
          lv_stext    TYPE scrtext_s,                                  " Short Text
          lv_mtext    TYPE scrtext_m,                                  " Medium
Text
          lv_ltext    TYPE scrtext_l,                                  " Long Text
          lv_colname  TYPE lvc_fname,                                  " Column
Name
          lv_position TYPE i VALUE 1,                                  " Position
          ls_line TYPE slis_listheader,                                "
          lv_char(15) TYPE c.
* Class declarations
  DATA: ls_rows        TYPE REF TO cl_salv_form_layout_grid,
        ls_row         TYPE REF TO cl_salv_form_layout_flow.
* Create Object Reference
  CREATE OBJECT ls_rows.
  TRY.
      cl_salv_table=>factory( IMPORTING r_salv_table = gr_table CHANGING t_table =
it_output ). "#EC
    CATCH cx_salv_msg.                                    "#EC
    CATCH cx_salv_not_found.                              "#EC
  ENDTRY.
  gr_columns = gr_table->get_columns( ).
  gr_columns->set_optimize( abap_true ).
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'VBELN'.
  lv_position = lv_position.
  lv_stext = 'Delivery No'(s28).
  lv_mtext = 'Delivery No'(s28).
  lv_ltext = 'Delivery No'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'POSNR'.
  lv_position = lv_position + 1.
  lv_stext = 'Delivery Item'(s28).
```

```
  lv_mtext = 'Delivery Item'(s28).
  lv_ltext = 'Delivery Item'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'POSNR'.
  lv_position = lv_position + 1.
  lv_stext = 'Delivery Item'(s28).
  lv_mtext = 'Delivery Item'(s28).
  lv_ltext = 'Delivery Item'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'ERDAT'.
  lv_position = lv_position + 1.
  lv_stext = 'Create Date'(s28).
  lv_mtext = 'Create Date'(s28).
  lv_ltext = 'Create Date'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'VKORG'.
  lv_position = lv_position + 1.
  lv_stext = 'Sales Org'(s28).
  lv_mtext = 'Sales Org'(s28).
  lv_ltext = 'Sales Org'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'WERKS'.
  lv_position = lv_position + 1.
  lv_stext = 'Plant'(s28).
  lv_mtext = 'Plant'(s28).
  lv_ltext = 'Plant'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'LFART'.
  lv_position = lv_position + 1.
  lv_stext = 'Delivery Type'(s28).
  lv_mtext = 'Delivery Type'(s28).
  lv_ltext = 'Delivery Type'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'WADAT'.
  lv_position = lv_position + 1.
  lv_stext = 'Planned Goods Issue Dt'(s28).
  lv_mtext = 'Planned Goods Issue Dt'(s28).
  lv_ltext = 'Planned Goods Issue Dt'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'LFDAT'.
  lv_position = lv_position + 1.
  lv_stext = 'Delivery Date'(s28).
  lv_mtext = 'Delivery Date'(s28).
  lv_ltext = 'Delivery Date'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'VBTYP'.
  lv_position = lv_position + 1.
  lv_stext = 'Sales Doc Categ'(s28).
```

```
  lv_mtext = 'Sales Doc Categ'(s28).
  lv_ltext = 'Sales Doc Categ'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'VSBED'.
  lv_position = lv_position + 1.
  lv_stext = 'Shipping Cond'(s28).
  lv_mtext = 'Shipping Cond'(s28).
  lv_ltext = 'Shipping Cond'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'KUNNR'.
  lv_position = lv_position + 1.
  lv_stext = 'Sold To'(s28).
  lv_mtext = 'Sold To'(s28).
  lv_ltext = 'Sold To'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'KUNAG'.
  lv_position = lv_position + 1.
  lv_stext = 'Ship To'(s28).
  lv_mtext = 'Ship To'(s28).
  lv_ltext = 'Ship To'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'WADAT_IST'.
  lv_position = lv_position + 1.
  lv_stext = 'Act Goods Issue Dt'(s28).
  lv_mtext = 'Act Goods Issue Dt'(s28).
  lv_ltext = 'Act Goods Issue Dt'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'PSTYV'.
  lv_position = lv_position + 1.
  lv_stext = 'Item Categ'(s28).
  lv_mtext = 'Item Categ'(s28).
  lv_ltext = 'Item Categ'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'MATNR'.
  lv_position = lv_position + 1.
  lv_stext = 'Material'(s28).
  lv_mtext = 'Material'(s28).
  lv_ltext = 'Material'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'LGORT'.
  lv_position = lv_position + 1.
  lv_stext = 'Storage Loc'(s28).
  lv_mtext = 'Storage Loc'(s28).
  lv_ltext = 'Storage Loc'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'KDMAT'.
  lv_position = lv_position + 1.
  lv_stext = 'Customer Mat No'(s28).
```

```
  lv_mtext = 'Customer Mat No'(s28).
  lv_ltext = 'Customer Mat No'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'LFIMG'.
  lv_position = lv_position + 1.
  lv_stext = 'Act Del Qty in SU'(s28).
  lv_mtext = 'Act Del Qty in SU'(s28).
  lv_ltext = 'Act Del Qty in SU'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'VRKME'.
  lv_position = lv_position + 1.
  lv_stext = 'Sales Unit'(s28).
  lv_mtext = 'Sales Unit'(s28).
  lv_ltext = 'Sales Unit'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'NTGEW'.
  lv_position = lv_position + 1.
  lv_stext = 'Net Weight'(s28).
  lv_mtext = 'Net Weight'(s28).
  lv_ltext = 'Net Weight'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'BRGEW'.
  lv_position = lv_position + 1.
  lv_stext = 'Gross Weight'(s28).
  lv_mtext = 'Gross Weight'(s28).
  lv_ltext = 'Gross Weight'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'GEWEI'.
  lv_position = lv_position + 1.
  lv_stext = 'Weight Unit'(s28).
  lv_mtext = 'Weight Unit'(s28).
  lv_ltext = 'Weight Unit'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'LGMNG'.
  lv_position = lv_position + 1.
  lv_stext = 'Act Del Qty in Stock Keeping'(s28).
  lv_mtext = 'Act Del Qty in Stock Keeping'(s28).
  lv_ltext = 'Act Del Qty in Stock Keeping'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'VBTYV'.
  lv_position = lv_position + 1.
  lv_stext = 'Sales Doc Categ'(s28).
  lv_mtext = 'Sales Doc Categ'(s28).
  lv_ltext = 'Sales Doc Categ'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
** Specify the Columnn name and position to be place in the ALV output.
*  lv_colname = 'KZWI1'.
*  lv_position = lv_position + 1.
*  lv_stext = 'Subtotal 1'(s28).
```

```
*  lv_mtext = 'Subtotal 1'(s28).
*  lv_ltext = 'Subtotal 1'(s29).
*  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
*
** Specify the Columnn name and position to be place in the ALV output.
*  lv_colname = 'KZWI2'.
*  lv_position = lv_position + 1.
*  lv_stext = 'Subtotal 2'(s28).
*  lv_mtext = 'Subtotal 2'(s28).
*  lv_ltext = 'Subtotal 2'(s29).
*  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
*
** Specify the Columnn name and position to be place in the ALV output.
*  lv_colname = 'KZWI3'.
*  lv_position = lv_position + 1.
*  lv_stext = 'Subtotal 3'(s28).
*  lv_mtext = 'Subtotal 3'(s28).
*  lv_ltext = 'Subtotal 3'(s29).
*  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
*
*
** Specify the Columnn name and position to be place in the ALV output.
*  lv_colname = 'KZWI4'.
*  lv_position = lv_position + 1.
*  lv_stext = 'Subtotal 4'(s28).
*  lv_mtext = 'Subtotal 4'(s28).
*  lv_ltext = 'Subtotal 4'(s29).
*  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
*
** Specify the Columnn name and position to be place in the ALV output.
*  lv_colname = 'KZWI5'.
*  lv_position = lv_position + 1.
*  lv_stext = 'Subtotal 5'(s28).
*  lv_mtext = 'Subtotal 5'(s28).
*  lv_ltext = 'Subtotal 5'(s29).
*  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
*
*
** Specify the Columnn name and position to be place in the ALV output.
*  lv_colname = 'KZWI6'.
*  lv_position = lv_position + 1.
*  lv_stext = 'Subtotal 6'(s28).
*  lv_mtext = 'Subtotal 6'(s28).
*  lv_ltext = 'Subtotal 6'(s29).
*  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
*
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'VGBEL'.
  lv_position = lv_position + 1.
  lv_stext = 'Refer Doc'(s28).
  lv_mtext = 'Refer Doc'(s28).
  lv_ltext = 'Refer Doc'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'VGPOS'.
  lv_position = lv_position + 1.
```

```
  lv_stext = 'Refer Item'(s28).
  lv_mtext = 'Refer Item'(s28).
  lv_ltext = 'Refer Item'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'POSAR'.
  lv_position = lv_position + 1.
  lv_stext = 'Item Type'(s28).
  lv_mtext = 'Item Type'(s28).
  lv_ltext = 'Item Type'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'VTWEG'.
  lv_position = lv_position + 1.
  lv_stext = 'Dist Chan'(s28).
  lv_mtext = 'Dist Chan'(s28).
  lv_ltext = 'Dist Chan'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'SPART'.
  lv_position = lv_position + 1.
  lv_stext = 'Division'(s28).
  lv_mtext = 'Division'(s28).
  lv_ltext = 'Division'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'SHKZG'.
  lv_position = lv_position + 1.
  lv_stext = 'Returns'(s28).
  lv_mtext = 'Returns'(s28).
  lv_ltext = 'Returns'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
** Specify the Columnn name and position to be place in the ALV output.
*  lv_colname = 'NETPR'.
*  lv_position = lv_position + 1.
*  lv_stext = 'Net Price'(s28).
*  lv_mtext = 'Net price'(s28).
*  lv_ltext = 'Net Price'(s29).
*  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
*
** Specify the Columnn name and position to be place in the ALV output.
*  lv_colname = 'WAVWR'.
*  lv_position = lv_position + 1.
*  lv_stext = 'Cost'(s28).
*  lv_mtext = 'Cost'(s28).
*  lv_ltext = 'Cost'(s29).
*  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
*
** Specify the Columnn name and position to be place in the ALV output.
*  lv_colname = 'NETWR'.
*  lv_position = lv_position + 1.
*  lv_stext = 'Net Value'(s28).
*  lv_mtext = 'Net Value'(s28).
*  lv_ltext = 'Net Value'(s29).
*  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
*
```

```
*
*
* Display the Header information.
  CALL METHOD gr_table->set_top_of_list
    EXPORTING
      value = ls_rows.
* Getting the All columns names
  gr_columns = gr_table->get_columns( ).
  gr_functions = gr_table->get_functions( ).
  gr_functions->set_all( abap_true ).
* To addd the sort functionality
  gr_sorts = gr_table->get_sorts( ).
*  gr_sorts->add_sort( columnname = 'VBELN'(s27) subtotal = abap_true ).
*  gr_sorts->add_sort( columnname = 'POSNR'(s30) subtotal = abap_true ).
  gr_agg = gr_table->get_aggregations( ).
  gr_agg->add_aggregation( 'LFIMG'(s66) ).
  gr_agg->add_aggregation( 'NTGEW'(s63) ).
  gr_agg->add_aggregation( 'BRGEW'(s46) ).
  gr_agg->add_aggregation( 'LGMNG'(s46) ).
*  gr_agg->add_aggregation( 'NETPR'(s46) ).
*  gr_agg->add_aggregation( 'NETWR'(s46) ).
*  gr_agg->add_aggregation( 'WAVWR'(s55) ).
*  gr_agg->add_aggregation( 'KZWI1'(s57) ).
*  gr_agg->add_aggregation( 'KZWI2'(s57) ).
*  gr_agg->add_aggregation( 'KZWI3'(s57) ).
*  gr_agg->add_aggregation( 'KZWI4'(s57) ).
*  gr_agg->add_aggregation( 'KZWI5'(s57) ).
*  gr_agg->add_aggregation( 'KZWI6'(s57) ).
*
*Display output table.
  gr_table->display( ).
*&---------------------------------------------------------------------*
*&      Form  F_SETCOLUMN_NAME
*&---------------------------------------------------------------------*
* Display the column according to the report output
*----------------------------------------------------------------------*
FORM f_setcolumn_name  USING    pu_colname  TYPE lvc_fname               " Field Name
                                pu_position TYPE i                      " Position
                                pu_lv_stext TYPE scrtext_s              " Short Text
                                pu_lv_mtext TYPE scrtext_m              " Medium
Text
                                pu_lv_ltext TYPE scrtext_l.             " Short Text
  TRY.
*     get the required column details to GR_COLUMN.
      gr_column = gr_columns->get_column( pu_colname ).
*     Short Text for column.
      gr_column->set_short_text( pu_lv_stext ).
*     Medium Text for column.
      gr_column->set_medium_text( pu_lv_mtext ).
*     Long Text for column.
      gr_column->set_long_text( pu_lv_ltext ).
*     Setting Column name and position.
      gr_columns->set_column_position(
                            columnname = pu_colname
                            position   = pu_position ).
    CATCH cx_salv_not_found.                                "#EC
```

```
  ENDTRY.
* clearing the variable used in the perform .
  CLEAR: pu_colname,pu_lv_stext,pu_lv_mtext,pu_lv_ltext.
ENDFORM.                    " F_SETCOLUMN_NAME
```

Billing Analysis Program Code

```
*&---------------------------------------------------------------------*
*& Report  ZBI_BILLING_ANALYSIS
*&
*&---------------------------------------------------------------------*
*&
*&
*&---------------------------------------------------------------------*
REPORT  ZBI_BILLING_ANALYSIS.
TABLES : VBRK,VBRP.
START-OF-SELECTION.
  SELECT-OPTIONS BIL_DOC FOR VBRK-VBELN.
  SELECT-OPTIONS CRTD_ON FOR SY-DATUM.
  SELECT-OPTIONS BILL_DT FOR SY-DATUM.
  SELECT-OPTIONS BIL_TYPE FOR VBRK-FKART.
  SELECT-OPTIONS BIL_CATG FOR VBRK-FKTYP.
  SELECT-OPTIONS COMP_CDE FOR VBRK-BUKRS.
  SELECT-OPTIONS SAL_ORG FOR VBRK-VKORG.
  SELECT-OPTIONS DIS_CHAN FOR VBRK-VTWEG.
  SELECT-OPTIONS DIVISION FOR VBRK-SPART.
  SELECT-OPTIONS MATERIAL FOR VBRP-MATNR.
  SELECT-OPTIONS PAYER FOR VBRK-KUNRG.
  SELECT-OPTIONS SOLD_TO FOR VBRK-KUNAG.
  SELECT-OPTIONS PLANT FOR VBRP-WERKS.
  SELECT-OPTIONS SERV_DT FOR VBRP-FBUDA.
END-OF-SELECTION.
  TYPES : BEGIN OF STR_BILL,
          VBELN TYPE VBRK-VBELN,
          POSNR TYPE VBRP-POSNR,
          FKART TYPE VBRK-FKART,
          FKTYP TYPE VBRK-FKTYP,
          VBTYP TYPE VBRK-VBTYP,
          VKORG TYPE VBRK-VKORG,
          VTWEG TYPE VBRK-VTWEG,
          KNUMV TYPE VBRK-KNUMV,
          VSBED TYPE VBRK-VSBED,
          FKDAT TYPE VBRK-FKDAT,
          BUKRS TYPE VBRK-BUKRS,
          ERDAT TYPE VBRK-ERDAT,
          KUNRG TYPE VBRK-KUNRG,
          KUNAG TYPE VBRK-KUNAG,
          BSTNK_VF TYPE VBRK-BSTNK_VF,
          FKIMG TYPE VBRP-FKIMG,
          VRKME TYPE VBRP-VRKME,
          FKLMG TYPE VBRP-FKLMG,
          NTGEW TYPE VBRP-NTGEW,
          BRGEW TYPE VBRP-BRGEW,
          GEWEI TYPE VBRP-GEWEI,
          PRSDT TYPE VBRP-PRSDT,
          FBUDA TYPE VBRP-FBUDA,
          NETWR TYPE VBRP-NETWR,
          VGBEL TYPE VBRP-VGBEL,
          VGPOS TYPE VBRP-VGPOS,
          VGTYP TYPE VBRP-VGTYP,
          AUBEL TYPE VBRP-AUBEL,
```

```
         AUPOS TYPE VBRP-AUPOS,
         MATNR TYPE VBRP-MATNR,
         MATKL TYPE VBRP-MATKL,
         PSTYV TYPE VBRP-PSTYV,
         POSAR TYPE VBRP-POSAR,
         SPART TYPE VBRP-SPART,
         WERKS TYPE VBRP-WERKS,
         SHKZG TYPE VBRP-SHKZG,
         WAVWR TYPE VBRP-WAVWR,
         KZWI1 TYPE VBRP-KZWI1,
         KZWI2 TYPE VBRP-KZWI2,
         KZWI3 TYPE VBRP-KZWI3,
         KZWI4 TYPE VBRP-KZWI4,
         KZWI5 TYPE VBRP-KZWI5,
         KZWI6 TYPE VBRP-KZWI6,
         END OF STR_BILL.
TYPES : BEGIN OF STR_VBRK,
        VBELN TYPE VBRK-VBELN,
        FKART TYPE VBRK-FKART,
        FKTYP TYPE VBRK-FKTYP,
        VBTYP TYPE VBRK-VBTYP,
        VKORG TYPE VBRK-VKORG,
        VTWEG TYPE VBRK-VTWEG,
        KNUMV TYPE VBRK-KNUMV,
        VSBED TYPE VBRK-VSBED,
        FKDAT TYPE VBRK-FKDAT,
        BUKRS TYPE VBRK-BUKRS,
        ERDAT TYPE VBRK-ERDAT,
        KUNRG TYPE VBRK-KUNRG,
        KUNAG TYPE VBRK-KUNAG,
        SPART TYPE VBRK-SPART,
        BSTNK_VF TYPE VBRK-BSTNK_VF,
        END OF STR_VBRK.
TYPES : BEGIN OF STR_VBRP,
         VBELN TYPE VBRP-VBELN,
         POSNR TYPE VBRP-POSNR,
         FKIMG TYPE VBRP-FKIMG,
         VRKME TYPE VBRP-VRKME,
         FKLMG TYPE VBRP-FKLMG,
         NTGEW TYPE VBRP-NTGEW,
         BRGEW TYPE VBRP-BRGEW,
         GEWEI TYPE VBRP-GEWEI,
         PRSDT TYPE VBRP-PRSDT,
         FBUDA TYPE VBRP-FBUDA,
         NETWR TYPE VBRP-NETWR,
         VGBEL TYPE VBRP-VGBEL,
         VGPOS TYPE VBRP-VGPOS,
         VGTYP TYPE VBRP-VGTYP,
         AUBEL TYPE VBRP-AUBEL,
         AUPOS TYPE VBRP-AUPOS,
         MATNR TYPE VBRP-VGBEL,
         MATKL TYPE VBRP-MATKL,
         PSTYV TYPE VBRP-PSTYV,
         POSAR TYPE VBRP-POSAR,
         SPART TYPE VBRP-SPART,
         WERKS TYPE VBRP-WERKS,
```

```
          SHKZG TYPE VBRP-SHKZG,
          WAVWR TYPE VBRP-WAVWR,
          KZWI1 TYPE VBRP-KZWI1,
          KZWI2 TYPE VBRP-KZWI2,
          KZWI3 TYPE VBRP-KZWI3,
          KZWI4 TYPE VBRP-KZWI4,
          KZWI5 TYPE VBRP-KZWI5,
          KZWI6 TYPE VBRP-KZWI6,
          END OF STR_VBRP.
  DATA : IT_VBRK TYPE TABLE OF STR_VBRK,
         IT_VBRP TYPE TABLE OF STR_VBRP,
         WA_VBRK TYPE STR_VBRK,
         WA_VBRP TYPE STR_VBRP,
         IT_BILL TYPE TABLE OF STR_BILL,
         WA_BILL TYPE STR_BILL,
         IT_OUTPUT TYPE TABLE OF STR_BILL,
         WA_OUTPUT TYPE STR_BILL.
  type-pools : slis .
*  data: fieldcatalog type slis_t_fieldcat_alv with header line,
*        gd_tab_group type slis_t_sp_group_alv,
*        gd_layout    type slis_layout_alv,
*        gd_repid     like sy-repid,
*        gt_events     type slis_t_event.
  DATA:   gr_table      TYPE REF TO cl_salv_table,
          gr_functions  TYPE REF TO cl_salv_functions,      "#EC *
          gr_columns    TYPE REF TO cl_salv_columns,        "#EC *
          gr_column     TYPE REF TO cl_salv_column,         "#EC *
          gr_sorts      TYPE REF TO cl_salv_sorts,
          gr_agg        TYPE REF TO cl_salv_aggregations.
  SELECT * FROM VBRK INTO CORRESPONDING FIELDS OF TABLE IT_VBRK WHERE
  VBELN IN BIL_DOC AND ERDAT IN CRTD_ON AND FKDAT IN BILL_DT AND
  FKART IN BIL_TYPE AND FKTYP IN BIL_CATG AND BUKRS IN COMP_CDE AND
  VKORG IN SAL_ORG AND VTWEG IN DIS_CHAN AND SPART IN DIVISION AND
  KUNRG IN PAYER AND KUNAG IN SOLD_TO.
  SELECT * FROM VBRP INTO CORRESPONDING FIELDS OF TABLE IT_VBRP
  FOR ALL ENTRIES IN IT_VBRK WHERE VBELN = IT_VBRK-VBELN AND
  MATNR IN MATERIAL AND WERKS IN PLANT AND FBUDA IN SERV_DT.
  LOOP AT IT_VBRP INTO WA_VBRP.
    WA_BILL-VBELN = WA_VBRP-VBELN.
    WA_BILL-POSNR = WA_VBRP-POSNR.
    WA_BILL-FKIMG = WA_VBRP-FKIMG.
    WA_BILL-VRKME = WA_VBRP-VRKME.
    WA_BILL-FKLMG = WA_VBRP-FKLMG.
    WA_BILL-NTGEW = WA_VBRP-NTGEW.
    WA_BILL-BRGEW = WA_VBRP-BRGEW.
    WA_BILL-GEWEI = WA_VBRP-GEWEI.
    WA_BILL-PRSDT = WA_VBRP-PRSDT.
    WA_BILL-FBUDA = WA_VBRP-FBUDA.
    WA_BILL-NETWR = WA_VBRP-NETWR.
    WA_BILL-VGBEL = WA_VBRP-VGBEL.
    WA_BILL-VGPOS = WA_VBRP-VGPOS.
    WA_BILL-VGTYP = WA_VBRP-VGTYP.
    WA_BILL-AUBEL = WA_VBRP-AUBEL.
    WA_BILL-AUPOS = WA_VBRP-AUPOS.
    WA_BILL-MATNR = WA_VBRP-MATNR.
    WA_BILL-MATKL = WA_VBRP-MATKL.
```

```
    WA_BILL-PSTYV = WA_VBRP-PSTYV.
    WA_BILL-POSAR = WA_VBRP-POSAR.
    WA_BILL-SPART = WA_VBRP-SPART.
    WA_BILL-WERKS = WA_VBRP-WERKS.
    WA_BILL-SHKZG = WA_VBRP-SHKZG.
    WA_BILL-WAVWR = WA_VBRP-WAVWR.
    WA_BILL-KZWI1 = WA_VBRP-KZWI1.
    WA_BILL-KZWI2 = WA_VBRP-KZWI2.
    WA_BILL-KZWI3 = WA_VBRP-KZWI3.
    WA_BILL-KZWI4 = WA_VBRP-KZWI4.
    WA_BILL-KZWI5 = WA_VBRP-KZWI5.
    WA_BILL-KZWI6 = WA_VBRP-KZWI6.
    APPEND WA_BILL TO IT_BILL.
    CLEAR : WA_BILL, WA_VBRP.
  ENDLOOP.
  LOOP AT IT_BILL INTO WA_BILL.
    READ TABLE IT_VBRK INTO WA_VBRK
    WITH KEY VBELN = WA_BILL-VBELN.
    IF SY-SUBRC EQ 0.
      WA_BILL-FKART = WA_VBRK-FKART.
      WA_BILL-FKTYP = WA_VBRK-FKTYP.
      WA_BILL-VBTYP = WA_VBRK-VBTYP.
      WA_BILL-VKORG = WA_VBRK-VKORG.
      WA_BILL-VTWEG = WA_VBRK-VTWEG.
      WA_BILL-VSBED = WA_VBRK-VSBED.
      WA_BILL-FKDAT = WA_VBRK-FKDAT.
      WA_BILL-BUKRS = WA_VBRK-BUKRS.
      WA_BILL-ERDAT = WA_VBRK-ERDAT.
      WA_BILL-KUNRG = WA_VBRK-KUNRG.
      WA_BILL-KUNAG = WA_VBRK-KUNAG.
      WA_BILL-SPART = WA_VBRK-SPART.
      WA_BILL-BSTNK_VF = WA_VBRK-BSTNK_VF.
      WA_BILL-KNUMV = WA_VBRK-KNUMV.
    ENDIF.
    MODIFY IT_BILL FROM WA_BILL.
    CLEAR : WA_BILL, WA_VBRK, WA_VBRP.
  ENDLOOP.
  IT_OUTPUT[] = IT_BILL[].
  DATA: ls_info     TYPE slis_listheader,
          lv_stext    TYPE scrtext_s,                                  " Short Text
          lv_mtext    TYPE scrtext_m,                                  " Medium
Text
          lv_ltext    TYPE scrtext_l,                                  " Long Text
          lv_colname  TYPE lvc_fname,                                  " Column
Name
          lv_position TYPE i VALUE 1,                                  " Position
          ls_line TYPE slis_listheader,                                "
          lv_char(15) TYPE c.
* Class declarations
  DATA: ls_rows        TYPE REF TO cl_salv_form_layout_grid,
        ls_row         TYPE REF TO cl_salv_form_layout_flow.
* Create Object Reference
  CREATE OBJECT ls_rows.
  TRY.
      cl_salv_table=>factory( IMPORTING r_salv_table = gr_table CHANGING t_table =
it_output ). "#EC
```

```
      CATCH cx_salv_msg.                                    "#EC
      CATCH cx_salv_not_found.                              "#EC
    ENDTRY.
    gr_columns = gr_table->get_columns( ).
    gr_columns->set_optimize( abap_true ).
  * Specify the Columnn name and position to be place in the ALV output.
    lv_colname = 'VBELN'.
    lv_position = 1.
    lv_stext = 'Billing Doc'(s28).
    lv_mtext = 'Billing Doc'(s28).
    lv_ltext = 'Billing Doc'(s29).
    PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
  * Specify the Columnn name and position to be place in the ALV output.
    lv_colname = 'POSNR'.
    lv_position = lv_position + 1.
    lv_stext = 'Billing Item'(s28).
    lv_mtext = 'Billing Item'(s28).
    lv_ltext = 'Billing Item'(s29).
    PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
  * Specify the Columnn name and position to be place in the ALV output.
    lv_colname = 'FKIMG'.
    lv_position = lv_position + 1.
    lv_stext = 'Act Inv Qty'(s28).
    lv_mtext = 'Act Inv Qty'(s28).
    lv_ltext = 'Act inv Qty'(s29).
    PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
  * Specify the Columnn name and position to be place in the ALV output.
    lv_colname = 'VRKME'.
    lv_position = lv_position + 1.
    lv_stext = 'Sales Unit'(s28).
    lv_mtext = 'Sales Unit'(s28).
    lv_ltext = 'Sales Unit'(s29).
    PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
  * Specify the Columnn name and position to be place in the ALV output.
    lv_colname = 'FKLMG'.
    lv_position = lv_position + 1.
    lv_stext = 'Bill Qty in Stock Keeping'(s28).
    lv_mtext = 'Bill Qty in Stock Keeping'(s28).
    lv_ltext = 'Bill Qty in Stock Keeping'(s29).
    PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
  * Specify the Columnn name and position to be place in the ALV output.
    lv_colname = 'NTGEW'.
    lv_position = lv_position + 1.
    lv_stext = 'Net Wt'(s28).
    lv_mtext = 'Net Wt'(s28).
    lv_ltext = 'Net Wt'(s29).
    PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
  * Specify the Columnn name and position to be place in the ALV output.
    lv_colname = 'BRGEW'.
    lv_position = lv_position + 1.
    lv_stext = 'Gross Wt'(s28).
    lv_mtext = 'Gross Wt'(s28).
    lv_ltext = 'Gross Wt'(s29).
    PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
  * Specify the Columnn name and position to be place in the ALV output.
    lv_colname = 'GEWEI'.
```

```
  lv_position = lv_position + 1.
  lv_stext = 'Wt Unit'(s28).
  lv_mtext = 'wt unit'(s28).
  lv_ltext = 'Wt Unit'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'PRSDT'.
  lv_position = lv_position + 1.
  lv_stext = 'Pricing Dt'(s28).
  lv_mtext = 'Pricing Dt'(s28).
  lv_ltext = 'Pricing Dt'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'FBUDA'.
  lv_position = lv_position + 1.
  lv_stext = 'Service Rendered Dt'(s28).
  lv_mtext = 'Service Rendered Dt'(s28).
  lv_ltext = 'Service Rendered Dt'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'NETWR'.
  lv_position = lv_position + 1.
  lv_stext = 'Net Value'(s28).
  lv_mtext = 'Net Value'(s28).
  lv_ltext = 'Net Value'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'VGBEL'.
  lv_position = lv_position + 1.
  lv_stext = 'Refer Doc'(s28).
  lv_mtext = 'Refer Doc'(s28).
  lv_ltext = 'Refer Doc'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'VGPOS'.
  lv_position = lv_position + 1.
  lv_stext = 'Refer Item'(s28).
  lv_mtext = 'Refer Item'(s28).
  lv_ltext = 'Refer Item'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'VGTYP'.
  lv_position = lv_position + 1.
  lv_stext = 'Preceeding Doc Categ'(s28).
  lv_mtext = 'Preceeding Doc Categ'(s28).
  lv_ltext = 'Preceeding Doc Categ'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'AUBEL'.
  lv_position = lv_position + 1.
  lv_stext = 'Sales Doc'(s28).
  lv_mtext = 'Sales Doc'(s28).
  lv_ltext = 'Sales Doc'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'AUPOS'.
```

```
  lv_position = lv_position + 1.
  lv_stext = 'Sales Item'(s28).
  lv_mtext = 'Sales Item'(s28).
  lv_ltext = 'Sales Item'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'MATNR'.
  lv_position = lv_position + 1.
  lv_stext = 'Material'(s28).
  lv_mtext = 'Material'(s28).
  lv_ltext = 'Material'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'MATKL'.
  lv_position = lv_position + 1.
  lv_stext = 'Material Grp'(s28).
  lv_mtext = 'Material Grp'(s28).
  lv_ltext = 'Material Grp'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'PSTYV'.
  lv_position = lv_position + 1.
  lv_stext = 'Sales Doc Item Categ'(s28).
  lv_mtext = 'Sales Doc Item Categ'(s28).
  lv_ltext = 'Sales Doc Item Categ'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'POSAR'.
  lv_position = lv_position + 1.
  lv_stext = 'Item Type'(s28).
  lv_mtext = 'Item Type'(s28).
  lv_ltext = 'Item Type'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'SPART'.
  lv_position = lv_position + 1.
  lv_stext = 'Division'(s28).
  lv_mtext = 'Division'(s28).
  lv_ltext = ''(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'WERKS'.
  lv_position = lv_position + 1.
  lv_stext = 'Plant'(s28).
  lv_mtext = 'Plant'(s28).
  lv_ltext = ''(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'SHKZG'.
  lv_position = lv_position + 1.
  lv_stext = 'Returns Item'(s28).
  lv_mtext = 'Returns Item'(s28).
  lv_ltext = 'Returns Item'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'WAVWR'.
```

```
  lv_position = lv_position + 1.
  lv_stext = 'Cost'(s28).
  lv_mtext = 'Cost'(s28).
  lv_ltext = 'Cost'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'KZWI1'.
  lv_position = lv_position + 1.
  lv_stext = 'Sub Total 1'(s28).
  lv_mtext = 'Sub Total 1'(s28).
  lv_ltext = 'Sub Total 1'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'KZWI2'.
  lv_position = lv_position + 1.
  lv_stext = 'Sub Total 2'(s28).
  lv_mtext = 'Sub Total 2'(s28).
  lv_ltext = 'Sub Total 2'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'KZWI3'.
  lv_position = lv_position + 1.
  lv_stext = 'Sub Total 3'(s28).
  lv_mtext = 'Sub Total 3'(s28).
  lv_ltext = 'Sub Total 3'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'KZWI4'.
  lv_position = lv_position + 1.
  lv_stext = 'Sub Total 4'(s28).
  lv_mtext = 'Sub Total 4'(s28).
  lv_ltext = 'Sub Total 4'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'KZWI5'.
  lv_position = lv_position + 1.
  lv_stext = 'Sub Total 5'(s28).
  lv_mtext = 'Sub Total 5'(s28).
  lv_ltext = 'Sub Total 5'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'KZWI6'.
  lv_position = lv_position + 1.
  lv_stext = 'Sub Total 6'(s28).
  lv_mtext = 'Sub Total 6'(s28).
  lv_ltext = 'Sub Total 6'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'FKART'.
  lv_position = lv_position + 1.
  lv_stext = 'Billing Type'(s28).
  lv_mtext = 'Billing Type'(s28).
  lv_ltext = 'Billing Type'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'FKTYP'.
```

```
  lv_position = lv_position + 1.
  lv_stext = 'Billing Categ'(s28).
  lv_mtext = 'Billing Categ'(s28).
  lv_ltext = 'Billing Categ'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'VBTYP'.
  lv_position = lv_position + 1.
  lv_stext = 'Sales Doc categ'(s28).
  lv_mtext = 'Sales Doc Categ'(s28).
  lv_ltext = 'Sales Doc Categ'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'VKORG'.
  lv_position = lv_position + 1.
  lv_stext = 'Sales Org'(s28).
  lv_mtext = 'Sales Org'(s28).
  lv_ltext = 'Sales Org'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'VTWEG'.
  lv_position = lv_position + 1.
  lv_stext = 'Dist Chan'(s28).
  lv_mtext = 'Dist Chan'(s28).
  lv_ltext = 'Dist Chan'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'VSBED'.
  lv_position = lv_position + 1.
  lv_stext = 'Ship Cond'(s28).
  lv_mtext = 'Ship Cond'(s28).
  lv_ltext = 'Ship Cond'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'FKDAT'.
  lv_position = lv_position + 1.
  lv_stext = 'Billing Date'(s28).
  lv_mtext = 'Billing Date'(s28).
  lv_ltext = 'Billing Date'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'BUKRS'.
  lv_position = lv_position + 1.
  lv_stext = 'Comp Code'(s28).
  lv_mtext = 'Comp Code'(s28).
  lv_ltext = ''(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'ERDAT'.
  lv_position = lv_position + 1.
  lv_stext = 'Created On'(s28).
  lv_mtext = 'Created On'(s28).
  lv_ltext = 'Created On'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'KUNRG'.
```

```
  lv_position = lv_position + 1.
  lv_stext = 'Sold To'(s28).
  lv_mtext = 'Sold To'(s28).
  lv_ltext = 'Sold To'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'KUNAG'.
  lv_position = lv_position + 1.
  lv_stext = 'Ship To'(s28).
  lv_mtext = 'Ship To'(s28).
  lv_ltext = 'Ship To'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'BSTNK_VF'.
  lv_position = lv_position + 1.
  lv_stext = 'Cust PO No'(s28).
  lv_mtext = 'Cust PO No'(s28).
  lv_ltext = 'Cust PO No'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Display the Header information.
  CALL METHOD gr_table->set_top_of_list
    EXPORTING
      value = ls_rows.
* Getting the All columns names
  gr_columns = gr_table->get_columns( ).
  gr_functions = gr_table->get_functions( ).
  gr_functions->set_all( abap_true ).
* To addd the sort functionality
  gr_sorts = gr_table->get_sorts( ).
*  gr_sorts->add_sort( columnname = 'VBELN'(s27) subtotal = abap_true ).
*  gr_sorts->add_sort( columnname = 'POSNR'(s30) subtotal = abap_true ).
  gr_agg = gr_table->get_aggregations( ).
  gr_agg->add_aggregation( 'NETWR'(s63) ).
  gr_agg->add_aggregation( 'WAVWR'(s55) ).
  gr_agg->add_aggregation( 'FKIMG'(s57) ).
  gr_agg->add_aggregation( 'KZWI1'(s57) ).
  gr_agg->add_aggregation( 'KZWI2'(s57) ).
  gr_agg->add_aggregation( 'KZWI3'(s57) ).
  gr_agg->add_aggregation( 'KZWI4'(s57) ).
  gr_agg->add_aggregation( 'KZWI5'(s57) ).
  gr_agg->add_aggregation( 'KZWI6'(s57) ).
*Display output table.
  gr_table->display( ).
*&---------------------------------------------------------------------*
*&      Form  F_SETCOLUMN_NAME
*&---------------------------------------------------------------------*
* Display the column according to the report output
*----------------------------------------------------------------------*
FORM f_setcolumn_name  USING    pu_colname  TYPE lvc_fname               " Field Name
                                pu_position TYPE i                      " Position
                                pu_lv_stext TYPE scrtext_s              " Short Text
                                pu_lv_mtext TYPE scrtext_m              " Medium
Text
                                pu_lv_ltext TYPE scrtext_l.             " Short Text
  TRY.
*     get the required column details to GR_COLUMN.
```

```
      gr_column = gr_columns->get_column( pu_colname ).
*     Short Text for column.
      gr_column->set_short_text( pu_lv_stext ).
*     Medium Text for column.
      gr_column->set_medium_text( pu_lv_mtext ).
*     Long Text for column.
      gr_column->set_long_text( pu_lv_ltext ).
*     Setting Column name and position.
      gr_columns->set_column_position(
                           columnname = pu_colname
                           position   = pu_position ).
    CATCH cx_salv_not_found.                                    "#EC
  ENDTRY.
* clearing the variable used in the perform .
  CLEAR: pu_colname,pu_lv_stext,pu_lv_mtext,pu_lv_ltext.
ENDFORM.                    " F_SETCOLUMN_NAME
```

Billing Conditions Analysis Program Code

```
*&---------------------------------------------------------------------*
*& Report  ZBI_BILLING_COND_ANALYSIS
*&
*&---------------------------------------------------------------------*
*&
*&
*&---------------------------------------------------------------------*
REPORT  ZBI_BILLING_COND_ANALYSIS.
TABLES : VBRK,VBRP,KONV.
START-OF-SELECTION.
  SELECT-OPTIONS BIL_DOC FOR VBRK-VBELN.
  SELECT-OPTIONS CRTD_ON FOR SY-DATUM.
  SELECT-OPTIONS BILL_DT FOR SY-DATUM.
  SELECT-OPTIONS BIL_TYPE FOR VBRK-FKART.
  SELECT-OPTIONS BIL_CATG FOR VBRK-FKTYP.
  SELECT-OPTIONS COMP_CDE FOR VBRK-BUKRS.
  SELECT-OPTIONS SAL_ORG FOR VBRK-VKORG.
  SELECT-OPTIONS DIS_CHAN FOR VBRK-VTWEG.
  SELECT-OPTIONS DIVISION FOR VBRK-SPART.
  SELECT-OPTIONS MATERIAL FOR VBRP-MATNR.
  SELECT-OPTIONS PAYER FOR VBRK-KUNRG.
  SELECT-OPTIONS SOLD_TO FOR VBRK-KUNAG.
  SELECT-OPTIONS PLANT FOR VBRP-WERKS.
  SELECT-OPTIONS SERV_DT FOR VBRP-FBUDA.
END-OF-SELECTION.
  TYPES : BEGIN OF STR_BILL,
          VBELN TYPE VBRK-VBELN,
          POSNR TYPE VBRP-POSNR,
          FKART TYPE VBRK-FKART,
          FKTYP TYPE VBRK-FKTYP,
          VBTYP TYPE VBRK-VBTYP,
          VKORG TYPE VBRK-VKORG,
          VTWEG TYPE VBRK-VTWEG,
          KNUMV TYPE VBRK-KNUMV,
          VSBED TYPE VBRK-VSBED,
          FKDAT TYPE VBRK-FKDAT,
          BUKRS TYPE VBRK-BUKRS,
          ERDAT TYPE VBRK-ERDAT,
          KUNRG TYPE VBRK-KUNRG,
          KUNAG TYPE VBRK-KUNAG,
          BSTNK_VF TYPE VBRK-BSTNK_VF,
          FKIMG TYPE VBRP-FKIMG,
          VRKME TYPE VBRP-VRKME,
          FKLMG TYPE VBRP-FKLMG,
          NTGEW TYPE VBRP-NTGEW,
          BRGEW TYPE VBRP-BRGEW,
          GEWEI TYPE VBRP-GEWEI,
          PRSDT TYPE VBRP-PRSDT,
          FBUDA TYPE VBRP-FBUDA,
          NETWR TYPE VBRP-NETWR,
          VGBEL TYPE VBRP-VGBEL,
          VGPOS TYPE VBRP-VGPOS,
          VGTYP TYPE VBRP-VGTYP,
          AUBEL TYPE VBRP-AUBEL,
```

```
          AUPOS TYPE VBRP-AUPOS,
          MATNR TYPE VBRP-MATNR,
          MATKL TYPE VBRP-MATKL,
          PSTYV TYPE VBRP-PSTYV,
          POSAR TYPE VBRP-POSAR,
          SPART TYPE VBRP-SPART,
          WERKS TYPE VBRP-WERKS,
          SHKZG TYPE VBRP-SHKZG,
          WAVWR TYPE VBRP-WAVWR,
          KZWI1 TYPE VBRP-KZWI1,
          KZWI2 TYPE VBRP-KZWI2,
          KZWI3 TYPE VBRP-KZWI3,
          KZWI4 TYPE VBRP-KZWI4,
          KZWI5 TYPE VBRP-KZWI5,
          KZWI6 TYPE VBRP-KZWI6,
          END OF STR_BILL.
TYPES :  BEGIN OF STR_BILL_COND,
          VBELN TYPE VBRK-VBELN,
          POSNR TYPE VBRP-POSNR,
          AUBEL TYPE VBRP-AUBEL,
          AUPOS TYPE VBRP-AUPOS,
          KSCHL TYPE KONV-KSCHL,
          FKART TYPE VBRK-FKART,
          FKTYP TYPE VBRK-FKTYP,
          VBTYP TYPE VBRK-VBTYP,
          VKORG TYPE VBRK-VKORG,
          VTWEG TYPE VBRK-VTWEG,
          VSBED TYPE VBRK-VSBED,
          FKDAT TYPE VBRK-FKDAT,
          BUKRS TYPE VBRK-BUKRS,
          ERDAT TYPE VBRK-ERDAT,
          KUNRG TYPE VBRK-KUNRG,
          KUNAG TYPE VBRK-KUNAG,
          BSTNK_VF TYPE VBRK-BSTNK_VF,
          FKIMG TYPE VBRP-FKIMG,
          VRKME TYPE VBRP-VRKME,
          FKLMG TYPE VBRP-FKLMG,
          NTGEW TYPE VBRP-NTGEW,
          BRGEW TYPE VBRP-BRGEW,
          GEWEI TYPE VBRP-GEWEI,
          PRSDT TYPE VBRP-PRSDT,
          FBUDA TYPE VBRP-FBUDA,
          NETWR TYPE VBRP-NETWR,
          VGBEL TYPE VBRP-VGBEL,
          VGPOS TYPE VBRP-VGPOS,
          VGTYP TYPE VBRP-VGTYP,
          MATNR TYPE VBRP-MATNR,
          MATKL TYPE VBRP-MATKL,
          PSTYV TYPE VBRP-PSTYV,
          POSAR TYPE VBRP-POSAR,
          SPART TYPE VBRP-SPART,
          WERKS TYPE VBRP-WERKS,
          SHKZG TYPE VBRP-SHKZG,
          WAVWR TYPE VBRP-WAVWR,
          KZWI1 TYPE VBRP-KZWI1,
          KZWI2 TYPE VBRP-KZWI2,
```

```
          KZWI3 TYPE VBRP-KZWI3,
          KZWI4 TYPE VBRP-KZWI4,
          KZWI5 TYPE VBRP-KZWI5,
          KZWI6 TYPE VBRP-KZWI6,
          KAWRT TYPE KONV-KAWRT,
          KBETR TYPE KONV-KBETR,
          WAERS TYPE KONV-WAERS,
          KPEIN TYPE KONV-KPEIN,
          KMEIN TYPE KONV-KMEIN,
          END OF STR_BILL_COND.
TYPES : BEGIN OF STR_VBRK,
        VBELN TYPE VBRK-VBELN,
        FKART TYPE VBRK-FKART,
        FKTYP TYPE VBRK-FKTYP,
        VBTYP TYPE VBRK-VBTYP,
        VKORG TYPE VBRK-VKORG,
        VTWEG TYPE VBRK-VTWEG,
        KNUMV TYPE VBRK-KNUMV,
        VSBED TYPE VBRK-VSBED,
        FKDAT TYPE VBRK-FKDAT,
        BUKRS TYPE VBRK-BUKRS,
        ERDAT TYPE VBRK-ERDAT,
        KUNRG TYPE VBRK-KUNRG,
        KUNAG TYPE VBRK-KUNAG,
        SPART TYPE VBRK-SPART,
        BSTNK_VF TYPE VBRK-BSTNK_VF,
        END OF STR_VBRK.
TYPES : BEGIN OF STR_VBRP,
        VBELN TYPE VBRP-VBELN,
        POSNR TYPE VBRP-POSNR,
        FKIMG TYPE VBRP-FKIMG,
        VRKME TYPE VBRP-VRKME,
        FKLMG TYPE VBRP-FKLMG,
        NTGEW TYPE VBRP-NTGEW,
        BRGEW TYPE VBRP-BRGEW,
        GEWEI TYPE VBRP-GEWEI,
        PRSDT TYPE VBRP-PRSDT,
        FBUDA TYPE VBRP-FBUDA,
        NETWR TYPE VBRP-NETWR,
        VGBEL TYPE VBRP-VGBEL,
        VGPOS TYPE VBRP-VGPOS,
        VGTYP TYPE VBRP-VGTYP,
        AUBEL TYPE VBRP-AUBEL,
        AUPOS TYPE VBRP-AUPOS,
        MATNR TYPE VBRP-VGBEL,
        MATKL TYPE VBRP-MATKL,
        PSTYV TYPE VBRP-PSTYV,
        POSAR TYPE VBRP-POSAR,
        SPART TYPE VBRP-SPART,
        WERKS TYPE VBRP-WERKS,
        SHKZG TYPE VBRP-SHKZG,
        WAVWR TYPE VBRP-WAVWR,
        KZWI1 TYPE VBRP-KZWI1,
        KZWI2 TYPE VBRP-KZWI2,
        KZWI3 TYPE VBRP-KZWI3,
        KZWI4 TYPE VBRP-KZWI4,
```

```
          KZWI5 TYPE VBRP-KZWI5,
          KZWI6 TYPE VBRP-KZWI6,
          END OF STR_VBRP.
  TYPES : BEGIN OF STR_KONV,
          KNUMV TYPE KONV-KNUMV,
          KPOSN TYPE KONV-KPOSN,
          STUNR TYPE KONV-STUNR,
          ZAEHK TYPE KONV-ZAEHK,
          KAPPL TYPE KONV-KAPPL,
          KSCHL TYPE KONV-KSCHL,
          KAWRT TYPE KONV-KAWRT,
          KBETR TYPE KONV-KBETR,
          WAERS TYPE KONV-WAERS,
          KPEIN TYPE KONV-KPEIN,
          KMEIN TYPE KONV-KMEIN,
          KNUMH TYPE KONV-KNUMH,
          KOPOS TYPE KONV-KOPOS,
          KINAK TYPE KONV-KINAK,
          END OF STR_KONV.
  DATA : IT_VBRK TYPE TABLE OF STR_VBRK,
         IT_VBRP TYPE TABLE OF STR_VBRP,
         IT_KONV TYPE TABLE OF STR_KONV,
         WA_VBRK TYPE STR_VBRK,
         WA_VBRP TYPE STR_VBRP,
         WA_KONV TYPE STR_KONV,
         IT_BILL TYPE TABLE OF STR_BILL,
         WA_BILL TYPE STR_BILL,
         IT_OUTPUT TYPE TABLE OF STR_BILL_COND,
         WA_OUTPUT TYPE STR_BILL_COND.
  type-pools : slis .
*  data: fieldcatalog type slis_t_fieldcat_alv with header line,
*        gd_tab_group type slis_t_sp_group_alv,
*        gd_layout    type slis_layout_alv,
*        gd_repid     like sy-repid,
*        gt_events     type slis_t_event.
  DATA:   gr_table      TYPE REF TO cl_salv_table,
          gr_functions  TYPE REF TO cl_salv_functions,      "#EC *
          gr_columns    TYPE REF TO cl_salv_columns,        "#EC *
          gr_column     TYPE REF TO cl_salv_column,         "#EC *
          gr_sorts      TYPE REF TO cl_salv_sorts,
          gr_agg        TYPE REF TO cl_salv_aggregations.
  SELECT * FROM VBRK INTO CORRESPONDING FIELDS OF TABLE IT_VBRK WHERE
  VBELN IN BIL_DOC AND ERDAT IN CRTD_ON AND FKDAT IN BILL_DT AND
  FKART IN BIL_TYPE AND FKTYP IN BIL_CATG AND BUKRS IN COMP_CDE AND
  VKORG IN SAL_ORG AND VTWEG IN DIS_CHAN AND SPART IN DIVISION AND
  KUNRG IN PAYER AND KUNAG IN SOLD_TO.
  SELECT * FROM VBRP INTO CORRESPONDING FIELDS OF TABLE IT_VBRP
  FOR ALL ENTRIES IN IT_VBRK WHERE VBELN = IT_VBRK-VBELN AND
  MATNR IN MATERIAL AND WERKS IN PLANT AND FBUDA IN SERV_DT.
  SELECT * FROM KONV INTO CORRESPONDING FIELDS OF TABLE
  IT_KONV FOR ALL ENTRIES IN IT_VBRK WHERE
  KNUMV = IT_VBRK-KNUMV AND KINAK NE 'X'.
  LOOP AT IT_VBRP INTO WA_VBRP.
    WA_BILL-VBELN = WA_VBRP-VBELN.
    WA_BILL-POSNR = WA_VBRP-POSNR.
    WA_BILL-FKIMG = WA_VBRP-FKIMG.
```

```
  WA_BILL-VRKME = WA_VBRP-VRKME.
  WA_BILL-FKLMG = WA_VBRP-FKLMG.
  WA_BILL-NTGEW = WA_VBRP-NTGEW.
  WA_BILL-BRGEW = WA_VBRP-BRGEW.
  WA_BILL-GEWEI = WA_VBRP-GEWEI.
  WA_BILL-PRSDT = WA_VBRP-PRSDT.
  WA_BILL-FBUDA = WA_VBRP-FBUDA.
  WA_BILL-NETWR = WA_VBRP-NETWR.
  WA_BILL-VGBEL = WA_VBRP-VGBEL.
  WA_BILL-VGPOS = WA_VBRP-VGPOS.
  WA_BILL-VGTYP = WA_VBRP-VGTYP.
  WA_BILL-AUBEL = WA_VBRP-AUBEL.
  WA_BILL-AUPOS = WA_VBRP-AUPOS.
  WA_BILL-MATNR = WA_VBRP-MATNR.
  WA_BILL-MATKL = WA_VBRP-MATKL.
  WA_BILL-PSTYV = WA_VBRP-PSTYV.
  WA_BILL-POSAR = WA_VBRP-POSAR.
  WA_BILL-SPART = WA_VBRP-SPART.
  WA_BILL-WERKS = WA_VBRP-WERKS.
  WA_BILL-SHKZG = WA_VBRP-SHKZG.
  WA_BILL-WAVWR = WA_VBRP-WAVWR.
  WA_BILL-KZWI1 = WA_VBRP-KZWI1.
  WA_BILL-KZWI2 = WA_VBRP-KZWI2.
  WA_BILL-KZWI3 = WA_VBRP-KZWI3.
  WA_BILL-KZWI4 = WA_VBRP-KZWI4.
  WA_BILL-KZWI5 = WA_VBRP-KZWI5.
  WA_BILL-KZWI6 = WA_VBRP-KZWI6.
  APPEND WA_BILL TO IT_BILL.
  CLEAR : WA_BILL, WA_VBRP.
ENDLOOP.
LOOP AT IT_BILL INTO WA_BILL.
  READ TABLE IT_VBRK INTO WA_VBRK
  WITH KEY VBELN = WA_BILL-VBELN.
  IF SY-SUBRC EQ 0.
    WA_BILL-FKART = WA_VBRK-FKART.
    WA_BILL-FKTYP = WA_VBRK-FKTYP.
    WA_BILL-VBTYP = WA_VBRK-VBTYP.
    WA_BILL-VKORG = WA_VBRK-VKORG.
    WA_BILL-VTWEG = WA_VBRK-VTWEG.
    WA_BILL-VSBED = WA_VBRK-VSBED.
    WA_BILL-FKDAT = WA_VBRK-FKDAT.
    WA_BILL-BUKRS = WA_VBRK-BUKRS.
    WA_BILL-ERDAT = WA_VBRK-ERDAT.
    WA_BILL-KUNRG = WA_VBRK-KUNRG.
    WA_BILL-KUNAG = WA_VBRK-KUNAG.
    WA_BILL-SPART = WA_VBRK-SPART.
    WA_BILL-BSTNK_VF = WA_VBRK-BSTNK_VF.
    WA_BILL-KNUMV = WA_VBRK-KNUMV.
  ENDIF.
  MODIFY IT_BILL FROM WA_BILL.
  CLEAR : WA_BILL, WA_VBRK, WA_VBRP.
ENDLOOP.
LOOP AT IT_KONV INTO WA_KONV.
  READ TABLE IT_BILL INTO WA_BILL WITH KEY KNUMV = WA_KONV-KNUMV
  POSNR = WA_KONV-KPOSN.
  WA_OUTPUT-VBELN = WA_BILL-VBELN.
```

```
    WA_OUTPUT-POSNR = WA_BILL-POSNR.
    WA_OUTPUT-KSCHL = WA_KONV-KSCHL.
    WA_OUTPUT-KAWRT = WA_KONV-KAWRT.
    WA_OUTPUT-KBETR = WA_KONV-KBETR.
    WA_OUTPUT-WAERS = WA_KONV-WAERS.
    WA_OUTPUT-KPEIN = WA_KONV-KPEIN.
    WA_OUTPUT-KMEIN = WA_KONV-KMEIN.
    WA_OUTPUT-FKIMG = WA_BILL-FKIMG.
    WA_OUTPUT-VRKME = WA_BILL-VRKME.
    WA_OUTPUT-FKLMG = WA_BILL-FKLMG.
    WA_OUTPUT-NTGEW = WA_BILL-NTGEW.
    WA_OUTPUT-BRGEW = WA_BILL-BRGEW.
    WA_OUTPUT-GEWEI = WA_BILL-GEWEI.
    WA_OUTPUT-PRSDT = WA_BILL-PRSDT.
    WA_OUTPUT-FBUDA = WA_BILL-FBUDA.
    WA_OUTPUT-NETWR = WA_BILL-NETWR.
    WA_OUTPUT-VGBEL = WA_BILL-VGBEL.
    WA_OUTPUT-VGPOS = WA_BILL-VGPOS.
    WA_OUTPUT-VGTYP = WA_BILL-VGTYP.
    WA_OUTPUT-AUBEL = WA_BILL-AUBEL.
    WA_OUTPUT-AUPOS = WA_BILL-AUPOS.
    WA_OUTPUT-MATNR = WA_BILL-MATNR.
    WA_OUTPUT-MATKL = WA_BILL-MATKL.
    WA_OUTPUT-PSTYV = WA_BILL-PSTYV.
    WA_OUTPUT-POSAR = WA_BILL-POSAR.
    WA_OUTPUT-SPART = WA_BILL-SPART.
    WA_OUTPUT-WERKS = WA_BILL-WERKS.
    WA_OUTPUT-SHKZG = WA_BILL-SHKZG.
    WA_OUTPUT-WAVWR = WA_BILL-WAVWR.
    WA_OUTPUT-KZWI1 = WA_BILL-KZWI1.
    WA_OUTPUT-KZWI2 = WA_BILL-KZWI2.
    WA_OUTPUT-KZWI3 = WA_BILL-KZWI3.
    WA_OUTPUT-KZWI4 = WA_BILL-KZWI4.
    WA_OUTPUT-KZWI5 = WA_BILL-KZWI5.
    WA_OUTPUT-KZWI6 = WA_BILL-KZWI6.
    WA_OUTPUT-FKART = WA_BILL-FKART.
    WA_OUTPUT-FKTYP = WA_BILL-FKTYP.
    WA_OUTPUT-VBTYP = WA_BILL-VBTYP.
    WA_OUTPUT-VKORG = WA_BILL-VKORG.
    WA_OUTPUT-VTWEG = WA_BILL-VTWEG.
    WA_OUTPUT-VSBED = WA_BILL-VSBED.
    WA_OUTPUT-FKDAT = WA_BILL-FKDAT.
    WA_OUTPUT-BUKRS = WA_BILL-BUKRS.
    WA_OUTPUT-ERDAT = WA_BILL-ERDAT.
    WA_OUTPUT-KUNRG = WA_BILL-KUNRG.
    WA_OUTPUT-KUNAG = WA_BILL-KUNAG.
    WA_OUTPUT-SPART = WA_BILL-SPART.
    WA_OUTPUT-BSTNK_VF = WA_BILL-BSTNK_VF.
    APPEND WA_OUTPUT TO IT_OUTPUT.
    CLEAR : WA_OUTPUT, WA_BILL, WA_KONV.
  ENDLOOP.
  DATA: ls_info     TYPE slis_listheader,
          lv_stext    TYPE scrtext_s,                                  " Short Text
          lv_mtext    TYPE scrtext_m,                                  " Medium
Text
          lv_ltext    TYPE scrtext_l,                                  " Long Text
```

```
          lv_colname  TYPE lvc_fname,                                    " Column
Name
          lv_position TYPE i VALUE 1,                                    " Position
          ls_line TYPE slis_listheader,                                  "
          lv_char(15) TYPE c.
* Class declarations
  DATA: ls_rows        TYPE REF TO cl_salv_form_layout_grid,
        ls_row         TYPE REF TO cl_salv_form_layout_flow.
* Create Object Reference
  CREATE OBJECT ls_rows.
  TRY.
      cl_salv_table=>factory( IMPORTING r_salv_table = gr_table CHANGING t_table =
it_output ). "#EC
    CATCH cx_salv_msg.                                          "#EC
    CATCH cx_salv_not_found.                                    "#EC
  ENDTRY.
  gr_columns = gr_table->get_columns( ).
  gr_columns->set_optimize( abap_true ).
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'VBELN'.
  lv_position = 1.
  lv_stext = 'Billing Doc'(s28).
  lv_mtext = 'Billing Doc'(s28).
  lv_ltext = 'Billing Doc'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'POSNR'.
  lv_position = lv_position + 1.
  lv_stext = 'Billing Item'(s28).
  lv_mtext = 'Billing Item'(s28).
  lv_ltext = 'Billing Item'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'KSCHL'.
  lv_position = lv_position + 1.
  lv_stext = 'Condition Type'(s28).
  lv_mtext = 'Condition Type'(s28).
  lv_ltext = 'Condition Type'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'KAWRT'.
  lv_position = lv_position + 1.
  lv_stext = 'Cond Base Value'(s28).
  lv_mtext = 'Cond Base Value'(s28).
  lv_ltext = 'Cond Base Value'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'KBETR'.
  lv_position = lv_position + 1.
  lv_stext = 'Rate'(s28).
  lv_mtext = 'Rate'(s28).
  lv_ltext = 'Rate'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'WAERS'.
  lv_position = lv_position + 1.
```

```
  lv_stext = 'Currency'(s28).
  lv_mtext = 'Currency'(s28).
  lv_ltext = 'Currency'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'KPEIN'.
  lv_position = lv_position + 1.
  lv_stext = 'Cond Pricing Unit'(s28).
  lv_mtext = 'Cond Pricing Unit'(s28).
  lv_ltext = 'Cond Pricing Unit'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'KMEIN'.
  lv_position = lv_position + 1.
  lv_stext = 'Condition Unit'(s28).
  lv_mtext = 'Condition Unit'(s28).
  lv_ltext = 'Condition Unit'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'FKIMG'.
  lv_position = lv_position + 1.
  lv_stext = 'Act Inv Qty'(s28).
  lv_mtext = 'Act Inv Qty'(s28).
  lv_ltext = 'Act inv Qty'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'VRKME'.
  lv_position = lv_position + 1.
  lv_stext = 'Sales Unit'(s28).
  lv_mtext = 'Sales Unit'(s28).
  lv_ltext = 'Sales Unit'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'FKLMG'.
  lv_position = lv_position + 1.
  lv_stext = 'Bill Qty in Stock Keeping'(s28).
  lv_mtext = 'Bill Qty in Stock Keeping'(s28).
  lv_ltext = 'Bill Qty in Stock Keeping'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'NTGEW'.
  lv_position = lv_position + 1.
  lv_stext = 'Net Wt'(s28).
  lv_mtext = 'Net Wt'(s28).
  lv_ltext = 'Net Wt'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'BRGEW'.
  lv_position = lv_position + 1.
  lv_stext = 'Gross Wt'(s28).
  lv_mtext = 'Gross Wt'(s28).
  lv_ltext = 'Gross Wt'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'GEWEI'.
  lv_position = lv_position + 1.
```

```
  lv_stext = 'Wt Unit'(s28).
  lv_mtext = 'wt unit'(s28).
  lv_ltext = 'Wt Unit'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'PRSDT'.
  lv_position = lv_position + 1.
  lv_stext = 'Pricing Dt'(s28).
  lv_mtext = 'Pricing Dt'(s28).
  lv_ltext = 'Pricing Dt'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'FBUDA'.
  lv_position = lv_position + 1.
  lv_stext = 'Service Rendered Dt'(s28).
  lv_mtext = 'Service Rendered Dt'(s28).
  lv_ltext = 'Service Rendered Dt'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'NETWR'.
  lv_position = lv_position + 1.
  lv_stext = 'Net Value'(s28).
  lv_mtext = 'Net Value'(s28).
  lv_ltext = 'Net Value'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'VGBEL'.
  lv_position = lv_position + 1.
  lv_stext = 'Refer Doc'(s28).
  lv_mtext = 'Refer Doc'(s28).
  lv_ltext = 'Refer Doc'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'VGPOS'.
  lv_position = lv_position + 1.
  lv_stext = 'Refer Item'(s28).
  lv_mtext = 'Refer Item'(s28).
  lv_ltext = 'Refer Item'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'VGTYP'.
  lv_position = lv_position + 1.
  lv_stext = 'Preceeding Doc Categ'(s28).
  lv_mtext = 'Preceeding Doc Categ'(s28).
  lv_ltext = 'Preceeding Doc Categ'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'AUBEL'.
  lv_position = lv_position + 1.
  lv_stext = 'Sales Doc'(s28).
  lv_mtext = 'Sales Doc'(s28).
  lv_ltext = 'Sales Doc'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'AUPOS'.
  lv_position = lv_position + 1.
```

```
  lv_stext = 'Sales Item'(s28).
  lv_mtext = 'Sales Item'(s28).
  lv_ltext = 'Sales Item'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'MATNR'.
  lv_position = lv_position + 1.
  lv_stext = 'Material'(s28).
  lv_mtext = 'Material'(s28).
  lv_ltext = 'Material'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'MATKL'.
  lv_position = lv_position + 1.
  lv_stext = 'Material Grp'(s28).
  lv_mtext = 'Material Grp'(s28).
  lv_ltext = 'Material Grp'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'PSTYV'.
  lv_position = lv_position + 1.
  lv_stext = 'Sales Doc Item Categ'(s28).
  lv_mtext = 'Sales Doc Item Categ'(s28).
  lv_ltext = 'Sales Doc Item Categ'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'POSAR'.
  lv_position = lv_position + 1.
  lv_stext = 'Item Type'(s28).
  lv_mtext = 'Item Type'(s28).
  lv_ltext = 'Item Type'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'SPART'.
  lv_position = lv_position + 1.
  lv_stext = 'Division'(s28).
  lv_mtext = 'Division'(s28).
  lv_ltext = ''(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'WERKS'.
  lv_position = lv_position + 1.
  lv_stext = 'Plant'(s28).
  lv_mtext = 'Plant'(s28).
  lv_ltext = ''(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'SHKZG'.
  lv_position = lv_position + 1.
  lv_stext = 'Returns Item'(s28).
  lv_mtext = 'Returns Item'(s28).
  lv_ltext = 'Returns Item'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'WAVWR'.
  lv_position = lv_position + 1.
```

```
  lv_stext = 'Cost'(s28).
  lv_mtext = 'Cost'(s28).
  lv_ltext = 'Cost'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'KZWI1'.
  lv_position = lv_position + 1.
  lv_stext = 'Sub Total 1'(s28).
  lv_mtext = 'Sub Total 1'(s28).
  lv_ltext = 'Sub Total 1'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'KZWI2'.
  lv_position = lv_position + 1.
  lv_stext = 'Sub Total 2'(s28).
  lv_mtext = 'Sub Total 2'(s28).
  lv_ltext = 'Sub Total 2'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'KZWI3'.
  lv_position = lv_position + 1.
  lv_stext = 'Sub Total 3'(s28).
  lv_mtext = 'Sub Total 3'(s28).
  lv_ltext = 'Sub Total 3'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'KZWI4'.
  lv_position = lv_position + 1.
  lv_stext = 'Sub Total 4'(s28).
  lv_mtext = 'Sub Total 4'(s28).
  lv_ltext = 'Sub Total 4'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'KZWI5'.
  lv_position = lv_position + 1.
  lv_stext = 'Sub Total 5'(s28).
  lv_mtext = 'Sub Total 5'(s28).
  lv_ltext = 'Sub Total 5'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'KZWI6'.
  lv_position = lv_position + 1.
  lv_stext = 'Sub Total 6'(s28).
  lv_mtext = 'Sub Total 6'(s28).
  lv_ltext = 'Sub Total 6'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'FKART'.
  lv_position = lv_position + 1.
  lv_stext = 'Billing Type'(s28).
  lv_mtext = 'Billing Type'(s28).
  lv_ltext = 'Billing Type'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'FKTYP'.
  lv_position = lv_position + 1.
```

```
  lv_stext = 'Billing Categ'(s28).
  lv_mtext = 'Billing Categ'(s28).
  lv_ltext = 'Billing Categ'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'VBTYP'.
  lv_position = lv_position + 1.
  lv_stext = 'Sales Doc categ'(s28).
  lv_mtext = 'Sales Doc Categ'(s28).
  lv_ltext = 'Sales Doc Categ'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'VKORG'.
  lv_position = lv_position + 1.
  lv_stext = 'Sales Org'(s28).
  lv_mtext = 'Sales Org'(s28).
  lv_ltext = 'Sales Org'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'VTWEG'.
  lv_position = lv_position + 1.
  lv_stext = 'Dist Chan'(s28).
  lv_mtext = 'Dist Chan'(s28).
  lv_ltext = 'Dist Chan'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'VSBED'.
  lv_position = lv_position + 1.
  lv_stext = 'Ship Cond'(s28).
  lv_mtext = 'Ship Cond'(s28).
  lv_ltext = 'Ship Cond'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'FKDAT'.
  lv_position = lv_position + 1.
  lv_stext = 'Billing Date'(s28).
  lv_mtext = 'Billing Date'(s28).
  lv_ltext = 'Billing Date'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'BUKRS'.
  lv_position = lv_position + 1.
  lv_stext = 'Comp Code'(s28).
  lv_mtext = 'Comp Code'(s28).
  lv_ltext = ''(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'ERDAT'.
  lv_position = lv_position + 1.
  lv_stext = 'Created On'(s28).
  lv_mtext = 'Created On'(s28).
  lv_ltext = 'Created On'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'KUNRG'.
  lv_position = lv_position + 1.
```

```
  lv_stext = 'Sold To'(s28).
  lv_mtext = 'Sold To'(s28).
  lv_ltext = 'Sold To'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'KUNAG'.
  lv_position = lv_position + 1.
  lv_stext = 'Ship To'(s28).
  lv_mtext = 'Ship To'(s28).
  lv_ltext = 'Ship To'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Specify the Columnn name and position to be place in the ALV output.
  lv_colname = 'BSTNK_VF'.
  lv_position = lv_position + 1.
  lv_stext = 'Cust PO No'(s28).
  lv_mtext = 'Cust PO No'(s28).
  lv_ltext = 'Cust PO No'(s29).
  PERFORM f_setcolumn_name USING lv_colname lv_position lv_stext lv_mtext lv_ltext.
* Display the Header information.
  CALL METHOD gr_table->set_top_of_list
    EXPORTING
      value = ls_rows.
* Getting the All columns names
  gr_columns = gr_table->get_columns( ).
  gr_functions = gr_table->get_functions( ).
  gr_functions->set_all( abap_true ).
* To addd the sort functionality
  gr_sorts = gr_table->get_sorts( ).
*  gr_sorts->add_sort( columnname = 'VBELN'(s27) subtotal = abap_true ).
*  gr_sorts->add_sort( columnname = 'POSNR'(s30) subtotal = abap_true ).
  gr_agg = gr_table->get_aggregations( ).
  gr_agg->add_aggregation( 'NETWR'(s63) ).
  gr_agg->add_aggregation( 'WAVWR'(s55) ).
  gr_agg->add_aggregation( 'FKIMG'(s57) ).
  gr_agg->add_aggregation( 'KZWI1'(s57) ).
  gr_agg->add_aggregation( 'KZWI2'(s57) ).
  gr_agg->add_aggregation( 'KZWI3'(s57) ).
  gr_agg->add_aggregation( 'KZWI4'(s57) ).
  gr_agg->add_aggregation( 'KZWI5'(s57) ).
  gr_agg->add_aggregation( 'KZWI6'(s57) ).
*Display output table.
  gr_table->display( ).
*&---------------------------------------------------------------------*
*&      Form  F_SETCOLUMN_NAME
*&---------------------------------------------------------------------*
* Display the column according to the report output
*----------------------------------------------------------------------*
FORM f_setcolumn_name  USING    pu_colname  TYPE lvc_fname               " Field Name
                                pu_position TYPE i                      " Position
                                pu_lv_stext TYPE scrtext_s              " Short Text
                                pu_lv_mtext TYPE scrtext_m              " Medium
Text
                                pu_lv_ltext TYPE scrtext_l.             " Short Text
  TRY.
*     get the required column details to GR_COLUMN.
      gr_column = gr_columns->get_column( pu_colname ).
```

```
*     Short Text for column.
      gr_column->set_short_text( pu_lv_stext ).
*     Medium Text for column.
      gr_column->set_medium_text( pu_lv_mtext ).
*     Long Text for column.
      gr_column->set_long_text( pu_lv_ltext ).
*     Setting Column name and position.
      gr_columns->set_column_position(
                           columnname = pu_colname
                           position   = pu_position ).
    CATCH cx_salv_not_found.                                "#EC
  ENDTRY.
* clearing the variable used in the perform .
  CLEAR: pu_colname,pu_lv_stext,pu_lv_mtext,pu_lv_ltext.
ENDFORM.                    " F_SETCOLUMN_NAME
```

Related Content

BW Data Modeling

http://help.sap.com/saphelp_nw04s/helpdata/en/e3/e60138fede083de10000009b38f8cf/frameset.htm

Query Design

http://help.sap.com/saphelp_nw04s/helpdata/en/e3/e60138fede083de10000009b38f8cf/frameset.htm

For more information, visit the Business Intelligence homepage.

Disclaimer and Liability Notice

This document may discuss sample coding or other information that does not include SAP official interfaces and therefore is not supported by SAP. Changes made based on this information are not supported and can be overwritten during an upgrade.

SAP will not be held liable for any damages caused by using or misusing the information, code or methods suggested in this document, and anyone using these methods does so at his/her own risk.

SAP offers no guarantees and assumes no responsibility or liability of any type with respect to the content of this technical article or code sample, including any liability resulting from incompatibility between the content within this document and the materials and services offered by SAP. You agree that you will not hold, or seek to hold, SAP responsible or liable with respect to the content of this document.