A **report** is a presentation of data in an organized structure. Many database management systems include a report writer that enables you to design and generate reports. SAP applications support report creation.

These reports consist of only one screen as an output. To create a classical report we use events classical reports is also known as event driven programming each event has its own importance during the creation of a classical report. Each event is associated with user action and is triggered only when the user performs that action.

Basically SAP users can use 3 types of reports. They are:

1. Classic reports
2. Interactive report
3. ALV Reports

In classic reports, we can see the output in single list where as in interactive reports we can have one basic list and twenty secondary lists interactive reports is also known as drill down reporting. ALV report consists of some pre-defined options like sort, filters, sum, downloading, print, changing the layout structure and many more.

Use of Report Programs

- They are used when data from a number of tables have to be selected and combined in the form of report.
- Used when the report has to be downloaded from SAP to an Excel sheet to be distributed across.
- Used when the report has to be mailed to a particular person.

Important Points In Report Program

- Transaction code for program creation is SE38.

- Report Programs are always Executable Programs. Program Type is always 1.
- Every Report program corresponds to a particular Application Type i.e. either with Sales & Distribution, FI - CO etc. It can also be Cross Application i.e. type '*'.
- Report Programming is an Event-driven programming.

- The first line of a report program is always Report ***<report-name>.***
- To suppress the list heading or the name of the program the addition ***No Standard Page Heading*** is used.
- The line size for a particular report can be set by using the addition ***line-size <size>.***
- The line count for a particular page can be set by using the addition ***line-count n(n1)***. N is the number of lines for the page and N1 is the number of lines reserved for the page footer.
- To display any information or error message we add a message class to the program using the addition: ***Message-id <message class name>.*** Message classes are maintained in SE91.

1. Classical Reports

A classical **report** is created by using the output data in the WRITE statement inside a loop.

Events in Classical Reports

Below are the list and **sequence** of events available , each event has it's own importance .

Load-of-praogram

This event is used to load program into memory for execution and this is the first event in execution sequence.

Initialization

This event is used to initialize variables, screen default values and other default actions.

At Selection-Screen output

By using this event we can manipulate dynamic selection-screen changes.

At Selection-Screen on field

This event is used to validate a single selection-screen input parameter.

```
Syntax: AT SELECTION-SCREEN ON <parameter name>. "Validate a input parameter
```

At Selection-Screen on value request

This event is used to provide value help (F4 help) for a input field.

```
Syntax: AT SELECTION-SCREEN ON VALUE REQUEST FOR <parameter name>. "Input search help fo
r a input parameters
```

At Selection-Screen on help request

By using this event we can provide F1 help for a input field.

```
Syntax: AT SELECTION-SCREEN ON HELP REQUEST FOR <parameter name>. "Input (F1) help for a i
nput parameters
```

At Selection-Screen

This event is used to validate multiple input fields

```
Syntax: AT SELECTION-SCREEN . "used to validate multiple input fields
```

Start-of-Selection

This is default event which is used to write actual business logic.

```
Syntax: START-OF-SELECTION. "Default event
```

End-of-Selection

We can use this event just to state that start-of-selection is ended, this event is used with logical databases, logical databases are in HR ABAP only. In normal ABAP we don`t have much importance .

```
Syntax: END-OF-SELECTION . "Start of selection is ended
```

Top-of-Page

This event prints constant heading for all pages.

```
Syntax: TOP-OF-PAGE."Page heading
```

End-of-Page

This event prints constant footer for all pages.

Before using this event, we need to reserve some lines for displaying footer.

```
Syntax: END-OF-PAGE . "Page footer

Example: REPORT ZPROGRAM LINE-COUNT 27(3). " Here we reserve 3 lines for footer
```

Program: Develop a Classical Report to display list of materials for a material input range(select-Options) and for a material type (screen input) with validations, search help and value help, the report can be able to download into excel sheet.

Input elements : Select-options for matnr (material no for MARA table), parameter mtart(Material type from MARA), a check box and a parameter limit for limiting no of results.

Whenever we click on download data, select file to download field will be enabled otherwise this should be disabled.

```
REPORT ZGTP_CLASSICAL_REPORT LINE-COUNT 34(2). "34 lines are for report space and 2 lines ar
e for footer space

TABLES : MARA.

TYPES: BEGIN OF TY_MARA,

       MATNR TYPE MARA-MATNR,

       ERSDA TYPE MARA-ERSDA,

       MTART TYPE MARA-MTART,

       MBRSH TYPE MARA-MBRSH,

       MATKL TYPE MARA-MATKL,

       MEINS TYPE MARA-MEINS,

      END OF TY_MARA.

DATA: IT_MARA TYPE TABLE OF TY_MARA. "material out put internal table

DATA:  WA_MARA TYPE TY_MARA. " work area

DATA: LV_MTART TYPE MARA-MTART.

DATA: LV_START_TIME TYPE SY-UZEIT.

DATA: LV_END_TIME TYPE SY-UZEIT.
```

```
SELECTION-SCREEN BEGIN OF BLOCK B1 WITH FRAME TITLE TEXT-001. "designs a block just for d
esign double click on TEXT-001 to add text


SELECT-OPTIONS: S_MATNR FOR MARA-MATNR. " Material range input

PARAMETERS: P_MTART TYPE MARA-MTART. "material type input


SELECTION-SCREEN END OF BLOCK B1.

PARAMETERS  P_DLOAD AS CHECKBOX USER-COMMAND UC1.

PARAMETERS  P_FILE TYPE RLGRAP-FILENAME MODIF ID DLD.

PARAMETERS P_LIMIT TYPE I. "Limit no of rows to display to avoid the burden on database


LOAD-OF-PROGRAM. "loads program into memory

  LV_START_TIME = SY-UZEIT. " see system variables www.sapnuts.com/resourse/system-variable.html


INITIALIZATION. "triggers second

  P_MTART = 'FERT'. "MATERIAL TYPE DEFAULT VALUE

  P_LIMIT = '50'. "Limit rows to 50


AT SELECTION-SCREEN OUTPUT. "For dynamic modifications

  IF P_DLOAD IS INITIAL.

    LOOP AT SCREEN.

      CHECK SCREEN-GROUP1 = 'DLD'.

      SCREEN-INPUT = '0'.

      MODIFY SCREEN.

    ENDLOOP.

  ENDIF.

AT SELECTION-SCREEN ON P_MTART. " Validate single input field at selection-screen is an alternative
 and good see http://www.sapnuts.com/courses/core-abap/classical-reports/selection-screen-event.html

AT SELECTION-SCREEN ON VALUE-REQUEST FOR P_MTART. "This event is not required here will u
se in the next lesson

  PERFORM MTART_VALUE_HELP.

AT SELECTION-SCREEN ON VALUE-REQUEST FOR P_FILE.

  PERFORM FILE_VALUE_HELP.

AT SELECTION-SCREEN ON HELP-REQUEST FOR P_MTART. " Provide help request F1 help.

  PERFORM MTART_HELP.
```

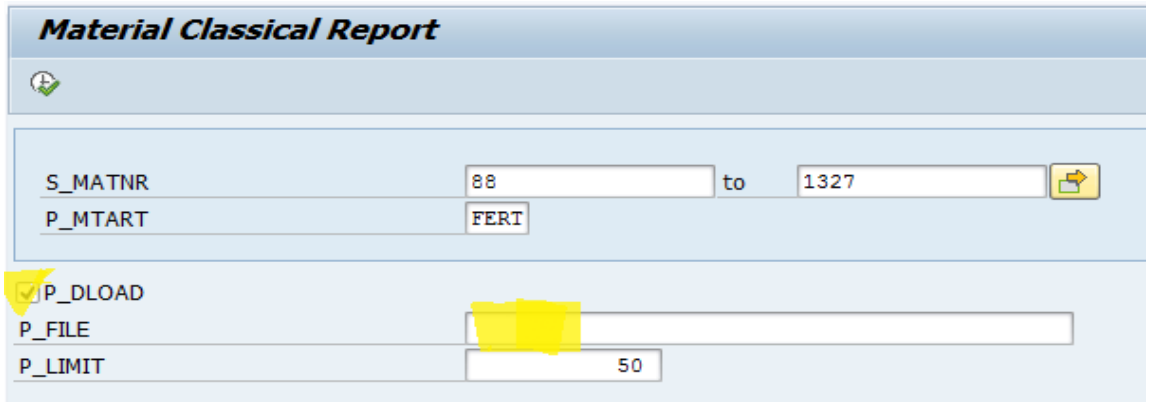
```
AT SELECTION-SCREEN.
  PERFORM VALIDATE_INPUTS.
START-OF-SELECTION.
  PERFORM GET_MATERIALS.
END-OF-SELECTION.
  LV_END_TIME = SY-UZEIT.
  PERFORM DISPLAY_OUTPUT.
  IF P_DLOAD = 'X'.
    PERFORM DOWNLOAD_DATA.
  ENDIF.
TOP-OF-PAGE.
  WRITE: 'Material Details ' COLOR 2.
END-OF-PAGE.
  WRITE:'The above materials are active materials available in database' COLOR 3.
  WRITE: 'Start time'.
  WRITE: LV_START_TIME.
  WRITE: 'End time'.
  WRITE: LV_END_TIME.
FORM VALIDATE_INPUTS.
  IF S_MATNR IS INITIAL OR P_MTART IS INITIAL.
    MESSAGE 'Please enter required inputs' TYPE 'E'.
  ELSE.
***Validate material type is valid or not
    SELECT MTART FROM MARA INTO LV_MTART
      UP TO 1 ROWS WHERE MTART = P_MTART.
    ENDSELECT.
    IF LV_MTART IS INITIAL.
      MESSAGE 'Material type is not available in MARA' TYPE 'E'.
    ENDIF.
  ENDIF.
ENDFORM.                    " VALIDATE_INPUTS
FORM GET_MATERIALS.
  SELECT MATNR ERSDA MTART MBRSH MATKL MEINS FROM MARA
     INTO TABLE IT_MARA
```

```
    UP TO P_LIMIT ROWS
    WHERE MATNR IN S_MATNR AND MTART = P_MTART.
ENDFORM.                    " GET_MATERIALS
FORM DISPLAY_OUTPUT .
  IF IT_MARA IS NOT INITIAL.
    LOOP AT IT_MARA INTO WA_MARA.
      WRITE :/ WA_MARA-MATNR, WA_MARA-ERSDA, WA_MARA-MTART, WA_MARA-MBRSH, WA_
MARA-MATKL, WA_MARA-MEINS .
    ENDLOOP.
  ELSE.
    WRITE :'No Data Found for your Query'.
  ENDIF.
ENDFORM.                    " DISPLAY_OUTPUT
FORM MTART_HELP.
  MESSAGE 'Enter a Material Type ' TYPE 'I'.
ENDFORM.                    " MTART_HELP
FORM MTART_VALUE_HELP.
  MESSAGE 'Material type input ex: FERT' TYPE 'I'.
ENDFORM.                    " MTART_VSLUE_HELP
FORM DOWNLOAD_DATA .
  DATA : LV_FILE TYPE STRING .
  LV_FILE = P_FILE .
  CALL FUNCTION 'GUI_DOWNLOAD'
    EXPORTING
*   BIN_FILESIZE                    =
      FILENAME                      = LV_FILE
     FILETYPE                       = 'ASC'
*   APPEND                         = ' '
     WRITE_FIELD_SEPARATOR          = 'X'
*   HEADER                         = '00'
*   TRUNC_TRAILING_BLANKS          = ' '
*   WRITE_LF                       = 'X'
*   COL_SELECT                     = ' '
*   COL_SELECT_MASK                = ' '
```

```
*   DAT_MODE                    = ' '
*   CONFIRM_OVERWRITE             = ' '
*   NO_AUTH_CHECK                 = ' '
*   CODEPAGE                    = ' '
*   IGNORE_CERR                   = ABAP_TRUE
*   REPLACEMENT                   = '#'
*   WRITE_BOM                   = ' '
*   TRUNC_TRAILING_BLANKS_EOL       = 'X'
*   WK1_N_FORMAT                  = ' '
*   WK1_N_SIZE                  = ' '
*   WK1_T_FORMAT                  = ' '
*   WK1_T_SIZE                  = ' '
*   WRITE_LF_AFTER_LAST_LINE        = ABAP_TRUE
*   SHOW_TRANSFER_STATUS          = ABAP_TRUE
* IMPORTING
*   FILELENGTH                  =
    TABLES
      DATA_TAB                  = IT_MARA
*   FIELDNAMES                  =
* EXCEPTIONS
*   FILE_WRITE_ERROR              = 1
*   NO_BATCH                    = 2
*   GUI_REFUSE_FILETRANSFER         = 3
*   INVALID_TYPE                  = 4
*   NO_AUTHORITY                  = 5
*   UNKNOWN_ERROR                 = 6
*   HEADER_NOT_ALLOWED            = 7
*   SEPARATOR_NOT_ALLOWED         = 8
*   FILESIZE_NOT_ALLOWED          = 9
*   HEADER_TOO_LONG               = 10
*   DP_ERROR_CREATE               = 11
*   DP_ERROR_SEND                 = 12
*   DP_ERROR_WRITE                = 13
*   UNKNOWN_DP_ERROR              = 14
```

```
*   ACCESS_DENIED                    = 15
*   DP_OUT_OF_MEMORY                 = 16
*   DISK_FULL                        = 17
*   DP_TIMEOUT                       = 18
*   FILE_NOT_FOUND                   = 19
*   DATAPROVIDER_EXCEPTION           = 20
*   CONTROL_FLUSH_ERROR              = 21
*   OTHERS                           = 2     .
  IF SY-SUBRC = 0.
    WRITE :/ 'Data downloaded to'.
    WRITE:P_FILE.
  ENDIF.
ENDFORM.                    " DOWNLOAD_DATA
FORM FILE_VALUE_HELP.
  CALL FUNCTION 'F4_FILENAME'
    EXPORTING
      FIELD_NAME = 'P_FILE'
    IMPORTING
      FILE_NAME  = P_FILE.
ENDFORM.                    " FILE_VALUE_HEL
```

Input Screen: when user clicks on P_DLOAD checkbox then only the P_FILE is enabled.



Output:

```
GTP Academy First Report
Material Details
88                    27.05.1997 FERT M 02004      ST
89                    27.05.1997 FERT M 02004      ST
578                   21.05.2002 FERT M 00207      ST
679                   25.10.2002 FERT M 012        ST
938                   28.07.2004 FERT M 00108      EA
1009                  01.10.2004 FERT P 017        EA
1012                  01.10.2004 FERT P 017        EA
1289                  28.04.2006 FERT M ZRAIL_MAT  ST
1301                  02.05.2006 FERT M ZRAIL_MAT  ST
1304                  02.05.2006 FERT M ZRAIL_MAT  ST
1308                  02.05.2006 FERT M ZRAIL_MAT  ST
1309                  02.05.2006 FERT M ZRAIL_MAT  M3
1310                  02.05.2006 FERT M ZRAIL_MAT  SET
1311                  02.05.2006 FERT M ZRAIL_MAT  ST
1312                  02.05.2006 FERT M ZRAIL_MAT  ST
1313                  02.05.2006 FERT M ZRAIL_MAT  ST
1314                  02.05.2006 FERT M ZRAIL_MAT  FT
1315                  02.05.2006 FERT M ZRAIL_MAT  FT
1316                  02.05.2006 FERT M ZRAIL_MAT  ST
1317                  02.05.2006 FERT M ZRAIL_MAT  ST
1318                  02.05.2006 FERT M ZRAIL_MAT  ST
1319                  02.05.2006 FERT M ZRAIL_MAT  ST
1320                  02.05.2006 FERT M ZRAIL_MAT  ST
1321                  02.05.2006 FERT M ZRAIL_MAT  ST
1322                  02.05.2006 FERT M ZRAIL_MAT  ST
1323                  02.05.2006 FERT M ZRAIL_MAT  ST
1324                  02.05.2006 FERT M ZRAIL_MAT  ST
1325                  02.05.2006 FERT M ZRAIL_MAT  MET
1326                  02.05.2006 FERT M ZRAIL_MAT  ST
1327                  02.05.2006 FERT M ZRAIL_MAT  ST
```

Using Selection Screen output for dynamic dynamic modifications on selection screen in SAP ABAP

At Selection Screen output is a selection-screen event, which is used to manipulate dynamic changes on selection-screen.

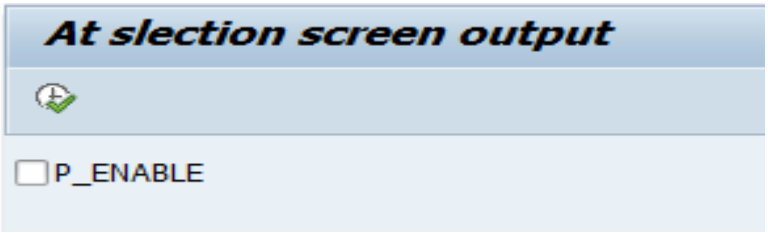
Loop At Screen. Screen is structure with Name, Group1, Group2, Group3, Group4, invisible, active, intensified etc fields, this holds the screen information at run time, Loop at Screen...Endloop. is used to loop through screen elements, and based on the values in above structure (SCREEN) we can manipulate changes.

MODIF ID : MODIF ID is a three character id (without quotes), we can process a screen elements group using this MODIF ID, this will be stored in SCREEN-GROUP1.

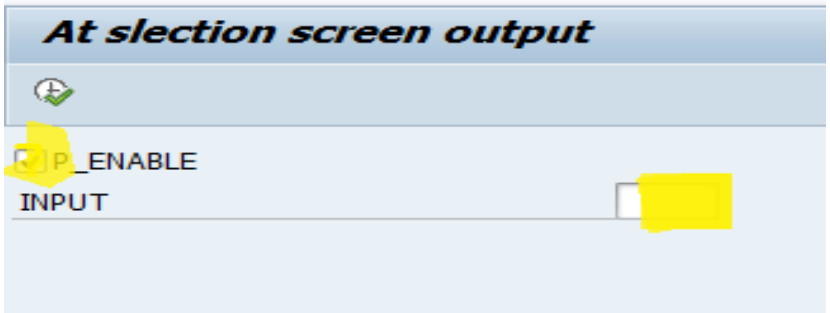
All Screen modifications should be done under AT SELECTION-SCREEN OUTPUT event only.

MODIFY SCREEN is keyword which is used to apply screen modification.

The below is the sample code for dynamic screen modification .



When user check enable input field check box, one input field will be enabled.




```
REPORT  ZGTP_SELECTION_SCREEN_OUTPUT.
PARAMETERS P_ENABLE AS CHECKBOX USER-COMMAND UC1.
PARAMETERS: INPUT(5) TYPE C MODIF ID IN1 .          "Based on modify id we will perform dynamic
                                                          operations
AT SELECTION-SCREEN OUTPUT.
  LOOP AT SCREEN.
    IF P_ENABLE = 'X'  . " If check box is selected
      IF SCREEN-GROUP1 = 'IN1' .
        SCREEN-ACTIVE = 1.
        MODIFY SCREEN.
      ENDIF.
    ELSE.
      IF SCREEN-GROUP1 = 'IN1' .
        SCREEN-ACTIVE = 0.
        MODIFY SCREEN.
      ENDIF.
    ENDIF.
  ENDLOOP.
```

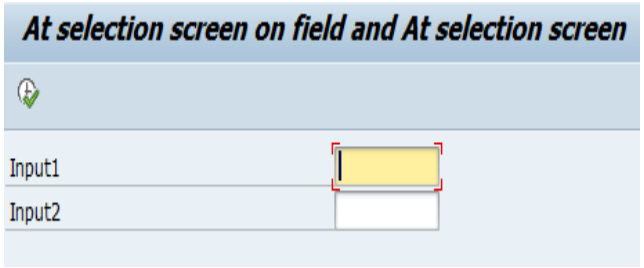
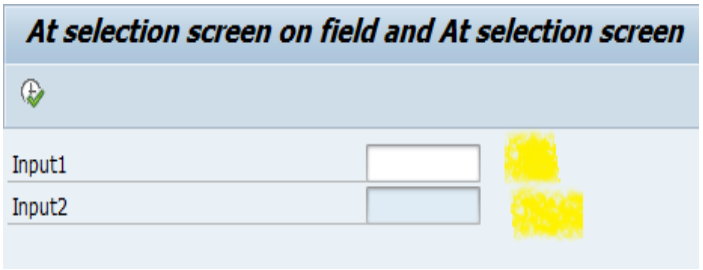
Difference between at selection screen on field and at selection screen events in SAP ABAP programming.

At selection screen on the field and at selection screen are selection-screen events which are used for input validations in SAP report programming.

At Selection Scteen on Field	**At Selection Screen**
This event is used to validate a single input field.	This event is used to validate multiple input fields.
If we this event, if any, error the error field will be highlighted and the remaining fields will be disabled.	By using this event, the error field is heightened and all the remaining fields will be enabled.

Example to explain At Selection-Screen on the field and At Selection-Screen difference.

In the below example we are going to validate two input fields and we will see the difference in screen behavior

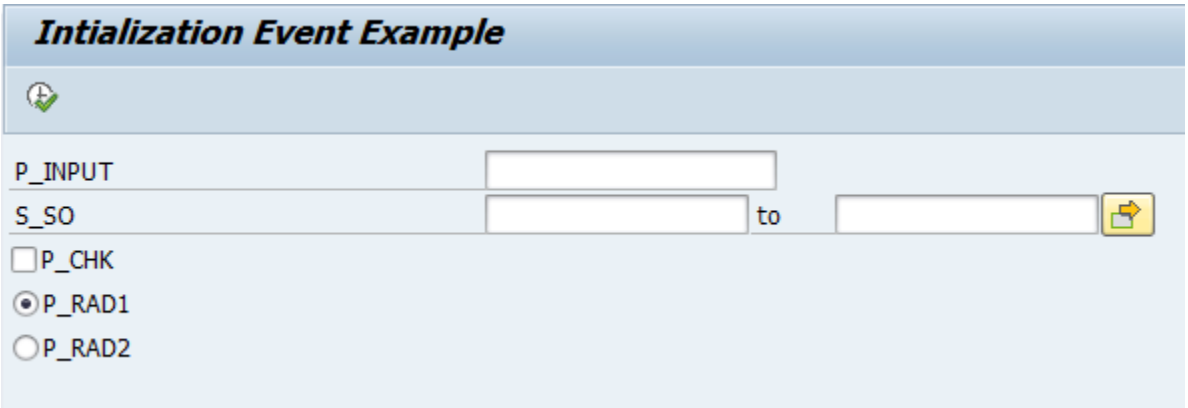
At Selection Screen	**At Selection Screen on Field**
REPORT ZGPN_SELECTION_SCREEN_EVENT. PARAMETERS P_FIELD1 TYPE CHAR10 . PARAMETERS P_FIELD2 TYPE CHAR10. AT SELECTION-SCREEN. IF P_FIELD1 IS INITIAL. MESSAGE 'Please enter field1' TYPE 'E'. ENDIF. IF P_FIELD2 IS INITIAL. MESSAGE 'Please enter field2' TYPE 'E'. ENDIF.	REPORT ZGPN_SELECTION_SCREEN_EVENT. PARAMETERS P_FIELD1 TYPE CHAR10 . PARAMETERS P_FIELD2 TYPE CHAR10. AT SELECTION-SCREEN ON P_FIELD1. IF P_FIELD1 IS INITIAL. MESSAGE 'Please enter field1' TYPE 'E'. ENDIF. AT SELECTION-SCREEN ON P_FIELD2. IF P_FIELD2 IS INITIAL. MESSAGE 'Please enter field2' TYPE 'E'. ENDIF.
	
After error message both input fields are enabled for input.	After error message all input fields are disabled, only the error field will be enables and heighlated .

Set default values on selection-screen using INITIALIZATION event in SAP ABAP programming

Most of the times in real-time business requirements we need to set default values on selection-screen elements like input fields, check boxes, select-options, radio buttons etc. We use the initialization event to set default values on selection screen.

```
REPORT ZGPN_INITIALIZATION.

TABLES: MARA. "tables decleration for select-options

PARAMETERS : P_INPUT TYPE CHAR20. "Inputfied with 20 character length.

SELECT-OPTIONS: S_SO FOR MARA-MATNR. "select-options

PARAMETERS P_CHK AS CHECKBOX. "check box

PARAMETERS P_RAD1 RADIOBUTTON GROUP RB1.

PARAMETERS P_RAD2 RADIOBUTTON GROUP RB1.
```

By using the above code we can display the below screen.



But we need to set default values to the input field, select-options(High, low), check box(default select), select radio button default.

Set Default values for parameters using INITIALIZATION

The below example explains you how to set default values using INITIALIZATION event to set default values on selection screen.

```
INITIALIZATION. "This event will trigger first

P_INPUT = 'GPN Academy'.
```

Set default values for Select-Options using Â initialization event

Technically select-options are nothing but internal tables with four fields, the four fields are SIGN(Stores I - Inclusive of Defined Values/Range and E - Exclusive of Defined Values/Range, OPTION(Stores EQ - Equal, NE- Not Equal, LT - Lower Than, LE - Lower Equal, GT - Greater Than, GE - Greater Equal, BT -

Between, CP - Covers Pattern and NP - Does Not Cover Pattern,LOW(Stores low value), HIGH(Stores high value), the below code is used to set default values for select-options.

```
INITIALIZATION.
  S_SO-LOW = '1'.
  S_SO-OPTION = 'BT'.
  S_SO-SIGN = 'I'.
  S_SO-HIGH = '100'.
  APPEND S_SO. " append select-options to screen
```

Set Default values for check box and radio buttons group

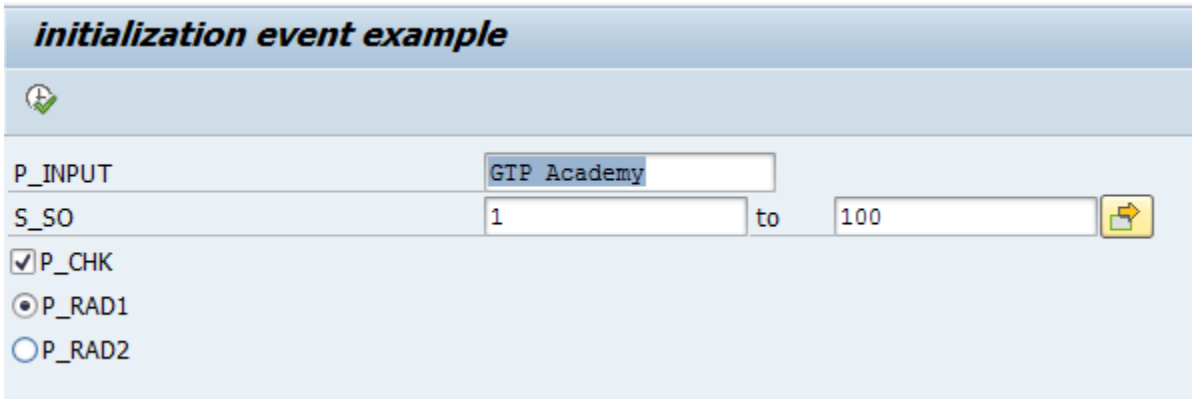
Check box and radio buttons store either X or space, X means selected, space means not selected, use the below code to default check box and radio buttons.

```
INITIALIZATION.
  P_CHK = 'X'.
  P_RAD2 = 'X'.
```

The final code will be

```
REPORT ZGTP_INITIALIZATION.
TABLES : MARA. "tables decleration for select-options
PARAMETERS: P_INPUT TYPE CHAR20. "Input fied with 20 character length.
SELECT-OPTIONS: S_SO FOR MARA-MATNR. "select-options
PARAMETERS P_CHK AS CHECKBOX. "check box
PARAMETERS P_RAD1 RADIOBUTTON GROUP RB1.
PARAMETERS P_RAD2 RADIOBUTTON GROUP RB1.
INITIALIZATION.
  P_INPUT = 'GTP Academy'.
  S_SO-LOW = '1'.
  S_SO-OPTION = 'BT'.
  S_SO-SIGN = 'I'.
  S_SO-HIGH = '100'.
  APPEND S_SO.
  P_CHK = 'X'.
```

Output:



2. Interactive Reports

As the name suggests, the user can interact with the report. We can have a drill down into the report data. For example, Column one of the report displays the material numbers, and the user feels that he needs some more specific data about the vendor for that material, he can HIDE that data under those material numbers.

And when the user clicks the material number, another report (actually sub report/secondary list) which displays the vendor details will be displayed.

We can have one basic list (number starts from 0) and 20 secondary lists (1 to 21).

Events In Interactive Reports:

1. AT LINE-SELECTION
2. AT USER-COMMAND
3. AT PF<key>
4. TOP-OF-PAGE DURING LINE-SELECTION.

HIDE statement holds the data to be displayed in the secondary list.

Sy-lisel: contains data of the selected line.

Sy-lsind: contains the level of report (from 0 to 21)

Interactive Report Events:

At Line-Selection

This event will trigger whenever the user double click on any list line.

```
Syntax: AT LINE-SELECTION. "Triggers line selection
```

At User Command

This event will trigger whenever user clicks on any custom buttons of the GUI.

```
Syntax: AT USER-COMMAND. "Triggers user command
```

At PF Status

This event will trigger whenever user clicks on any function buttons.

```
Syntax: AT PF <function key>. "Triggers user command
```

Top of Page During line selection

This is used to print heading for secondary lists in interactive reports.

```
Syntax: TOP-OF-PAGE DURING LINE-SELECTION. "Prints secondary list header
```

Techniques used in interactive reporting

Hide area

It is a key word which is used to store the data into a temporary memory call as HIDE area.

Functionality of HIDE is

- Whenever the user uses the HIDE statement, the data will be stored in 'HIDE' area along with line numbers.
- Whenever user double clicks on any list line the system takes the line number and checks the HIDE area for the corresponding data in that particular line, then the data will be returned to the HIDE variables.

```
Syntax: HIDE <WA>. "Store total work area in hide area

        OR

       HIDE <WA-FIELD>. "Store work area field in hide area
```

Program: Develop an interactive report to display material basic details in basic list, material plant details in secondary list for a material type input and display header and footer for primary and secondary list.

Requirement Analysis: In the above requirement, we have to get material details for a material type input (Parameter input for MTART field), whenever the user double clicks on any record of basic list, it will go to the second screen and display list plants for that material, display page header and footer for the report.

SAP Tables to be used are: MARA (Material Master), MARC (Material Plants).

Step1: Define report heading.

Go to SE38, create a program ZGTP_INTERACTIVE_REPORT. In order to display footer information we have to provide some space for the footer, it can be defined in the report definition (First line of the report), to provide space for footer we use below syntax.

```
REPORT ZGTP_INTERACTIVE_REPORT LINE-COUNT 34(2) NO STANDARD PAGE HEADING. "Leav
e some pages for footer and hide standard heading
```

In the above deceleration we have provided 34 lines for report and 2 lines for footer i.e., 34(2), we don`t need a standard page heading so we used NO STANDARD PAGE HEADING.

Step2: Data decelerations and Selection screen.

Declare the required internal tables, work areas, variables etc. and add selection screen element parameter for material type input.

```
DATA: IT_MARA TYPE TABLE OF MARA, "Mara internal table
      WA_MARA TYPE MARA, "Mara work area
      IT_MARC TYPE TABLE OF MARC, "marc internal table
      WA_MARC TYPE MARC. "Marc work area

PARAMETERS P_MTART TYPE MARA-MTART. "Selection screen element input field
```

Step3: Add code to get material basic details.

Add logic to get materials for the material type input under START-OF-SELECTION event.

```
START-OF-SELECTION.
  SELECT * FROM MARA
    INTO TABLE IT_MARA
    WHERE MTART = P_MTART.
```

Display materials and use HIDE technique

Display materials and use HIDE technique (HIDE area) to store line data.

```
  LOOP AT IT_MARA INTO WA_MARA.
    WRITE:/ WA_MARA-MATNR, WA_MARA-MTART, WA_MARA-MATKL, WA_MARA-MBRSH
    HIDE WA_MARA. "Store line details in HIDE area
  ENDLOOP.
```

Step4: Get plant details using hide area

Get material plants from MARC table based on HIDE area storage under AT LINE-SELECTION event.

```
AT LINE-SELECTION.
  SELECT * FROM MARC
    INTO TABLE IT_MARC
    WHERE MATNR = WA_MARA-MATNR.
```

Step5: Display plant data

Display material plant data.

```
  LOOP AT IT_MARC INTO WA_MARC.
    WRITE :/ WA_MARC-MATNR, WA_MARC-WERKS.
  ENDLOOP.
```

Step6: Display top of page for basic list and secondary list

Display page heading for basic list under TOP-OF-PAGE event and display secondary list heading under TOP-OF-PAGE DURING LINE-SELECTION event.

```
TOP-OF-PAGE.
  WRITE: 'Material Basic Details' COLOR 5.
TOP-OF-PAGE DURING LINE-SELECTION.
  WRITE: 'List of Plants for material:' WA_MARA-MATNR COLOR 6.
```

Step7: Display footer for basic list

Display footer information for basic material list.

```
  WRITE: 'Report Generated at:' SY-DATUM COLOR 1.
```

Final report after modularization is below

```
REPORT ZGTP_INTERACTIVE_REPORT LINE-COUNT 33(3) NO STANDARD PAGE HEADING. "leave
 some pages for footer and hide standard heading
DATA: IT_MARA TYPE TABLE OF MARA, "Mara internal table
      WA_MARA TYPE MARA, "Mara work area
      IT_MARC TYPE TABLE OF MARC, "marc internal table
      WA_MARC TYPE MARC. "marc work area
```

```
PARAMETERS P_MTART TYPE MARA-MTART. "selection screen element input field
INITIALIZATION. "initialization event
AT SELECTION-SCREEN. "at selection screen event to validate inputs
  PERFORM VALIDATE_INPUT. "Subroutine to validate input
START-OF-SELECTION.
  PERFORM GET_MATERIAL_DATA.
  PERFORM DISPLAY_MATERIALS.
TOP-OF-PAGE.
  PERFORM DISPLAY_HEADER.
END-OF-PAGE.
  PERFORM DISPLAY_FOOTER.
AT LINE-SELECTION.
  PERFORM GET_PLANT_DATA.
  PERFORM DISPLAY_PLANT_DATA.
TOP-OF-PAGE DURING LINE-SELECTION.
  PERFORM DISPLAY_LIST_HEADER.
FORM VALIDATE_INPUT.
  IF P_MTART IS INITIAL.
    MESSAGE ‘Please enters input' TYPE 'E'.
  ENDIF.
ENDFORM.                    “VALIDATE_INPUT
FORM GET_MATERIAL_DATA.
  SELECT * FROM MARA
    INTO TABLE IT_MARA
    UP TO 50 ROWS
    WHERE MTART = P_MTART.
ENDFORM.                    "GET_MATERIAL_DATA
FORM DISPLAY_MATERIALS.
  LOOP AT IT_MARA INTO WA_MARA.
    WRITE: / WA_MARA-MATNR, WA_MARA-MTART, WA_MARA-MATKL, WA_MARA-MBRSH.
    HIDE WA_MARA. "store line details in HIDE area
  ENDLOOP.
```

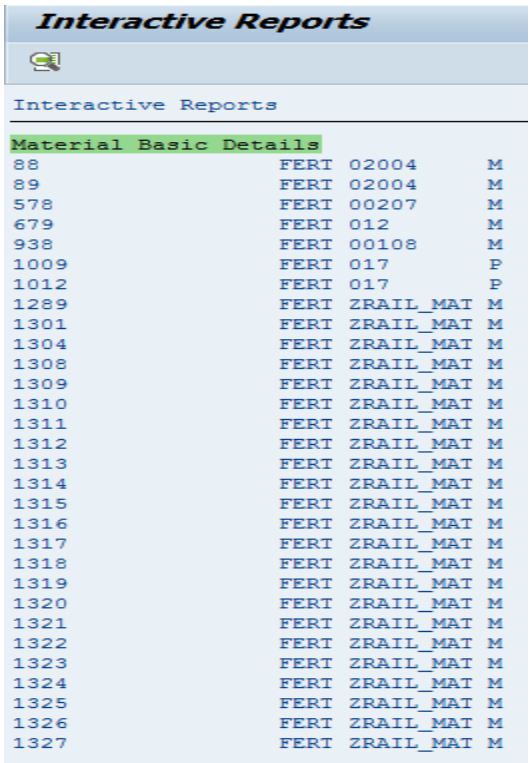
```
ENDFORM.                    "DISPLAY_MATERIALS
FORM DISPLAY_HEADER.
  WRITE: 'Material Basic Details' COLOR 5.
ENDFORM.                    "DISPLAY_HEADER
FORM DISPLAY_FOOTER.
  WRITE: 'Report Generated at:', SY-DATUM COLOR 1.
ENDFORM.                    "DISPLAY_FOOTER
FORM GET_PLANT_DATA.
  SELECT * FROM MARC
    INTO TABLE IT_MARC
    WHERE MATNR = WA_MARA-MATNR.
ENDFORM.                    "GET_PLANT_DATA
FORM DISPLAY_PLANT_DATA.
  LOOP AT IT_MARC INTO WA_MARC.
    WRITE: / WA_MARC-MATNR, WA_MARC-WERKS.
  ENDLOOP.
ENDFORM.                    "DISPLAY_PLANT_DATA
FORM DISPLAY_LIST_HEADER.
  WRITE: 'List of Plants for material:', WA_MARA-MATNR COLOR 6.
ENDFORM.                    "DISPLAY_LIST_HEADER
```

Testing:

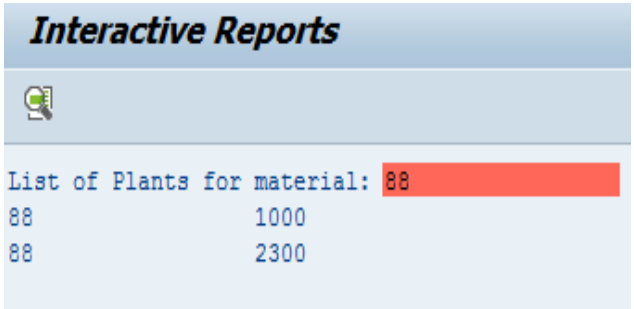
To test the above report goes to MARA table (SE11-MARA-DISPLAY-CONTENETS), get a material type ex: FERT, HALB etc, execute the report, provide material type and execute. The list of materials will be displayed; double click on any record, the corresponding material plants will be displayed in secondary list.

Output:

Basic List:



Secondary List:



GET CURSOR

This key word is used to read the field name and field value where the mouse cursor is placed or double clicks action is raised. It doesn't use hide area.

```
Syntax: GET CURSOR FIELD <V_FIELDNAME>,
                FIELDVALUE <V_FIELDVALUE>.
```

Program: Develop a material master report, which displays a list of materials for a range of materials (select-options input).

1. If the user double clicks on material number, it should display the details of that material in secondary list.

2. If the user clicks on material type, it should display all the materials of that material type.

Requirement Analysis: To fulfill this requirement, we need to get material details from MARA for selection options input and need to get the click position of basic list (whether user clicked on material no or material type) and need to display data based on user click.

GET CURSOR is a key word, which is used to get cursor position with field name and value, by using this key work we can get value and field name at cursor.

SAP Tables to be used: for the above requirement, we are going to use MARA table.

For this requirement, everything is same as previous program except some changes at line selection event. At line selection we use key word GET CURSOR.

```
DATA: FNAM (30), FVAL (50).

  GET CURSOR FIELD FNAM VALUE FVAL.

  CONDENSE FNAM.

  CONDENSE FVAL.

  IF FNAM = 'WA_MARA-MATNR'.

    SELECT SINGLE * FROM MARA INTO WA_MARA WHERE MATNR = FVAL.

    WRITE: / WA_MARA-MATNR, WA_MARA-MBRSH, WA_MARA-MTART, WA_MARA-MATKL, WA_M
ARA-MEINS, WA_MARA-ERSDA, WA_MARA-ERNAM.

  ELSEIF FNAM = 'WA_MARA-MTART'.

    SELECT * FROM MARA INTO TABLE IT_MARA UP TO 50 ROWS WHERE MTART = FVAL.

    LOOP AT IT_MARA INTO WA_MARA.

      WRITE: / WA_MARA-MATNR, WA_MARA-MTART, WA_MARA-MATKL.

    ENDLOOP.

  ENDIF.
```

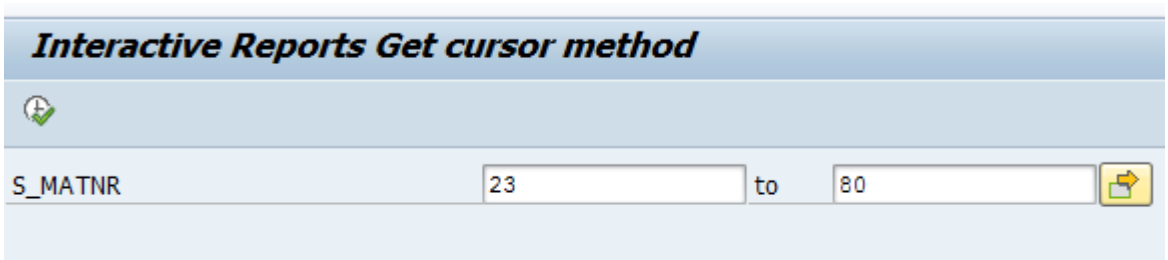
Final code will be

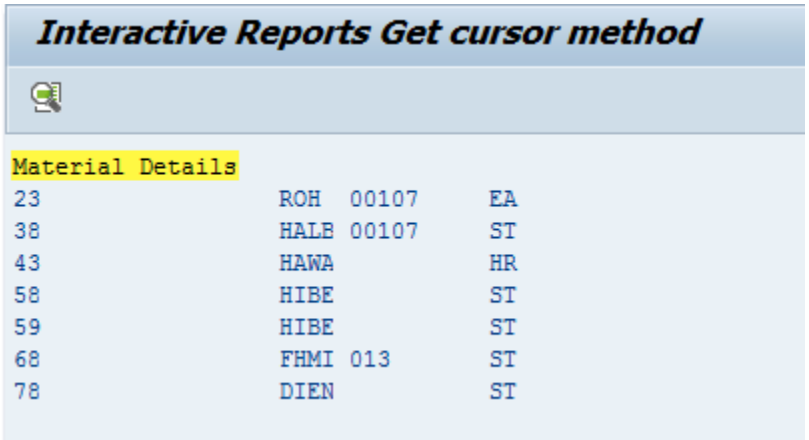
```
REPORT ZGTP_INTERACTIVE_GETCURSOR LINE-COUNT 30(3) NO STANDARD PAGE HEADING.

DATA: IT_MARA TYPE TABLE OF MARA,

      WA_MARA TYPE MARA,

      IT_MAKT TYPE TABLE OF MAKT,

      WA_MAKT TYPE MAKT.

DATA: FNAM (30), FVAL 50).

SELECT-OPTIONS: S_MATNR FOR WA_MARA-MATNR.

INITIALIZATION.
```

```
AT SELECTION-SCREEN.
  PERFORM VALIDATE_INPUT.
START-OF-SELECTION.
  PERFORM GET_DATA.
  PERFORM DISPLAY_DATA.
TOP-OF-PAGE.
  PERFORM DISPLAY_TOPOFPAGE.
AT LINE-SELECTION.
  PERFORM DISPLAY_SECONDARYLIST.
TOP-OF-PAGE DURING LINE-SELECTION.
  PERFORM LINE_TOPOFPAGE.
FORM VALIDATE_INPUT.
  IF S_MATNR IS INITIAL.
    MESSAGE 'Enter material input' TYPE 'E'.
  ENDIF.
ENDFORM.                    "VALIDATE_INPUT
FORM GET_DATA.
  SELECT * FROM MARA INTO TABLE IT_MARA WHERE MATNR IN S_MATNR.
ENDFORM.                    "GET_DATA
FORM DISPLAY_DATA.
  LOOP AT IT_MARA INTO WA_MARA.
    WRITE: / WA_MARA-MATNR, WA_MARA-MTART, WA_MARA-MATKL, WA_MARA-MEINS.
  ENDLOOP.
ENDFORM.                    "DISPLAY_DATA
FORM DISPLAY_SECONDARYLIST.
  GET CURSOR FIELD FNAM VALUE FVAL.
  CONDENSE FNAM.
  CONDENSE FVAL.
  IF FNAM = 'WA_MARA-MATNR'.
    SELECT SINGLE * FROM MARA INTO WA_MARA WHERE MATNR = FVAL .
    WRITE: / WA_MARA-MATNR, WA_MARA-MBRSH, WA_MARA-MTART, WA_MARA-MATKL, WA_M
ARA-MEINS, WA_MARA-ERSDA, WA_MARA-ERNAM.
  ELSEIF FNAM = 'WA_MARA-MTART'.
```

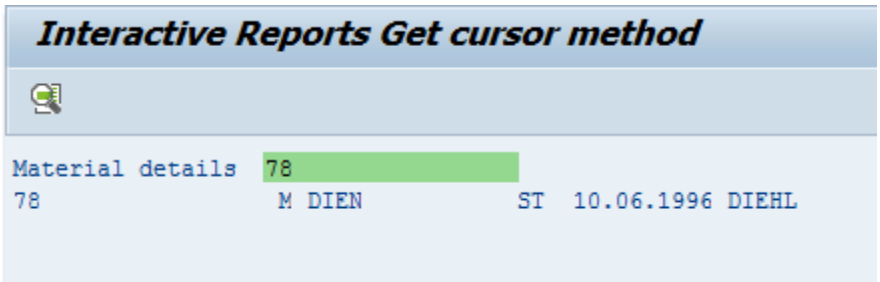
```
    SELECT * FROM MARA INTO TABLE IT_MARA UP TO 50 ROWS WHERE MTART = FVAL.
    LOOP AT IT_MARA INTO WA_MARA.
      WRITE: / WA_MARA-MATNR, WA_MARA-MTART, WA_MARA-MATKL.
    ENDLOOP.
  ENDIF.

ENDFORM.                    "DISPLAY_SECONDARYLIST
FORM DISPLAY_TOPOFPAGE.
  WRITE: / 'Material Details' COLOR 3.
ENDFORM.                    "DISPLAY_TOPOFPAGE
FORM LINE_TOPOFPAGE.
  IF FNAM = 'WA_MARA-MATNR'.
    WRITE: / 'Material details ', WA_MARA-MATNR COLOR 5.
  ELSEIF FNAM = 'WA_MARA-MTART'.
    WRITE: / 'Material with material type ', WA_MARA-MTART COLOR 5.
  ENDIF.
ENDFORM.                    "LINE_TOPOFPAGE
```

Output:

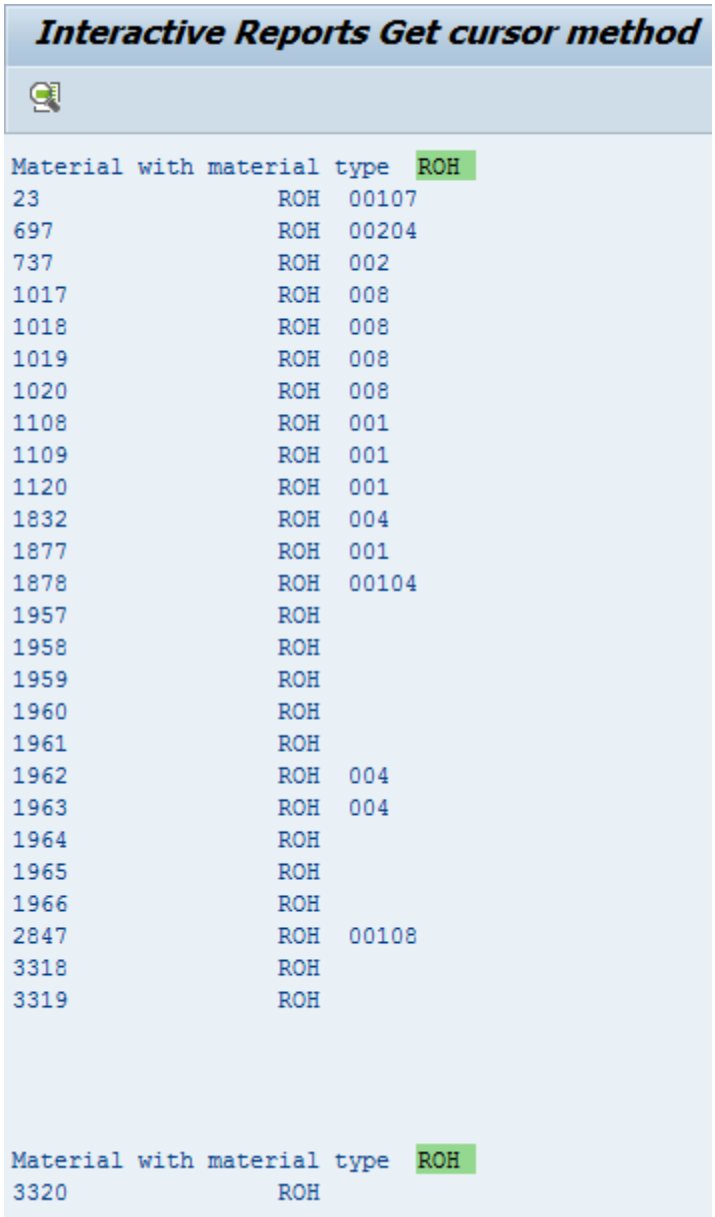




When user clicks on Material number (78) then the output will be:



When the user clicks on Material type Then the output will be:



TREE LIST INTERACTIVE REPORT:

As we already know SAP supports multiple types of drill down reports like ABAP Interactive reports, ALV Interactive etc., in this lesson we are going to develop a drill down report with TREE structure.

Program: Display list of materials for a give input(Material number ranges) with descriptions in different languages in the form of a tree.

To create a *TREE LIST*, we use RS_TREE_CONSTRUCT and RS_TREE_LIST_DISPLAY Function Modules

RS_TREE_CONSTRUCT is used to construct a TREE node.

RS_TREE_LIST_DISPLAY is used to display/print constructed TREE.

Steps to follow to create TREE LIST

1. Data Declarations for required tables
2. Get Data From Required Tables
3. Construct TREE node
4. Display TREE

Go to SE38, create a report with name *ZSAPN_TREE_MATERIAL* and follow below steps.

Data Declarations

Internal tables and work area declarations for required tables, in our requirement, we are building tree for MARA(Material) and MAKT(Material descriptions multiple languages).

```
***Material Basic Data Declarations
TYPES: BEGIN OF TY_MARA,
       MATNR TYPE MARA-MATNR,
       MTART TYPE MARA-MTART,
       MBRSH TYPE MARA-MBRSH,
       MATKL TYPE MARA-MATKL,
       MEINS TYPE MARA-MEINS,
     END OF TY_MARA.
DATA: IT_MARA TYPE TABLE OF TY_MARA.
DATA: WA_MARA TYPE TY_MARA.
***Material Descriptions Declarations
TYPES: BEGIN OF TY_MAKT,
       MATNR TYPE MAKT-MATNR,
       SPRAS TYPE MAKT-SPRAS,
       MAKTX TYPE MAKT-MAKTX,
     END OF TY_MAKT.
DATA: IT_MAKT TYPE TABLE OF TY_MAKT.
DATA : WA_MAKT TYPE TY_MAKT.
```

Get Data from required tables MARA and MAKT

Get data from tables MARA and MAKT under START-OF-SELECTION event using FOR ALL ENTRIES.

```
**Get Data From Tables
  SELECT MATNR MTART MBRSH MATKL MEINS FROM MARA INTO TABLE IT_MARA UP TO 50 RO
WS WHERE MATNR IN S_MATNR.
```

```
 SELECT MATNR SPRAS MAKTX FROM MAKT INTO TABLE IT_MAKT FOR ALL ENTRIES IN IT_MA
RA WHERE MATNR = IT_MARA-MATNR.
```

Construct TREE node suing RS_TREE_CONSTRUCT

Function module RS_TREE_CONSTRUCT has a table parameter with name NODETAB, this parameter holds the TLEVEL (Level of a row ex:1, 2 etc), NAME,TEXT,TEXT1,TEXT2 - TEXT9(field name ex: MATNR), NLENGTH,TLENGTH,TLENGTH1 - TLENGTH9(Length of the field ex: 18), COLOR, TCOLOR, TCOLOR1 to TCOLOR9 etc.

Note: By using RS_TREE_LIST_DISPLAY and RS_TREE_CONSTRUCT function modules, we can display maximum of 10 columns in each level.

```
***Data Table deceleration for FM RS_TREE_CONSTRUCT

DATA:  IT_NODE TYPE STANDARD TABLE OF SNODETEXT,

       WA_NODE TYPE SNODETEXT.

**Declare Constants for TREE

CONSTANTS:

      C_COL_KEY   TYPE C LENGTH 1 VALUE COL_KEY,

      C_COL_FIELD  TYPE C LENGTH 1 VALUE COL_NORMAL,

      C_COL_MATNR TYPE C LENGTH 1 VALUE COL_KEY,

      C_COL_MAKTX TYPE C LENGTH 1 VALUE COL_POSITIVE.
```

Create Root level

```
 "Create root node at level 1

 WA_NODE-TLEVEL = 1.

 WA_NODE-NAME   = 'Materials'.

 WA_NODE-NLENGTH = 20.

 WA_NODE-COLOR = C_COL_KEY.

 WA_NODE-TEXT   = 'Material Master Report'.

 WA_NODE-TLENGTH = 50.

 APPEND WA_NODE TO IT_NODE.
```

Loop through IT_MARA and IT_MAKT(inside IT_MARA loop) and build level 2 and level 3

```
LOOP AT IT_MARA INTO WA_MARA.

    CLEAR WA_NODE.

    WA_NODE-TLEVEL = 2. "Node Level 2
```

```
"Material Number
WA_NODE-NAME   = WA_MARA-MATNR.
WA_NODE-NLENGTH = 18.
WA_NODE-COLOR = C_COL_MATNR.
"Material Type
WA_NODE-TEXT1   = WA_MARA-MTART.
WA_NODE-TLENGTH1 = 4.
WA_NODE-TCOLOR1 = C_COL_FIELD.
"Industry Sector
WA_NODE-TEXT2   = WA_MARA-MBRSH.
WA_NODE-TLENGTH2 = 1.
WA_NODE-TCOLOR2 = C_COL_FIELD.
"Material Group
WA_NODE-TEXT3   = WA_MARA-MATKL.
WA_NODE-TLENGTH3 = 4.
WA_NODE-TCOLOR3 = C_COL_FIELD.
"Unit of Measure
WA_NODE-TEXT4   = WA_MARA-MEINS.
WA_NODE-TLENGTH4 = 3.
WA_NODE-TCOLOR4 = C_COL_FIELD.
APPEND WA_NODE TO IT_NODE.
LOOP AT IT_MAKT INTO WA_MAKT WHERE MATNR = WA_MARA-MATNR.
  CLEAR WA_NODE.
  WA_NODE-TLEVEL = 3. "Node level 3
  "Material Number
  WA_NODE-NAME   = WA_MAKT-MATNR.
  WA_NODE-NLENGTH = 18.
  WA_NODE-COLOR = C_COL_MAKTX.
  "Language ISO Code
  WA_NODE-TEXT   = WA_MAKT-SPRAS.
  WA_NODE-TLENGTH = 2.
  WA_NODE-TCOLOR = C_COL_FIELD.
```

```
      "Material Description
      WA_NODE-TEXT1   = WA_MAKT-MAKTX.
      WA_NODE-TLENGTH1 = 40.
      WA_NODE-TCOLOR1 = C_COL_FIELD.
      APPEND WA_NODE TO IT_NODE.
    ENDLOOP.
  ENDLOOP.
```

Call Function module RS_TREE_CONSTRUCT

```
**Construct Tree
  CALL FUNCTION 'RS_TREE_CONSTRUCT'
*   EXPORTING
*     INSERT_ID               = '000000'
*     RELATIONSHIP            = ' '
*     LOG                     =
    TABLES
      NODETAB           = IT_NODE
    EXCEPTIONS
      TREE_FAILURE       = 1
      ID_NOT_FOUND       = 2
      WRONG_RELATIONSHIP = 3
      OTHERS             = 4.
  IF SY-SUBRC <> 0.
    WRITE 'Error in Tree Construction'.
  ENDIF.
```

Display TREE

Finally Display *TREE* by calling Function module RS_TREE_LIST_DISPLAY

```
***Display TREE
  CALL FUNCTION 'RS_TREE_LIST_DISPLAY'
    EXPORTING
      CALLBACK_PROGRAM = SY-REPID.
```

Final and Full program source code

```
*&---------------------------------------------------------------------*
REPORT ZGTP_TREE_MATERIAL.
***Material Basic Data Declarations
TYPES: BEGIN OF TY_MARA,
       MATNR TYPE MARA-MATNR,
       MTART TYPE MARA-MTART,
       MBRSH TYPE MARA-MBRSH,
       MATKL TYPE MARA-MATKL,
       MEINS TYPE MARA-MEINS,
     END OF TY_MARA.
DATA: IT_MARA TYPE TABLE OF TY_MARA.
DATA: WA_MARA TYPE TY_MARA.
***Material Descriptions Declarations
TYPES: BEGIN OF TY_MAKT,
       MATNR TYPE MAKT-MATNR,
       SPRAS TYPE MAKT-SPRAS,
       MAKTX TYPE MAKT-MAKTX,
     END OF TY_MAKT.
DATA: IT_MAKT TYPE TABLE OF TY_MAKT.
DATA: WA_MAKT TYPE TY_MAKT.
***Data Table declaration for FM RS_TREE_CONSTRUCT
DATA:  IT_NODE TYPE STANDARD TABLE OF SNODETEXT,
      WA_NODE TYPE SNODETEXT.
**Declare Constants for TREE
CONSTANTS:
     C_COL_KEY   TYPE C LENGTH 1 VALUE COL_KEY,
     C_COL_FIELD  TYPE C LENGTH 1 VALUE COL_NORMAL,
     C_COL_MATNR TYPE C LENGTH 1 VALUE COL_KEY,
     C_COL_MAKTX TYPE C LENGTH 1 VALUE COL_POSITIVE.
**Selection Screen
SELECT-OPTIONS : S_MATNR FOR WA_MARA-MATNR.
```

```
START-OF-SELECTION.

**Get Data From Tables

  SELECT MATNR MTART MBRSH MATKL MEINS FROM MARA INTO TABLE IT_MARA UP TO 50 RO
WS WHERE MATNR IN S_MATNR.

  SELECT MATNR SPRAS MAKTX FROM MAKT INTO TABLE IT_MAKT FOR ALL ENTRIES IN IT_MA
RA WHERE MATNR = IT_MARA-MATNR.

  "Create root node at level 1

  WA_NODE-TLEVEL = 1.

  WA_NODE-NAME   = 'Materials'.

  WA_NODE-NLENGTH = 20.

  WA_NODE-COLOR = C_COL_KEY.

  WA_NODE-TEXT   = 'Material Master Report'.

  WA_NODE-TLENGTH = 50.

  APPEND WA_NODE TO IT_NODE.

  LOOP AT IT_MARA INTO WA_MARA.

    CLEAR WA_NODE.

    WA_NODE-TLEVEL = 2. "Node Level 2

    "Material Number

    WA_NODE-NAME   = WA_MARA-MATNR.

    WA_NODE-NLENGTH = 18.

    WA_NODE-COLOR = C_COL_MATNR.

    "Material Type

    WA_NODE-TEXT1   = WA_MARA-MTART.

    WA_NODE-TLENGTH1 = 4.

    WA_NODE-TCOLOR1 = C_COL_FIELD.

    "Industry Sector

    WA_NODE-TEXT2   = WA_MARA-MBRSH.

    WA_NODE-TLENGTH2 = 1.

    WA_NODE-TCOLOR2 = C_COL_FIELD.

    "Material Group

    WA_NODE-TEXT3   = WA_MARA-MATKL.

    WA_NODE-TLENGTH3 = 4.

    WA_NODE-TCOLOR3 = C_COL_FIELD.
```

```
    "Unit of Measure
    WA_NODE-TEXT4   = WA_MARA-MEINS.
    WA_NODE-TLENGTH4 = 3.
    WA_NODE-TCOLOR4 = C_COL_FIELD.
    APPEND WA_NODE TO IT_NODE.
    LOOP AT IT_MAKT INTO WA_MAKT WHERE MATNR = WA_MARA-MATNR.
      CLEAR WA_NODE.
      WA_NODE-TLEVEL = 3. "Node level 3
      "Material Number
      WA_NODE-NAME   = WA_MAKT-MATNR.
      WA_NODE-NLENGTH = 18.
      WA_NODE-COLOR = C_COL_MAKTX.
      "Language ISO Code
      WA_NODE-TEXT   = WA_MAKT-SPRAS.
      WA_NODE-TLENGTH = 2.
      WA_NODE-TCOLOR = C_COL_FIELD.
      "Material Description
      WA_NODE-TEXT1   = WA_MAKT-MAKTX.
      WA_NODE-TLENGTH1 = 40.
      WA_NODE-TCOLOR1 = C_COL_FIELD.
      APPEND WA_NODE TO IT_NODE.
    ENDLOOP.
  ENDLOOP.
**Construct Tree
  CALL FUNCTION 'RS_TREE_CONSTRUCT'
*   EXPORTING
*     INSERT_ID               = '000000'
*     RELATIONSHIP            = ' '
*     LOG                     =
    TABLES
      NODETAB           = IT_NODE
    EXCEPTIONS
```

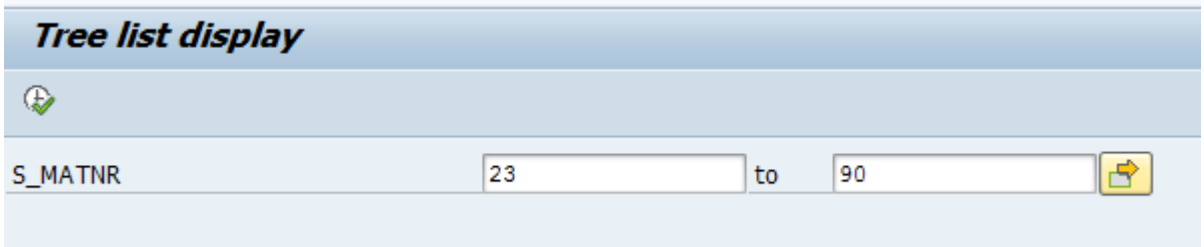
```
    TREE_FAILURE       = 1
    ID_NOT_FOUND       = 2
    WRONG_RELATIONSHIP = 3
    OTHERS             = 4.
  IF SY-SUBRC <> 0.
   WRITE 'Error in Tree Construction'.
  ENDIF.
***Display TREE
  CALL FUNCTION 'RS_TREE_LIST_DISPLAY'
   EXPORTING
    CALLBACK_PROGRAM = SY-REPID.
```

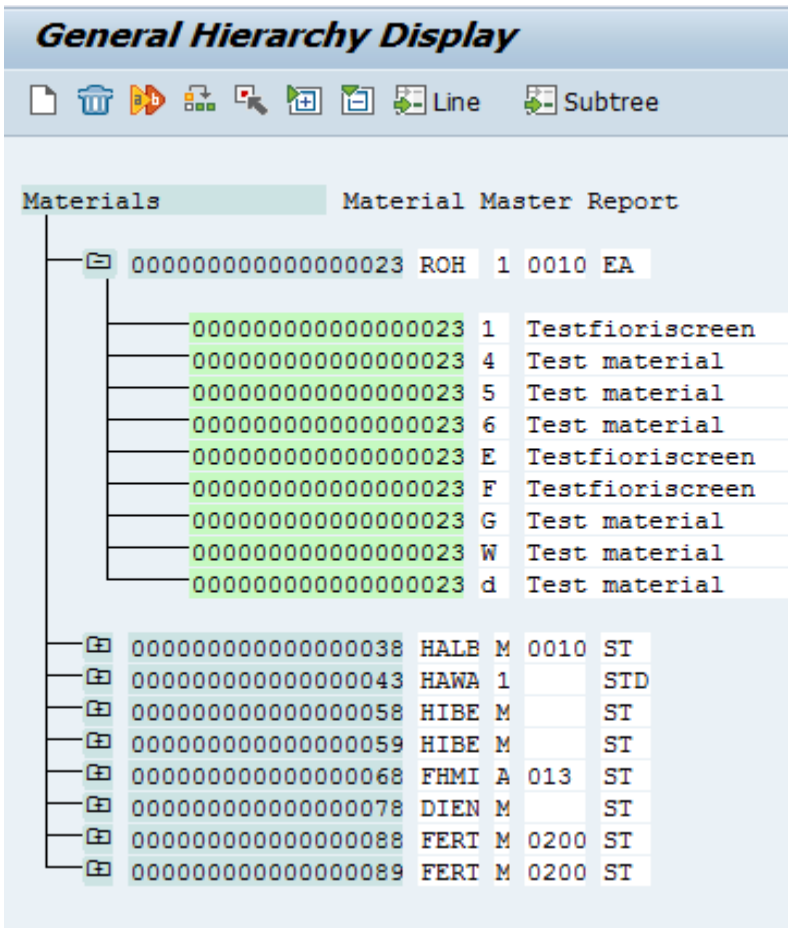
Note: By using RS_TREE_LIST_DISPLAY and RS_TREE_CONSTRUCT function modules, we can display maximum of 10 columns in each level.

Testing the above report

- Go to SE11 -> MAKT
- Get material numbers which have more the one description (different languages)
- Execute the program and provide the material numbers as inputs (which we got from MAKT)
- Execute, expand TREE and observe

Output:





3. ALV Reports

ALV (ABAP List Viewer) is pre-defined report format in SAP. Advantages of ALV Reports

- Better look and feel.
- ALV report consists of some per-defined options like sort, filters, sum, downloading, print, changing the layout structure and many more.

List of Function Modules used to develop ALV reports.
The below are the list of Function Modules we use to display ALV reports in different formats.

REUSE_ALV_GRID_DISPLAY.	"Display ALV grid format
REUSE_ALV_LIST_DISPLAY.	"Display ALV List format
REUSE_ALV_COMMENTARY_WRITE.	"Display Top of page, logo, etc.
REUSE_ALV_FIELDCATELOGUE_MERGE.	"Used to generate field catalogue
REUSE_ALV_EVENTS_GET.	"Use events in ALV
REUSE_ALV_HEIRARCHY_LIST_DISPLAY.	"Display ALV Hierarchy
REUSE_ALV_BLOCKED_LIST_DISPLAY.	"Display blocked list

Events In ALV:

Events in ALV Reports are:

1. SLIS_PRINT_ALV.
2. SLIS_T_LISTHEADER.
3. SLIS_T_EVENT.
4. SLIS_T_SORTINFO_ALV.
5. SLIS_T_LAYOUT_ALV.
6. SLIS_T_FIELDCAT_ALV.

ALV Report with Structure:

Requirement: Develop an ALV report to display material details with all fields from MARA table for a material range input (Select-Options input).

In this requirement, we have to display all fields from MARA (Material master table) in ALV format. We use REUSE_ALV_GRID_DISPLAY Function module to display an ALV report.

Steps to create ALV Report with Structure.

Step1: Declare Internal table for MARA table.

Step2: Print Select-Options.

Step3: Get data from database using select statements.

Step4: Call Function Module REUSE_ALV_GRID_DISPLAY and pass structure name, program name and internal table name.

ALV Report with Field Catalog:

Field Catalog

Field catalog is an internal table which is used to pass a list of fields to display in ALV report, we can set different properties to fields which are going to display in ALV.

Type Group

It is a data dictionary object which contains all the reusable user-defined types.

Example for a type group is SLIS, which contains all the user-defined types for developing ALV reports.

TYPE-POOLS is a keyword which is used to assign the type-group to a ALV report .

Syntax :

```
TYPE-POOLS: SLIS.          "TO USE FIELD CATALOG WE HAVE TO INCLUDE SLIS  TYPE- POOLS.
DATA :  <IT_FCAT> TYPE SLIS_T_FIELDCAT_ALV.  "INTERNAL TABLE FOR FIELD CATALOG.
DATA :  <WA_FCAT> TYPE SLIS_FIELDCAT_ALV.  " WORK AREA FOR FIELD CATLOG
```

Properties of field catalog.

```
WA_FCAT-COL_POS = '1'.                    "Specify position of a field
```

```
WA_FCAT-FIELDNAME = 'MATNR'.              "Specify field name
WA_FCAT-TABNAME = 'IT_MARA'.              "Specify internal table name
WA_FCAT-SELTEXT_M = 'MATERIALNO'.         "Specify text to display column header
WA_FCAT-KEY = 'X'.                        "Specify if it is a key field
APPEND WA_FCAT TO IT_FCAT.                "Append to field catalog internal table
```

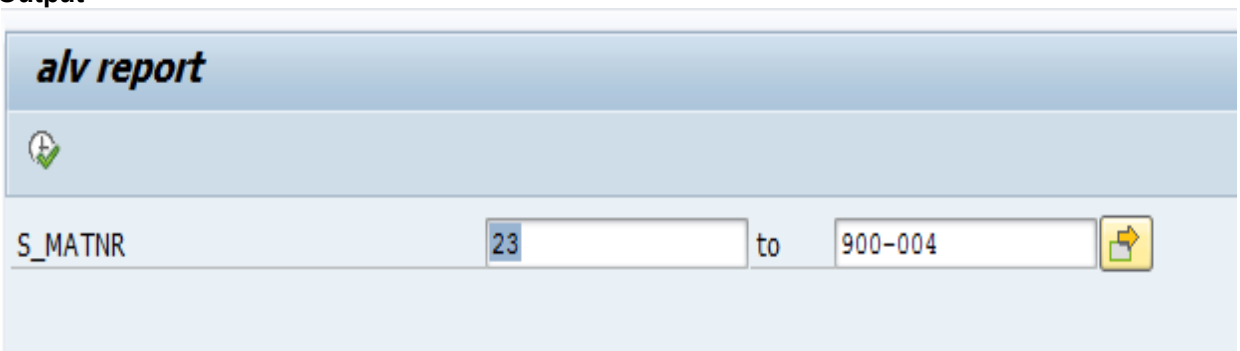
Program: Develop an ALV report to display Material no (MATNR), Material type (MTART), Industry Sector (MBRSH) and Basic Unit Of measure (MEINS) for a range of material input (Select-Options).
To develop above report, we have to use field catalog (because we have to display four fields only from MARA) and we have to pass field catalog parameter to Function Module REUSE_ALV_GIRD_DISPLAY.

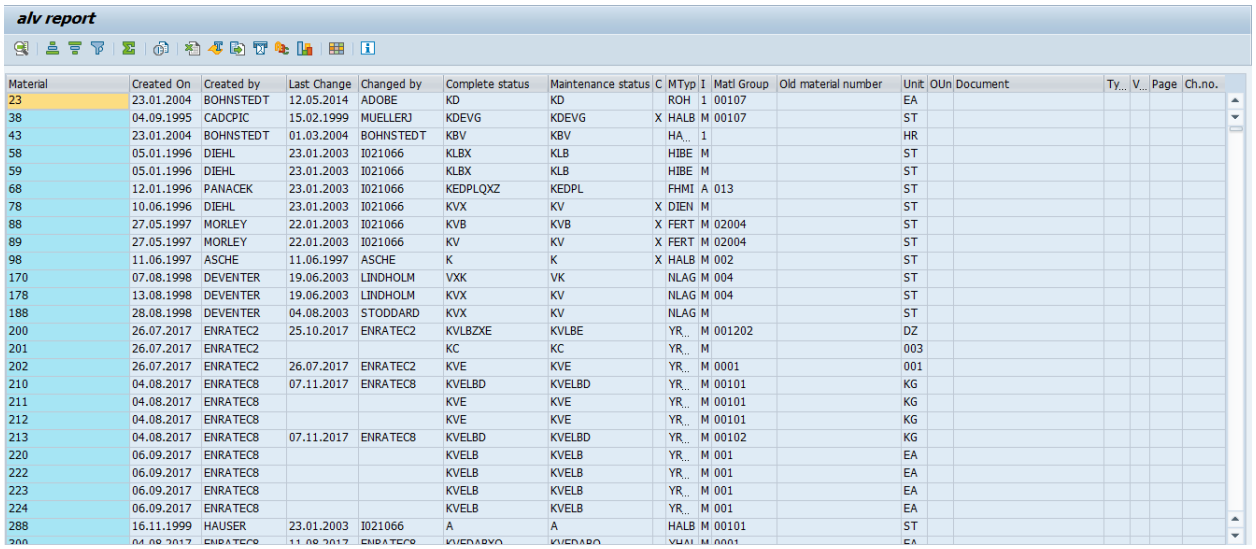
```
REPORT ZGTP_ALV_FCAT.
TABLES : MARA.
TYPE-POOLS SLIS .
TYPES : BEGIN OF TY_MARA,  "User defined internal table type
        MATNR TYPE MARA-MATNR,
        MTART TYPE MARA-MTART,
        MBRSH TYPE MARA-MBRSH,
        MEINS TYPE MARA-MEINS,
      END OF TY_MARA.
DATA : IT_MARA TYPE TABLE OF TY_MARA .
DATA : WA_MARA TYPE TY_MARA .
DATA : IT_FCAT TYPE SLIS_T_FIELDCAT_ALV .
DATA : WA_FCAT LIKE LINE OF IT_FCAT .
SELECT-OPTIONS: S_MATNR FOR MARA-MATNR.
START-OF-SELECTION.
  PERFORM GET_DATA.
  PERFORM CREATE_FCAT.
END-OF-SELECTION.
  PERFORM DISP_ALV.
*&---------------------------------------------------------------------*
*&      Form  GET_DATA
*&---------------------------------------------------------------------*
FORM GET_DATA .
```

```
  SELECT MATNR MTART MBRSH MEINS FROM MARA
     INTO TABLE IT_MARA
     WHERE MATNR IN S_MATNR.
ENDFORM.                    " GET_DATA
*&---------------------------------------------------------------------*
FORM DISP_ALV .
  CALL FUNCTION 'REUSE_ALV_GRID_DISPLAY'
    EXPORTING
      I_CALLBACK_PROGRAM = SY-REPID
      IT_FIELDCAT        = IT_FCAT "PASS FIELD CATALOG TO ALV
    TABLES
      T_OUTTAB           = IT_MARA.
ENDFORM.                    " DISP_ALV
*&---------------------------------------------------------------------*
*&      Form  CREATE_FCAT
*&---------------------------------------------------------------------*
FORM CREATE_FCAT .
  WA_FCAT-COL_POS = '1' .
  WA_FCAT-FIELDNAME = 'MATNR' .
  WA_FCAT-TABNAME = 'IT_MARA' .
  WA_FCAT-SELTEXT_M = 'MATERIALNO' .
  WA_FCAT-KEY = 'X' .
  APPEND WA_FCAT TO IT_FCAT .
  CLEAR WA_FCAT .
  WA_FCAT-COL_POS = '2' .
  WA_FCAT-FIELDNAME = 'MTART' .
  WA_FCAT-TABNAME = 'IT_MARA' .
  WA_FCAT-SELTEXT_M = 'MATERIALTYPE' .
*  WA_FCAT-NO_OUT = 'X' .
  WA_FCAT-HOTSPOT = 'X' .
  APPEND WA_FCAT TO IT_FCAT .
  CLEAR WA_FCAT .
```

```
  WA_FCAT-COL_POS = '3' .
  WA_FCAT-FIELDNAME = 'MBRSH' .
  WA_FCAT-REF_FIELDNAME = 'MBRSH' .
  WA_FCAT-REF_TABNAME = 'MARA' .
*  WA_FCAT-TABNAME = 'IT_MARA' .
*  WA_FCAT-SELTEXT_M = 'INDSECTOR' .
*  WA_FCAT-EDIT = 'X' .
  APPEND WA_FCAT TO IT_FCAT .
  CLEAR WA_FCAT .
  WA_FCAT-COL_POS = '4' .
  WA_FCAT-FIELDNAME = 'MEINS' .
  WA_FCAT-TABNAME = 'IT_MARA' .
  WA_FCAT-SELTEXT_M = 'MAT.UNITS' .
  WA_FCAT-EMPHASIZE = 'C610'.
  APPEND WA_FCAT TO IT_FCAT .
  CLEAR WA_FCAT .
ENDFORM.                    " CREATE_FCAT
```

Output





alv report

Material	Created On	Created by	Last Change	Changed by	Complete status	Maintenance status	C	MTyp	I	Matl Group	Old material number	Unit	OUn	Document	Ty...	V...	Page	Ch.no.
23	23.01.2004	BOHNSTEDT	12.05.2014	ADOBE	KD	KD		ROH	1	00107		EA						
38	04.09.1995	CADCPIC	15.02.1999	MUELLERJ	KDEVG	KDEVG	X	HALB	M	00107		ST						
43	23.01.2004	BOHNSTEDT	01.03.2004	BOHNSTEDT	KBV	KBV		HA...	1			HR						
58	05.01.1996	DIEHL	23.01.2003	I021066	KLBX	KLB		HIBE	M			ST						
59	05.01.1996	DIEHL	23.01.2003	I021066	KLBX	KLB		HIBE	M			ST						
68	12.01.1996	PANACEK	23.01.2003	I021066	KEDPLQXZ	KEDPL		FHMI	A	013		ST						
78	10.06.1996	DIEHL	23.01.2003	I021066	KVX	KV	X	DIEN	M			ST						
88	27.05.1997	MORLEY	22.01.2003	I021066	KVB	KVB	X	FERT	M	02004		ST						
89	27.05.1997	MORLEY	22.01.2003	I021066	KV	KV	X	FERT	M	02004		ST						
98	11.06.1997	ASCHE	11.06.1997	ASCHE	K	K	X	HALB	M	002		ST						
170	07.08.1998	DEVENTER	19.06.2003	LINDHOLM	VXK	VK		NLAG	M	004		ST						
178	13.08.1998	DEVENTER	19.06.2003	LINDHOLM	KVX	KV		NLAG	M	004		ST						
188	28.08.1998	DEVENTER	04.08.2003	STODDARD	KVX	KV		NLAG	M			ST						
200	26.07.2017	ENRATEC2	25.10.2017	ENRATEC2	KVLBZXE	KVLBE		YR...	M	001202		DZ						
201	26.07.2017	ENRATEC2			KC	KC		YR...	M			003						
202	26.07.2017	ENRATEC2	26.07.2017	ENRATEC2	KVE	KVE		YR...	M	0001		001						
210	04.08.2017	ENRATEC8	07.11.2017	ENRATEC8	KVELBD	KVELBD		YR...	M	00101		KG						
211	04.08.2017	ENRATEC8			KVE	KVE		YR...	M	00101		KG						
212	04.08.2017	ENRATEC8			KVE	KVE		YR...	M	00101		KG						
213	04.08.2017	ENRATEC8	07.11.2017	ENRATEC8	KVELBD	KVELBD		YR...	M	00102		KG						
220	06.09.2017	ENRATEC8			KVELB	KVELB		YR...	M	001		EA						
222	06.09.2017	ENRATEC8			KVELB	KVELB		YR...	M	001		EA						
223	06.09.2017	ENRATEC8			KVELB	KVELB		YR...	M	001		EA						
224	06.09.2017	ENRATEC8			KVELB	KVELB		YR...	M	001		EA						
288	16.11.1999	HAUSER	23.01.2003	I021066	A	A		HALB	M	00101		ST						
300	04.08.2017	ENRATEC8	11.08.2017	ENRATEC8	KVEDABXO	KVEDABXO		YHAL	M	0001		EA						

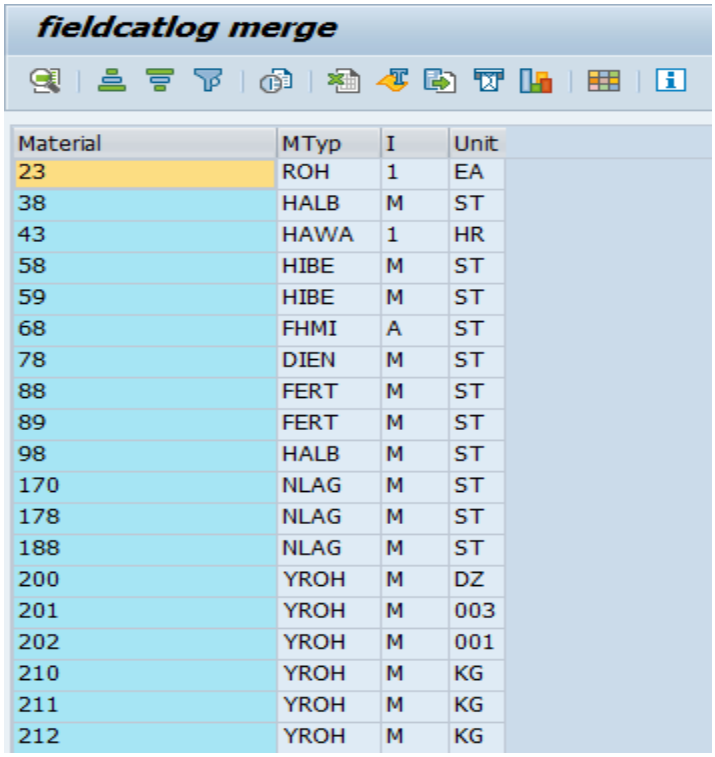
Field Catalog Merge:

Field Catalog can be generated in two ways, one is manual with field catalog internal table and another one is with automatic field catalog generation...like below one, but this one uses old syntax, this one is obsolete and is not recommended.

```
*&---------------------------------------------------------------------*
REPORT  ZGTP_ALV_WITH_FCAT_MERGE.
TYPE-POOLS SLIS .
DATA : BEGIN OF I_MARA OCCURS 0,
       MATNR LIKE MARA-MATNR,
       MTART LIKE MARA-MTART,
       MBRSH LIKE MARA-MBRSH,
       MEINS LIKE MARA-MEINS,
     END OF I_MARA.
DATA : I_FCAT TYPE SLIS_T_FIELDCAT_ALV .
DATA : WA_FCAT LIKE LINE OF I_FCAT .
START-OF-SELECTION .
  PERFORM GET_DATA .
  PERFORM CREATE_FCAT_MERGE .
END-OF-SELECTION .
  PERFORM DISP_ALV .
*&---------------------------------------------------------------------*
*&      Form  GET_DATA
```

```
*&---------------------------------------------------------------------*
FORM GET_DATA .
  SELECT * FROM MARA
     INTO CORRESPONDING FIELDS OF TABLE I_MARA.
ENDFORM.                    " GET_DATA
*&---------------------------------------------------------------------*
*&      Form  DISP_ALV
*&---------------------------------------------------------------------*
FORM DISP_ALV .
  CALL FUNCTION 'REUSE_ALV_GRID_DISPLAY'
    EXPORTING
      I_CALLBACK_PROGRAM = SY-REPID
      IT_FIELDCAT        = I_FCAT
    TABLES
      T_OUTTAB           = I_MARA.
ENDFORM.                    " DISP_ALV
*&---------------------------------------------------------------------*
*&      Form  CREATE_FCAT_MERGE
*&---------------------------------------------------------------------*
FORM CREATE_FCAT_MERGE.
  CALL FUNCTION 'REUSE_ALV_FIELDCATALOG_MERGE'
    EXPORTING
      I_PROGRAM_NAME     = SY-REPID
      I_INTERNAL_TABNAME = 'I_MARA'
      I_INCLNAME         = SY-REPID
    CHANGING
      CT_FIELDCAT        = I_FCAT.
ENDFORM.                    " CREATE_FCAT_MERGE
```

Output:



fieldcatlog merge

Material	MTyp	I	Unit
23	ROH	1	EA
38	HALB	M	ST
43	HAWA	1	HR
58	HIBE	M	ST
59	HIBE	M	ST
68	FHMI	A	ST
78	DIEN	M	ST
88	FERT	M	ST
89	FERT	M	ST
98	HALB	M	ST
170	NLAG	M	ST
178	NLAG	M	ST
188	NLAG	M	ST
200	YROH	M	DZ
201	YROH	M	003
202	YROH	M	001
210	YROH	M	KG
211	YROH	M	KG
212	YROH	M	KG

ALV Report with layout:

Layout:

Layout is a structure which is used to decorate or embellish the output of ALV Report.

Program: Display list of materials for a material type, with all fields of output is editable, no horizontal and vertical lines and hotspot on material no.

Please follow previous lesson steps to create a fieldcatlog in ALV, in addition to that we will be adding additional properties hotspot, editable to output fields using layout options.

The below is the code to add layout to ALV.

```
DATA : WA_LAYOUT TYPE SLIS_LAYOUT_ALV .
  WA_LAYOUT-ZEBRA = 'X'.
  WA_LAYOUT-COLWIDTH_OPTIMIZE = 'X'.
  WA_LAYOUT-EDIT = 'X'.
  WA_LAYOUT-NO_VLINE = 'X'.
  WA_LAYOUT-NO_HLINE = 'X'.
```

We will supply remaining options through fieldcatlog also.

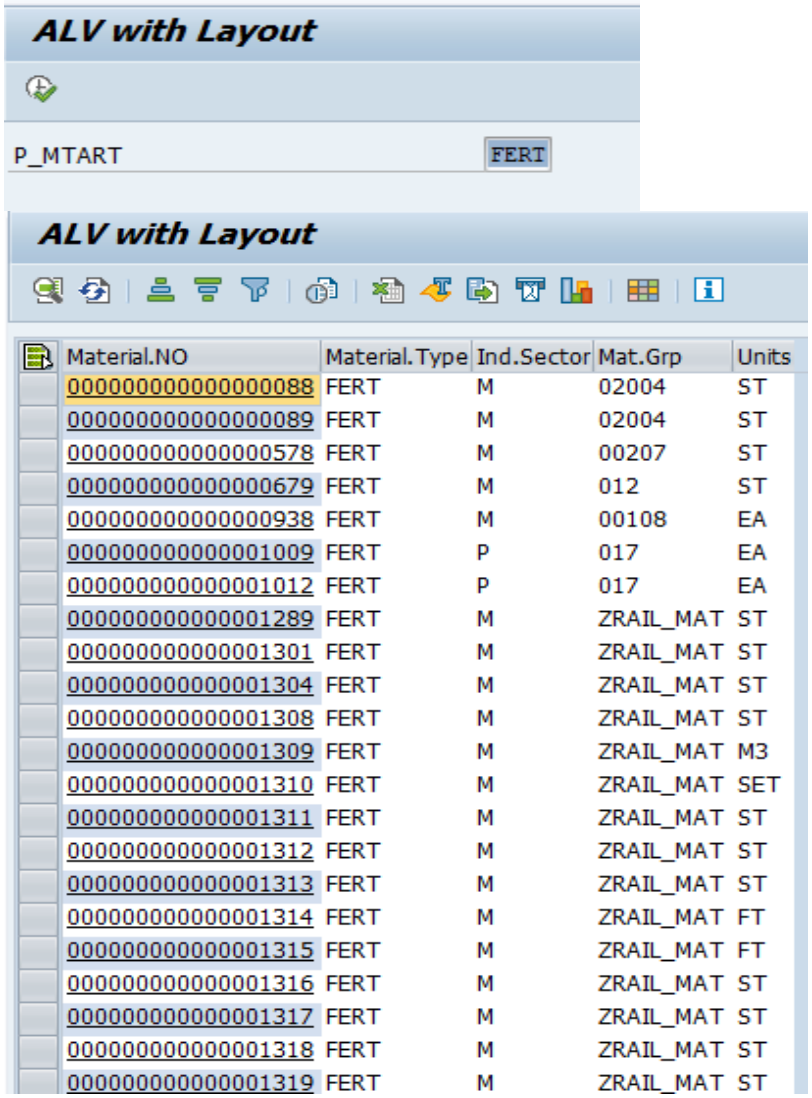
Full referance code for using layout in ALV

```
REPORT  ZGTP_ALV_LAYOUT.
TYPE-POOLS: SLIS.
*DATA DECLARAATIONS
DATA: I_MARA TYPE TABLE OF MARA .
DATA: WA_MARA TYPE MARA .
*ALV DECLARATIONS
DATA : I_FCAT TYPE SLIS_T_FIELDCAT_ALV .
DATA : WA_FCAT TYPE SLIS_FIELDCAT_ALV .
DATA : V_POS TYPE I .
DATA : WA_LAYOUT TYPE SLIS_LAYOUT_ALV .
PARAMETERS : P_MTART TYPE MARA-MTART.

START-OF-SELECTION .
  PERFORM GET_DATA .
  PERFORM CREATE_FCAT .
  PERFORM CREATE_LAYOUT.
  PERFORM DISPLAY_DATA .
END-OF-SELECTION .
FORM GET_DATA .
  SELECT * FROM MARA
      INTO TABLE I_MARA
       UP TO 100 ROWS WHERE MTART = P_MTART .
ENDFORM.                    " GET_DATA
FORM DISPLAY_DATA .
  CALL FUNCTION 'REUSE_ALV_GRID_DISPLAY'
    EXPORTING
      I_CALLBACK_PROGRAM = SY-REPID
      IS_LAYOUT          = WA_LAYOUT
      IT_FIELDCAT        = I_FCAT
    TABLES
      T_OUTTAB           = I_MARA.
ENDFORM.                    " DISPLAY_DATA
FORM DISPLAY_DATA_LIST.
```

```
  CALL FUNCTION 'REUSE_ALV_LIST_DISPLAY'
    EXPORTING
      I_STRUCTURE_NAME = 'MARA'
    TABLES
      T_OUTTAB         = I_MARA.
  IF SY-SUBRC NE 0.
* MESSAGE ID SY-MSGID TYPE SY-MSGTY NUMBER SY-MSGNO
*        WITH SY-MSGV1 SY-MSGV2 SY-MSGV3 SY-MSGV4.
  ENDIF.
ENDFORM.                    " DISPLAY_DATA_LIST
FORM CREATE_FCAT.
  V_POS = 1.
  WA_FCAT-COL_POS = V_POS.
  WA_FCAT-FIELDNAME = 'MATNR' .
  WA_FCAT-SELTEXT_M = 'Material.NO' .
  WA_FCAT-HOTSPOT = 'X'.
  APPEND WA_FCAT TO I_FCAT.
  CLEAR WA_FCAT.

  V_POS = V_POS + 1.
  WA_FCAT-COL_POS = V_POS.
  WA_FCAT-FIELDNAME = 'MTART'.
  WA_FCAT-SELTEXT_M = 'Material Type.
  APPEND WA_FCAT TO I_FCAT.
  CLEAR WA_FCAT.

  V_POS = V_POS + 1.
  WA_FCAT-COL_POS = V_POS.
  WA_FCAT-FIELDNAME = 'MBRSH'.
  WA_FCAT-SELTEXT_M = 'Ind.Sector'.
  APPEND WA_FCAT TO I_FCAT.
  CLEAR WA_FCAT.

  V_POS = V_POS + 1.
```

```
  WA_FCAT-COL_POS = V_POS .
  WA_FCAT-FIELDNAME = 'MATKL' .
  WA_FCAT-SELTEXT_M = 'Mat.Grp' .
  APPEND WA_FCAT TO I_FCAT .
  CLEAR WA_FCAT .



  V_POS = V_POS   1.
  WA_FCAT-COL_POS = V_POS .
  WA_FCAT-FIELDNAME = 'MEINS' .
  WA_FCAT-SELTEXT_M = 'Units' .
  APPEND WA_FCAT TO I_FCAT .
  CLEAR WA_FCAT .

ENDFORM.                    " CREATE_FCAT
FORM CREATE_LAYOUT .
  WA_LAYOUT-ZEBRA = 'X' .
  WA_LAYOUT-COLWIDTH_OPTIMIZE = 'X' .
  WA_LAYOUT-EDIT = 'X' .
  WA_LAYOUT-NO_VLINE = 'X' .
  WA_LAYOUT-NO_HLINE = 'X' .

ENDFORM.                    " CREATE_LAYOUT
```

Output: Screen (all rows are editable and hotspot on material no)



Material.NO	Material.Type	Ind.Sector	Mat.Grp	Units
000000000000000088	FERT	M	02004	ST
000000000000000089	FERT	M	02004	ST
000000000000000578	FERT	M	00207	ST
000000000000000679	FERT	M	012	ST
000000000000000938	FERT	M	00108	EA
000000000000001009	FERT	P	017	EA
000000000000001012	FERT	P	017	EA
000000000000001289	FERT	M	ZRAIL_MAT	ST
000000000000001301	FERT	M	ZRAIL_MAT	ST
000000000000001304	FERT	M	ZRAIL_MAT	ST
000000000000001308	FERT	M	ZRAIL_MAT	ST
000000000000001309	FERT	M	ZRAIL_MAT	M3
000000000000001310	FERT	M	ZRAIL_MAT	SET
000000000000001311	FERT	M	ZRAIL_MAT	ST
000000000000001312	FERT	M	ZRAIL_MAT	ST
000000000000001313	FERT	M	ZRAIL_MAT	ST
000000000000001314	FERT	M	ZRAIL_MAT	FT
000000000000001315	FERT	M	ZRAIL_MAT	FT
000000000000001316	FERT	M	ZRAIL_MAT	ST
000000000000001317	FERT	M	ZRAIL_MAT	ST
000000000000001318	FERT	M	ZRAIL_MAT	ST
000000000000001319	FERT	M	ZRAIL_MAT	ST

ALV with totals and sub totals:

To calculate totals and sub-totals in ALV we need to sort the internal table in ascending order and we need to set SUBTOT = 'X' of SORT in ALV.

Program: Display list of sales order for sales order range with totals and subtotals of price

Please follow previous lesson steps to create a fieldcatlog in ALV, in addition to that we will be adding additional properties hotspot, editable to output fields using layout options.

The below code is used to display totals and subtotals in ALV.

```
DATA : I_SORT TYPE SLIS_T_SORTINFO_ALV .

DATA : WA_SORT LIKE LINE OF I_SORT .

WA_SORT-FIELDNAME = 'VBELN '.

WA_SORT-UP = 'X'.

WA_SORT-SUBTOT = 'X '.
```

```
APPEND WA_SORT TO I_SORT .
```

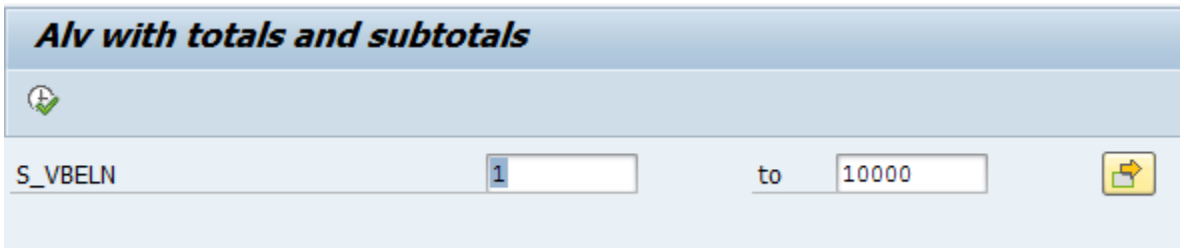
Full reference code for displaying totals and subtotals in ALV

```
REPORT  ZALV_WITH_TOTALS_SUBTOT.
TYPE-POOLS SLIS .
tables : vbap.
TYPES : BEGIN OF TY_VBAP,
        VBELN TYPE VBAP-VBELN,
        POSNR TYPE VBAP-POSNR,
        MATNR TYPE VBAP-MATNR,
        NETWR TYPE VBAP-NETWR,
      END OF TY_VBAP.
DATA :  I_VBAP TYPE TABLE OF TY_VBAP,
             WA_VBAP TYPE TY_VBAP,
             I_FCAT TYPE SLIS_T_FIELDCAT_ALV,
             WA_FCAT LIKE LINE OF I_FCAT,
             I_SORT TYPE SLIS_T_SORTINFO_ALV,
             WA_SORT LIKE LINE OF I_SORT.
select-options : s_vbeln for vbap-vbeln.
START-OF-SELECTION .
  PERFORM GET_DATA .
  PERFORM CREATE_FCAT.
  PERFORM CALC_SUBTOT.

END-OF-SELECTION .
  PERFORM DISP_ALV .
FORM GET_DATA .
  SELECT VBELN POSNR MATNR NETWR FROM VBAP
      INTO TABLE I_VBAP where vbeln in s_vbeln.
ENDFORM.                    " GET_DATA
FORM DISP_ALV .
  CALL FUNCTION 'REUSE_ALV_GRID_DISPLAY'
   EXPORTING
     I_CALLBACK_PROGRAM                = SY-REPID
```

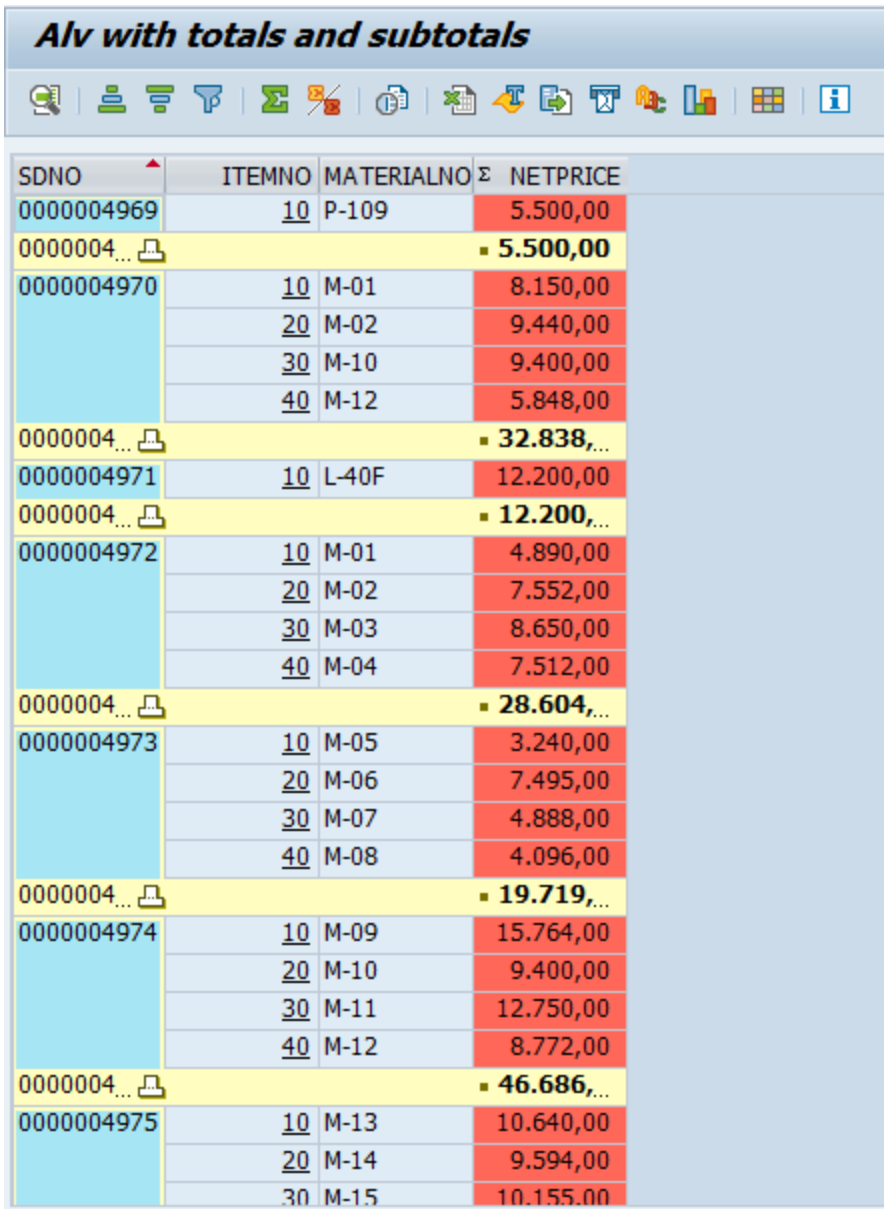
```
    IT_FIELDCAT                    = I_FCAT
    IT_SORT                        = I_SORT
   TABLES
     T_OUTTAB                      = I_VBAP          .
  IF SY-SUBRC NE 0.
* MESSAGE ID SY-MSGID TYPE SY-MSGTY NUMBER SY-MSGNO
*        WITH SY-MSGV1 SY-MSGV2 SY-MSGV3 SY-MSGV4.
  ENDIF.
ENDFORM.                    " DISP_ALV
FORM CREATE_FCAT .
  WA_FCAT-COL_POS = '1' .
  WA_FCAT-FIELDNAME = 'VBELN' .
  WA_FCAT-TABNAME = 'I_VBAP' .
  WA_FCAT-SELTEXT_M = 'SDNO' .
  WA_FCAT-KEY = 'X' .
  APPEND WA_FCAT TO I_FCAT .
  CLEAR WA_FCAT .

  WA_FCAT-COL_POS = '2' .
  WA_FCAT-FIELDNAME = 'POSNR' .
  WA_FCAT-TABNAME = 'I_VBAP' .
  WA_FCAT-SELTEXT_M = 'ITEMNO' .
*  WA_FCAT-NO_OUT = 'X' .
  WA_FCAT-HOTSPOT = 'X' .
  APPEND WA_FCAT TO I_FCAT .
  CLEAR WA_FCAT .

  WA_FCAT-COL_POS = '3' .
  WA_FCAT-FIELDNAME = 'MATNR' .
  WA_FCAT-TABNAME = 'I_VBAP' .
  WA_FCAT-SELTEXT_M = 'MATERIALNO' .
*  WA_FCAT-EDIT = 'X' .
  APPEND WA_FCAT TO I_FCAT .
  CLEAR WA_FCAT .
```

```
  WA_FCAT-COL_POS = '4' .
  WA_FCAT-FIELDNAME = 'NETWR' .
  WA_FCAT-TABNAME = 'I_VBAP' .
  WA_FCAT-SELTEXT_M = 'NETPRICE' .
  WA_FCAT-EMPHASIZE = 'C610'.
  WA_FCAT-DO_SUM = 'X' .
  APPEND WA_FCAT TO I_FCAT .
  CLEAR WA_FCAT .

ENDFORM.                    " CREATE_FCAT
FORM CALC_SUBTOT .
  WA_SORT-FIELDNAME = 'VBELN '.
  WA_SORT-UP = 'X'.
  WA_SORT-SUBTOT = 'X '.
  APPEND WA_SORT TO I_SORT .
ENDFORM.                    " CALC_SUBTOT
```

Input screen



Output:



Alv with totals and subtotals

SDNO	ITEMNO	MATERIALNO	Σ NETPRICE
0000004969	10	P-109	5.500,00
0000004...			5.500,00
0000004970	10	M-01	8.150,00
	20	M-02	9.440,00
	30	M-10	9.400,00
	40	M-12	5.848,00
0000004...			32.838,...
0000004971	10	L-40F	12.200,00
0000004...			12.200,...
0000004972	10	M-01	4.890,00
	20	M-02	7.552,00
	30	M-03	8.650,00
	40	M-04	7.512,00
0000004...			28.604,...
0000004973	10	M-05	3.240,00
	20	M-06	7.495,00
	30	M-07	4.888,00
	40	M-08	4.096,00
0000004...			19.719,...
0000004974	10	M-09	15.764,00
	20	M-10	9.400,00
	30	M-11	12.750,00
	40	M-12	8.772,00
0000004...			46.686,...
0000004975	10	M-13	10.640,00
	20	M-14	9.594,00
	30	M-15	10.155,00

Blocked List In ALV:

Blocked list ALV is used to display multiple ALV`s on the same screen with blocks.

List of Function Modules used for blocked list ALV:

REUSE_ALV_BLOCK_LIST_INIT: is used to initialize blocked list ALV.

REUSE_ALV_BLOCK_LIST_APPEND: is used to add blocked list ALV's(we can add multiple).

REUSE_ALV_BLOCK_LIST_DISPLAY: is used to display blocked list ALV.

Code to display blocked list ALV:

```
REPORT ZGTP_ALV_BLOCKEDLIST.
TYPES: BEGIN OF TY_MARA, "user defined type for mara
       MATNR TYPE MARA-MATNR,
       MTART TYPE MARA-MTART,
       MBRSH TYPE MARA-MBRSH,
       MATKL TYPE MARA-MATKL,
       MEINS TYPE MARA-MEINS,
      END OF TY_MARA.
DATA : IT_MARA TYPE TABLE OF TY_MARA, "mara internal table
      WA_MARA TYPE TY_MARA. "mara work area
DATA : T_FCAT TYPE SLIS_T_FIELDCAT_ALV. "field catalog for MARA table
DATA : W_FCAT LIKE LINE OF T_FCAT.

TYPES: BEGIN OF TY_MAKT, "user defined type for MAKT
       MATNR TYPE MAKT-MATNR,
       SPRAS TYPE MAKT-SPRAS,
       MAKTX TYPE MAKT-MAKTX,
      END OF TY_MAKT.
DATA : IT_MAKT TYPE TABLE OF TY_MAKT, "makt internal table
      WA_MAKT TYPE TY_MAKT.
DATA : T_FCAT_MAKT TYPE SLIS_T_FIELDCAT_ALV. "makt field catalog
DATA : W_FCAT_MAKT LIKE LINE OF T_FCAT_MAKT.
PARAMETERS : P_MTART TYPE MARA-MTART. "material type input

START-OF-SELECTION.
  SELECT MATNR
         MTART
         MBRSH
         MATKL
         MEINS FROM MARA "get MARA data
    INTO TABLE IT_MARA UP TO 10 ROWS WHERE MTART = P_MTART.
  IF NOT IT_MARA IS INITIAL .
    SELECT MATNR
           SPRAS
```

```
        MAKTX FROM MAKT INTO TABLE IT_MAKT "get makt data
    FOR ALL ENTRIES IN IT_MARA WHERE MATNR = IT_MARA-MATNR.
  ENDIF.
***build fcat for MARA
  W_FCAT-COL_POS       = '1'. "coloum position
  W_FCAT-FIELDNAME     = 'MATNR'. "column name
  W_FCAT-TABNAME       = 'IT_MARA'. "table
  W_FCAT-REF_TABNAME   = 'MARA'. "table
  W_FCAT-REF_FIELDNAME = 'MATNR'. "referance field, it will show descriptions automatically
  APPEND W_FCAT TO T_FCAT.
  CLEAR W_FCAT.

  W_FCAT-COL_POS       = '2'.
  W_FCAT-FIELDNAME     = 'MTART'.
  W_FCAT-TABNAME       = 'IT_MARA'.
  W_FCAT-REF_TABNAME   = 'MARA'.
  W_FCAT-REF_FIELDNAME = 'MTART'.
  APPEND W_FCAT TO T_FCAT.
  CLEAR W_FCAT.

  W_FCAT-COL_POS       = '3'.
  W_FCAT-FIELDNAME     = 'MBRSH'.
  W_FCAT-TABNAME       = 'IT_MARA'.
  W_FCAT-REF_TABNAME   = 'MARA'.
  W_FCAT-REF_FIELDNAME = 'MBRSH'.
  APPEND W_FCAT TO T_FCAT.
  CLEAR W_FCAT.

  W_FCAT-COL_POS       = '4'.
  W_FCAT-FIELDNAME     = 'MATKL'.
  W_FCAT-TABNAME       = 'IT_MARA'.
  W_FCAT-REF_TABNAME   = 'MARA'.
  W_FCAT-REF_FIELDNAME = 'MATKL'.
  APPEND W_FCAT TO T_FCAT.
```

```
  CLEAR W_FCAT.

  W_FCAT-COL_POS       = '5'.
  W_FCAT-FIELDNAME     = 'MEINS'.
  W_FCAT-TABNAME       = 'IT_MARA'.
  W_FCAT-REF_TABNAME   = 'MARA'.
  W_FCAT-REF_FIELDNAME = 'MEINS'.
  APPEND W_FCAT TO T_FCAT.
  CLEAR W_FCAT.

***build fcat for MAKT
  W_FCAT_MAKT-COL_POS       = '1'.
  W_FCAT_MAKT-FIELDNAME     = 'MATNR'.
  W_FCAT_MAKT-TABNAME       = 'IT_MAKT'.
  W_FCAT_MAKT-REF_TABNAME   = 'MAKT'.
  W_FCAT_MAKT-REF_FIELDNAME = 'MATNR'.
  APPEND W_FCAT_MAKT TO T_FCAT_MAKT.
  CLEAR W_FCAT_MAKT.
  W_FCAT_MAKT-COL_POS       = '2'.
  W_FCAT_MAKT-FIELDNAME     = 'SPRAS'.
  W_FCAT_MAKT-TABNAME       = 'IT_MAKT'.
  W_FCAT_MAKT-REF_TABNAME   = 'MAKT'.
  W_FCAT_MAKT-REF_FIELDNAME = 'SPRAS'.
  APPEND W_FCAT_MAKT TO T_FCAT_MAKT.
  CLEAR W_FCAT_MAKT.
  W_FCAT_MAKT-COL_POS       = '3'.
  W_FCAT_MAKT-FIELDNAME     = 'MAKTX'.
  W_FCAT_MAKT-TABNAME       = 'IT_MAKT'.
  W_FCAT_MAKT-REF_TABNAME   = 'MAKT'.
  W_FCAT_MAKT-REF_FIELDNAME = 'MAKTX'.
  APPEND W_FCAT_MAKT TO T_FCAT_MAKT.
  CLEAR W_FCAT_MAKT.
*  * init
  CALL FUNCTION 'REUSE_ALV_BLOCK_LIST_INIT' "initialize Block List ALV
```

```
    EXPORTING
      I_CALLBACK_PROGRAM = SY-REPID.

  DATA S_EVENTS TYPE SLIS_T_EVENT.
  DATA S_LAYOUT TYPE SLIS_LAYOUT_ALV.
  S_LAYOUT-ZEBRA = 'X'.
  CALL FUNCTION 'REUSE_ALV_BLOCK_LIST_APPEND' "append ALV lists
    EXPORTING
      IS_LAYOUT   = S_LAYOUT "set layout
      IT_FIELDCAT = T_FCAT "set field catalog
      I_TABNAME   = 'IT_MARA' "table
      IT_EVENTS   = S_EVENTS "events
*     IT_SORT     =
*     I_TEXT      = ' '
    TABLES
      T_OUTTAB    = IT_MARA "out put table
*   EXCEPTIONS
*     PROGRAM_ERROR                     = 1
*     MAXIMUM_OF_APPENDS_REACHED        = 2
*     OTHERS      = 3
    .
  IF SY-SUBRC <> 0.
* Implement suitable error handling here
  ENDIF.

  CALL FUNCTION 'REUSE_ALV_BLOCK_LIST_APPEND' "append ALV lists
    EXPORTING
      IS_LAYOUT   = S_LAYOUT "set layout
      IT_FIELDCAT = T_FCAT_MAKT "set field catalog
      I_TABNAME   = 'IT_MAKT' "table
      IT_EVENTS   = S_EVENTS "events
*     IT_SORT     =
*     I_TEXT      = ' '
    TABLES
```

```
     T_OUTTAB    = IT_MAKT "out put table
*   EXCEPTIONS
*     PROGRAM_ERROR                    = 1
*     MAXIMUM_OF_APPENDS_REACHED       = 2
*     OTHERS      = 3
    .
  IF SY-SUBRC <> 0.
* Implement suitable error handling here
  ENDIF.
  CALL FUNCTION 'REUSE_ALV_BLOCK_LIST_DISPLAY' . "display blocked list
```

Interactive ALV Report:

To make an interactive report, we use USER COMMAND event as a call back user command for ALV.

Interactive ALV Report code will be:

```
REPORT ZGTP_ALV_INTERACTIVE.
TYPE-POOLS SLIS .
TYPES : BEGIN OF TY_MARA,  "User defined internal table type
        MATNR TYPE MARA-MATNR,
        MTART TYPE MARA-MTART,
        MBRSH TYPE MARA-MBRSH,
        MEINS TYPE MARA-MEINS,
      END OF TY_MARA.

DATA : IT_MARA TYPE TABLE OF TY_MARA ."internal table
DATA : WA_MARA TYPE TY_MARA . "work area

DATA : IT_FCAT TYPE SLIS_T_FIELDCAT_ALV . "field catalog table
DATA : WA_FCAT LIKE LINE OF IT_FCAT . "field catalog work area
PARAMETERS : P_MTART TYPE MARA-MTART. "material type input

START-OF-SELECTION.
**get table data
  SELECT MATNR MTART MBRSH MEINS FROM MARA "get data from MARA
```

```
    INTO TABLE IT_MARA UP TO 50 ROWS
    WHERE MTART = P_MTART.
*** generate field catalogue
  WA_FCAT-COL_POS = '1' . "column position
  WA_FCAT-FIELDNAME = 'MATNR' . "column name
  WA_FCAT-TABNAME = 'IT_MARA' . "table
  WA_FCAT-SELTEXT_M = 'Material' . "Column label
  WA_FCAT-KEY = 'X' . "is a key field
  WA_FCAT-HOTSPOT = 'X' . "Set hotspot for matnr
  APPEND WA_FCAT TO IT_FCAT . "append to fcat
  CLEAR WA_FCAT .
  WA_FCAT-COL_POS = '2' .
  WA_FCAT-FIELDNAME = 'MBRSH' .
  WA_FCAT-TABNAME = 'IT_MARA' .
  WA_FCAT-SELTEXT_M = 'Industry Sec' .
  APPEND WA_FCAT TO IT_FCAT .
  CLEAR WA_FCAT .

  WA_FCAT-COL_POS = '3' .
  WA_FCAT-FIELDNAME = 'MTART' .
  WA_FCAT-TABNAME = 'IT_MARA' .
  WA_FCAT-SELTEXT_M = 'Material Type' .
  APPEND WA_FCAT TO IT_FCAT .
  CLEAR WA_FCAT .

  WA_FCAT-COL_POS = '4' .
  WA_FCAT-FIELDNAME = 'MEINS' .
  WA_FCAT-TABNAME = 'IT_MARA' .
  WA_FCAT-SELTEXT_M = 'Base.Unit' .
  WA_FCAT-REF_TABNAME = 'MARA' .
  APPEND WA_FCAT TO IT_FCAT .
  CLEAR WA_FCAT .
**display ALV
  CALL FUNCTION 'REUSE_ALV_GRID_DISPLAY'
```

```
    EXPORTING
      I_CALLBACK_PROGRAM      = SY-REPID
      I_CALLBACK_USER_COMMAND = 'USER_COMMAND' "user command form
      IT_FIELDCAT             = IT_FCAT "PASS FIELD CATALOG TO ALV
    TABLES
      T_OUTTAB                = IT_MARA. "output table

**for to handle user command
FORM USER_COMMAND USING R_UCOMM LIKE SY-UCOMM
                        RS_SELFIELD TYPE SLIS_SELFIELD.
  CASE R_UCOMM.
    WHEN '&IC1'. "standard Function code for doubel click
      READ TABLE IT_MARA INTO WA_MARA INDEX RS_SELFIELD-TABINDEX.
      IF SY-SUBRC = 0.
        SET PARAMETER ID 'MAT' FIELD WA_MARA-MATNR. "set parameter id
        CALL TRANSACTION 'MM03' AND SKIP FIRST SCREEN. "call transaction
      ENDIF.
  ENDCASE.
ENDFORM.                    "user_command
```

ALV Report With OOPS:

We all know how to develop ALV reports using standard SAP Function Modules, now we are going to learn developing ALV reports using Object Oriented approach.

List of most commonly used classes for OOALV.

- CL_GUI_ALV_GRID.
- CL_GUI_CUSTOM_CONTAINER.
- CL_DD_DOCUMENT.
- CL_GUI_ALV_TREE_SIMPLE.
- CL_GUI_CONTAINER.
- CL_GUI_SPLITTER_CONTAINER.

What are the advantages of Object oriented ALV in SAP?

- We have 'n' number of events available in the classes when compared to ALV with function modules which is flexible for programmer to develop ALV`s for various scenarios.
- We can display more than one ALV grid in single screen.

- By using Object Oriented approach we can control the size of ALV grid by using custom container.
- We can place other UI elements like input field, check box etc on the screen.

Steps need to follow to create OOALV

1. Create Screen
2. Insert Custom Container UI element.
3. Create Module.
4. Create instance for Custom Container and add instance to ALV.
5. Get data from tables
6. Set data to ALV.

Step 1: Data deceleration's for ALV, Custom Container and user defined types of MARA.

Add data deceleration's for ALV grid, custom container UI element and for MARA internal table.Add select-options for material number.

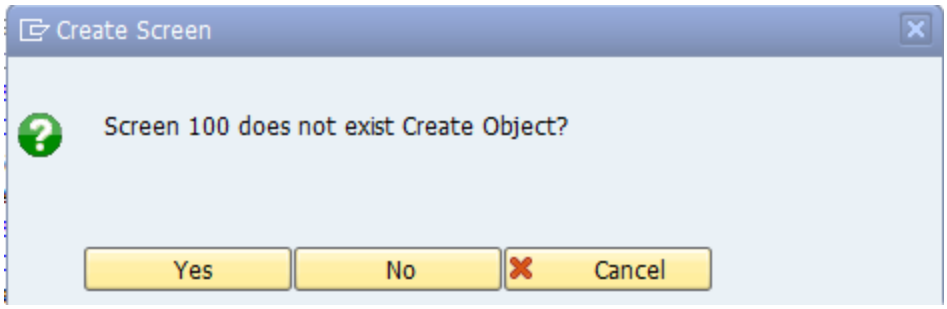
```
DATA : LO_CONT TYPE REF TO CL_GUI_CUSTOM_CONTAINER. "Custom Container

DATA : LO_ALV TYPE REF TO CL_GUI_ALV_GRID. "ALV Grid

TYPES: BEGIN OF TY_MARA,

       MATNR TYPE MARA-MATNR,

       MTART TYPE MARA-MTART,

       MBRSH TYPE MARA-MBRSH,

       MATKL TYPE MARA-MATKL,

       MEINS TYPE MARA-MEINS,

      END OF TY_MARA.

DATA : IT_MARA TYPE TABLE OF TY_MARA,

      WA_MARA TYPE TY_MARA.

TABLES: MARA.

SELECT-OPTIONS: S_MATNR FOR MARA-MATNR.
```

Step 2: Create Screen

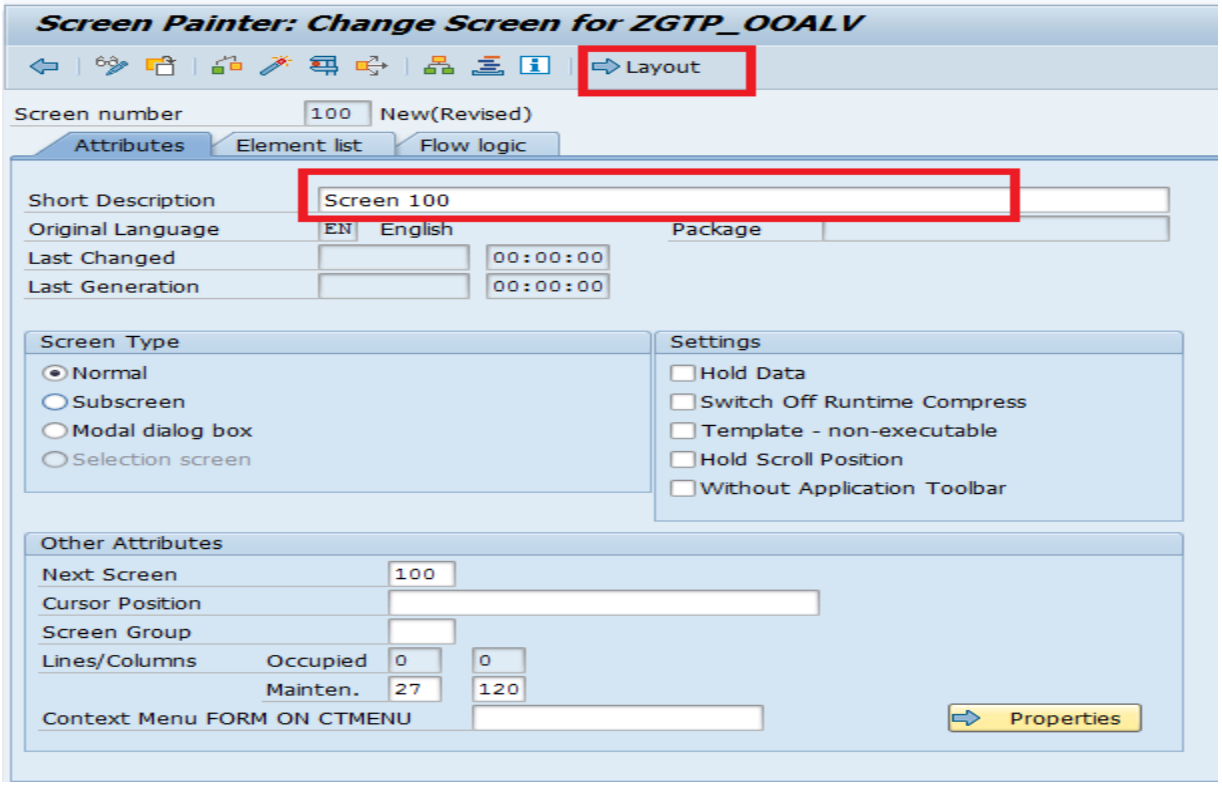
Create Screen in start of selection by using key work CALL SCREEN .

```
START-OF-SELECTION.

**Here 100 is the screen number which we are going to create, you can create any like: 200, 300 etc

CALL SCREEN 100. "double click on 100 to create a screen
```

After calling screen, double click on 100,

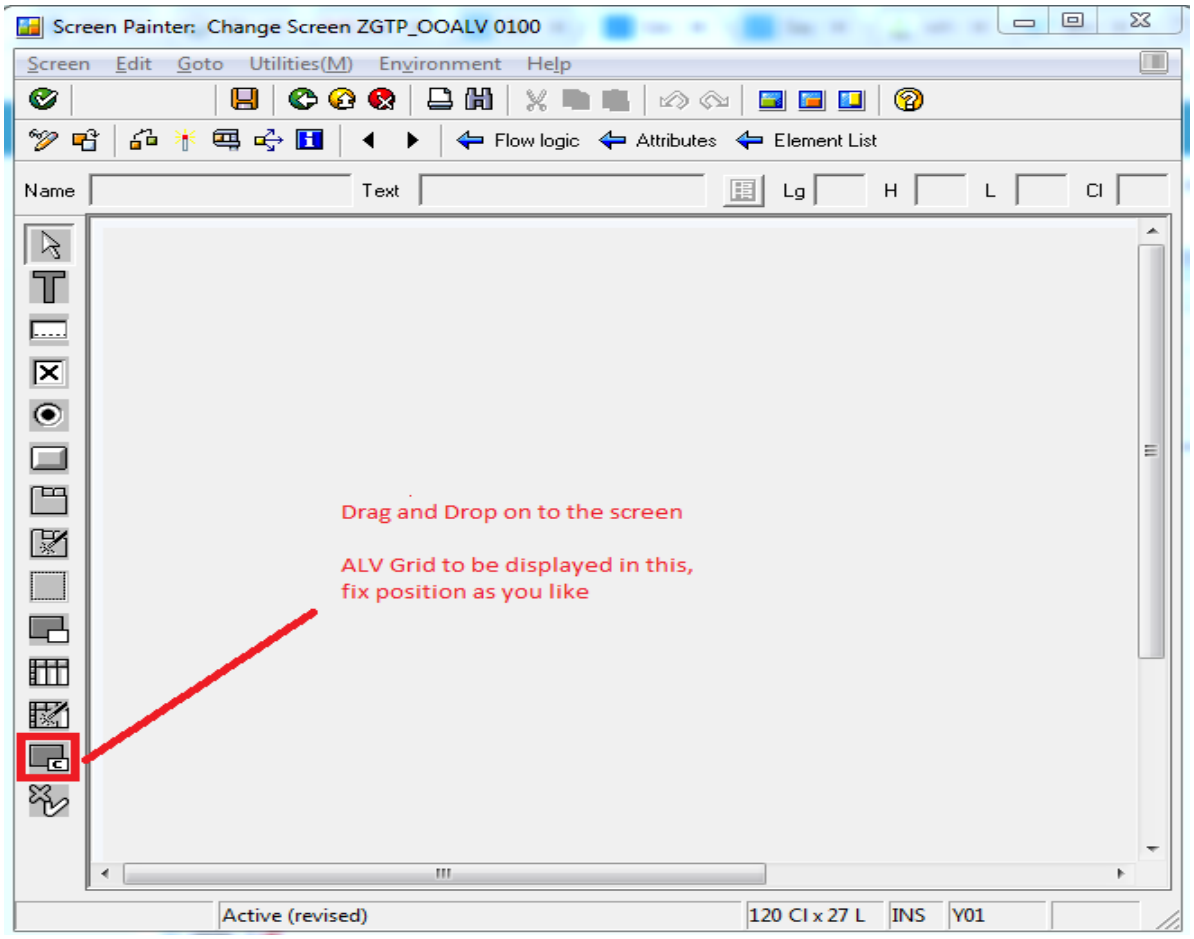


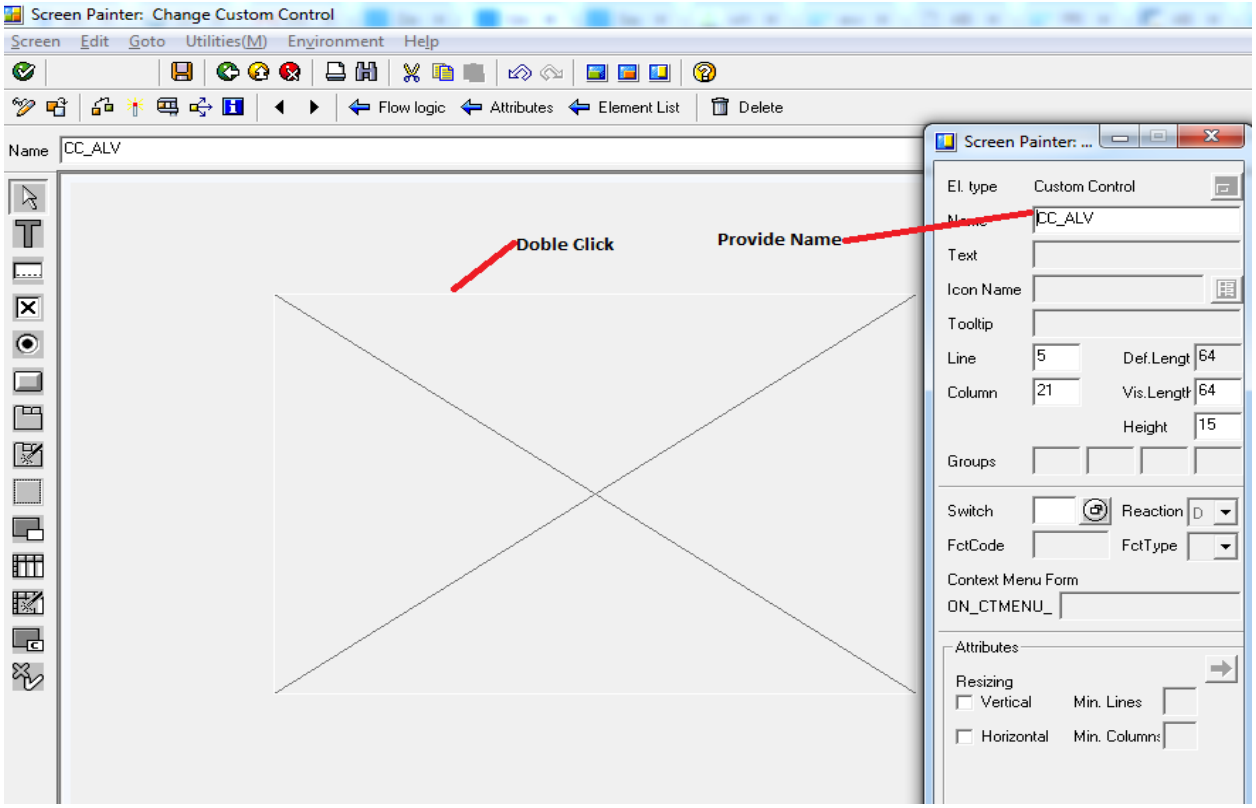
Provide short description, click on layout, layout designer will open(If you use it first it will take some time, wait till it comes, if you face any issue while opening contact BASIS, there might be configuration missing.).



Step 3: Insert Custom Container UI element

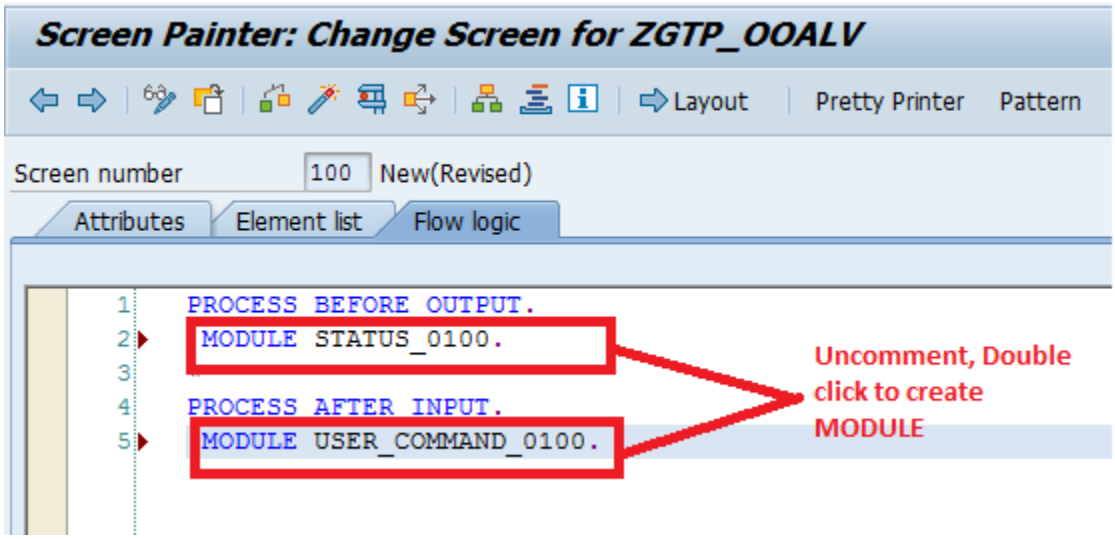
Drag and drop custom container UI element, double click and provide a name and save, activate and close layout designer.


Screen Painter: Change Screen ZGTP_OOALV 0100
Screen Edit Goto Utilities(M) Environment Help
Flow logic
Attributes
Element List
Name
Text
Lg
H
L
Cl
Drag and Drop on to the screen
ALV Grid to be displayed in this,
fix position as you like
Active (revised)
120 Cl x 27 L
INS
Y01



Step 4:Create Modules

Click on flow logic, uncomment MODULES, double click on each one(one at one time), click yes and select main program.



Step 5, 6: Build Field Catalog.

Field Catalog: It is a structure which is used to specify out put structure with field names, field positions, descriptions and additional properties.To see the list of properties of OOALV field catalog, go to se11 and check the structure LVC_S_FCAT.

Building of field catalog is give below.

```
**Declare field catalog table and structure as per class method
DATA : IT_FIELDCATALOG TYPE LVC_T_FCAT,
       WA_FIELDCATALOG TYPE LVC_S_FCAT.
***Build Field Catalogue
  WA_FIELDCATALOG-COL_POS = '1'. "Set column1
  WA_FIELDCATALOG-FIELDNAME = 'MATNR'. "set field
  WA_FIELDCATALOG-TABNAME = 'MARA'. "set table
  WA_FIELDCATALOG-SCRTEXT_M = 'Material No'. "set column description
  APPEND WA_FIELDCATALOG TO IT_FIELDCATALOG. "append to table
  CLEAR : WA_FIELDCATALOG.
  WA_FIELDCATALOG-COL_POS = '2'. "set column2
  WA_FIELDCATALOG-FIELDNAME = 'MTART'.
  WA_FIELDCATALOG-TABNAME = 'MARA'.
  WA_FIELDCATALOG-SCRTEXT_M = 'Material Type'.
  APPEND WA_FIELDCATALOG TO IT_FIELDCATALOG.
  CLEAR : WA_FIELDCATALOG.
  WA_FIELDCATALOG-COL_POS = '3'. "set column 3
  WA_FIELDCATALOG-FIELDNAME = 'MBRSH'.
  WA_FIELDCATALOG-TABNAME = 'MARA'.
  WA_FIELDCATALOG-SCRTEXT_M = 'Industry Sector'.
  APPEND WA_FIELDCATALOG TO IT_FIELDCATALOG.
  CLEAR : WA_FIELDCATALOG.
  WA_FIELDCATALOG-COL_POS = '4'. "set column 4
  WA_FIELDCATALOG-FIELDNAME = 'MATKL'.
  WA_FIELDCATALOG-TABNAME = 'MARA'.
  WA_FIELDCATALOG-SCRTEXT_M = 'Material Group'.
  APPEND WA_FIELDCATALOG TO IT_FIELDCATALOG.
  CLEAR : WA_FIELDCATALOG.
  WA_FIELDCATALOG-COL_POS = '5'. "set column 5
  WA_FIELDCATALOG-FIELDNAME = 'MEINS'.
  WA_FIELDCATALOG-TABNAME = 'MARA'.
  WA_FIELDCATALOG-SCRTEXT_M = 'Unit Of Measure'.
  APPEND WA_FIELDCATALOG TO IT_FIELDCATALOG.
```

```
  CLEAR : WA_FIELDCATALOG.
**End field catalogue
MODULE STATUS_0100 OUTPUT.
**Create object for Custom container
  CREATE OBJECT LO_CONT
    EXPORTING
*     PARENT         =
      CONTAINER_NAME = 'CC_ALV' "container name whcih we have created
.
**Create Object for ALV Grid
  CREATE OBJECT LO_ALV
    EXPORTING
      I_PARENT = LO_CONT "Object of custom container
.
**Get data from MARA for user input
  SELECT MATNR MTART MBRSH MATKL MEINS FROM MARA
    INTO TABLE IT_MARA WHERE MATNR IN S_MATNR.

**Display ALV data using structure
  CALL METHOD LO_ALV->SET_TABLE_FOR_FIRST_DISPLAY
    EXPORTING
*     I_BUFFER_ACTIVE  =
*     I_BYPASSING_BUFFER           =
*     I_CONSISTENCY_CHECK          =
*     I_STRUCTURE_NAME = 'MARA'
*     IS_VARIANT       =
*     I_SAVE           =
*     I_DEFAULT        = 'X'
*     IS_LAYOUT        =
*     IS_PRINT         =
*     IT_SPECIAL_GROUPS            =
*     IT_TOOLBAR_EXCLUDING         =
*     IT_HYPERLINK     =
*     IT_ALV_GRAPHICS  =
```

```
*     IT_EXCEPT_QINFO  =
*     IR_SALV_ADAPTER  =
    CHANGING
      IT_OUTTAB       = IT_MARA
      IT_FIELDCATALOG = IT_FIELDCATALOG
*     IT_SORT         =
*     IT_FILTER       =
*  EXCEPTIONS
*     INVALID_PARAMETER_COMBINATION = 1
*     PROGRAM_ERROR    = 2
*     TOO_MANY_LINES   = 3
*     OTHERS           = 4
    .
  IF SY-SUBRC <> 0.
* Implement suitable error handling here
  ENDIF.


ENDMODULE.                    " STATUS_0100  OUTPUT
```

Final and Modularized code will be

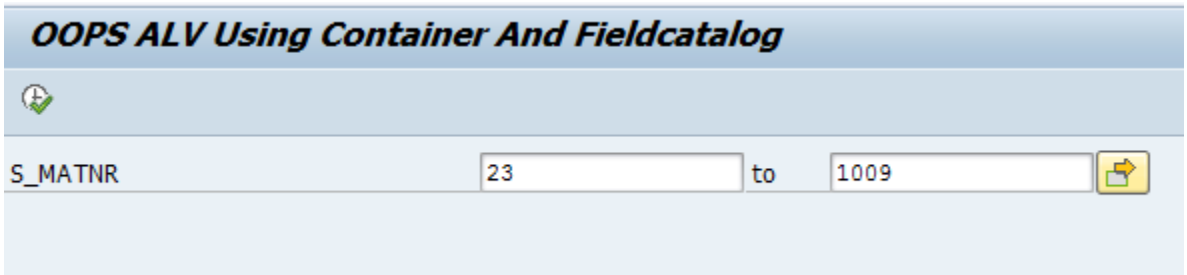
```
REPORT ZSAN_OOALV_FCAT.
TABLES: MARA.
TYPES: BEGIN OF TY_MARA,
        MATNR TYPE MARA-MATNR,
        MTART TYPE MARA-MTART,
        MBRSH TYPE MARA-MBRSH,
        MATKL TYPE MARA-MATKL,
        MEINS TYPE MARA-MEINS,
      END OF TY_MARA.
DATA : IT_MARA TYPE TABLE OF TY_MARA,
       WA_MARA TYPE TY_MARA.
DATA : LO_CONT TYPE REF TO CL_GUI_CUSTOM_CONTAINER. "custom container
DATA : LO_ALV TYPE REF TO CL_GUI_ALV_GRID. "alv grid
DATA : IT_FIELDCATALOG TYPE LVC_T_FCAT,
```

```
      WA_FIELDCATALOG TYPE LVC_S_FCAT.
SELECT-OPTIONS: S_MATNR FOR MARA-MATNR.
CALL SCREEN 100. "double click to create
MODULE STATUS_0100 OUTPUT.
*  SET PF-STATUS 'xxxxxxxx'.
*  SET TITLEBAR 'xxx'.
  CREATE OBJECT LO_CONT
    EXPORTING
*     PARENT        =
      CONTAINER_NAME = 'CC_ALV'.
  CREATE OBJECT LO_ALV
    EXPORTING
      I_PARENT = LO_CONT.
***Build Field Catalogue
  WA_FIELDCATALOG-COL_POS = '1'. "Set column1
  WA_FIELDCATALOG-FIELDNAME = 'MATNR'.
  WA_FIELDCATALOG-TABNAME = 'MARA'.
  WA_FIELDCATALOG-SCRTEXT_M = 'Material No'.
  APPEND WA_FIELDCATALOG TO IT_FIELDCATALOG.
  CLEAR : WA_FIELDCATALOG.
  WA_FIELDCATALOG-COL_POS = '2'. "set column2
  WA_FIELDCATALOG-FIELDNAME = 'MTART'.
  WA_FIELDCATALOG-TABNAME = 'MARA'.
  WA_FIELDCATALOG-SCRTEXT_M = 'Material Type'.
  APPEND WA_FIELDCATALOG TO IT_FIELDCATALOG.
  CLEAR : WA_FIELDCATALOG.
  WA_FIELDCATALOG-COL_POS = '3'. "set column 3
  WA_FIELDCATALOG-FIELDNAME = 'MBRSH'.
  WA_FIELDCATALOG-TABNAME = 'MARA'.
  WA_FIELDCATALOG-SCRTEXT_M = 'Industry Sector'.
  APPEND WA_FIELDCATALOG TO IT_FIELDCATALOG.
  CLEAR : WA_FIELDCATALOG.
  WA_FIELDCATALOG-COL_POS = '4'. "set column 4
  WA_FIELDCATALOG-FIELDNAME = 'MATKL'.
```

```
  WA_FIELDCATALOG-TABNAME = 'MARA'.
  WA_FIELDCATALOG-SCRTEXT_M = 'Material Group'.
  APPEND WA_FIELDCATALOG TO IT_FIELDCATALOG.
  CLEAR : WA_FIELDCATALOG.
  WA_FIELDCATALOG-COL_POS = '5'. "set column 5
  WA_FIELDCATALOG-FIELDNAME = 'MEINS'.
  WA_FIELDCATALOG-TABNAME = 'MARA'.
  WA_FIELDCATALOG-SELTEXT = 'Unit Of Measure'.
  APPEND WA_FIELDCATALOG TO IT_FIELDCATALOG.
  CLEAR : WA_FIELDCATALOG.
**End field catalogue
  SELECT MATNR MTART MBRSH MATKL MEINS FROM MARA
    INTO TABLE IT_MARA WHERE MATNR IN S_MATNR.

  CALL METHOD LO_ALV->SET_TABLE_FOR_FIRST_DISPLAY
*  EXPORTING
*    I_BUFFER_ACTIVE              =
*    I_BYPASSING_BUFFER           =
*    I_CONSISTENCY_CHECK          =
*    I_STRUCTURE_NAME             =
*    IS_VARIANT                   =
*    I_SAVE                       =
*    I_DEFAULT                    = 'X'
*    IS_LAYOUT                    =
*    IS_PRINT                     =
*    IT_SPECIAL_GROUPS            =
*    IT_TOOLBAR_EXCLUDING         =
*    IT_HYPERLINK                 =
*    IT_ALV_GRAPHICS              =
*    IT_EXCEPT_QINFO              =
*    IR_SALV_ADAPTER              =
    CHANGING
      IT_OUTTAB       = IT_MARA
      IT_FIELDCATALOG = IT_FIELDCATALOG
```

```
*    IT_SORT        =
*    IT_FILTER      =
*  EXCEPTIONS
*    INVALID_PARAMETER_COMBINATION = 1
*    PROGRAM_ERROR   = 2
*    TOO_MANY_LINES  = 3
*    OTHERS          = 4
    .
  IF SY-SUBRC <> 0.
* Implement suitable error handling here
  ENDIF.



ENDMODULE.                " STATUS_0100  OUTPUT


MODULE USER_COMMAND_0100 INPUT.


ENDMODULE.                " USER_COMMAND_0100  INPUT
```

Input Screen is as fallow:



Output Screen is:

Material No	Material Type	Industry Se...	Material Gro...	
000000000000000023	ROH	1	00107	EA
000000000000000038	HALB	M	00107	ST
000000000000000043	HAWA	1		STD
000000000000000058	HIBE	M		ST
000000000000000059	HIBE	M		ST
000000000000000068	FHMI	A	013	ST
000000000000000078	DIEN	M		ST
000000000000000088	FERT	M	02004	ST
000000000000000089	FERT	M	02004	ST
000000000000000098	HALB	M	002	ST
000000000000000170	NLAG	M	004	ST
000000000000000178	NLAG	M	004	ST