[ALV TUTOTIAL FOR BEGINERS WITH EXAMPLES](https://wiki.scn.sap.com/wiki/display/ABAP/ALV+TUTOTIAL+FOR+BEGINERS+WITH+EXAMPLES)

 Introduction  
   
This is a discussion on ALV reporting-a step by step operation manual about creating an ALV report of medium complexity. This documentation will not make you an expert in ALV. But, definitely, it will come handy at the time of need. It refers an example report YSUBALV, as a reference and will represent relevant blocks of the code at different areas of discussion.    
**Contents**   
ALV (ABAP LIST VIEWER)

Step 1: DATA DECLARATION

Example:

Step 2: Selecting the variants(Optional)

Step 3: (Defining Output Characteristics)

Step 4: (Build up events table)

Step 5: (Report Output list description)

Step 6: (Pass Selection-screen Information)

Step 7: (Deciding Sort Criteria)

Step 8: (Final Step)

Step 9: (Display Internal Output Table)

Sample code.

**Description:**Sap provides a set of ALV (ABAP LIST VIEWER) function modules, which can be put into use to embellish the output of a report. This set of ALV functions is used to enhance the readability and functionality of any report output. Cases arise in sap when the output of a report contains columns extending more than 255 characters in length.  In such cases, this set of ALV functions can help choose selected columns and arrange the different columns from a report output and also save different variants for report display. This is a very efficient tool for dynamically sorting and arranging the columns from a report output. The report output can contain up to 90 columns in the display with the wide array of display options.  
 The commonly used ALV functions used for this purpose are;  
1.      REUSE\_ALV\_VARIANT\_DEFAULT\_GET  
2.      REUSE\_ALV\_VARIANT\_F4  
3.      REUSE\_ALV\_VARIANT\_EXISTENCE  
4.      REUSE\_ALV\_EVENTS\_GET  
5.      REUSE\_ALV\_COMMENTARY\_WRITE  
6.      REUSE\_ALV\_FIELDCATALOG\_MERGE  
7.      REUSE\_ALV\_LIST\_DISPLAY  
8.      REUSE\_ALV\_GRID\_DISPLAY  
9.      REUSE\_ALV\_POPUP\_TO\_SELECT  
The different steps used for getting the above function modules into use are described below.

Step 1: DATA DECLARATION

Sap standard type pools:  SLIS, KKBLO.  
Sap standard tables types taken from the type pools are:   
SLIS\_LAYOUT\_ALV,  
SLIS\_T\_FIELDCAT\_ALV  
SLIS\_T\_LISTHEADER,  
SLIS\_T\_EVENT,  
SLIS\_SELFIELD.  
 Internal tables to be used in the program declared based on the above table types  
DATA:   I\_LAYOUT                     TYPE SLIS\_LAYOUT\_ALV,  
            I\_FIELDTAB                  TYPE SLIS\_T\_FIELDCAT\_ALV,  
            I\_HEADING                   TYPE SLIS\_T\_LISTHEADER,  
I\_EVENTS                     TYPE SLIS\_T\_EVENT.  
 TYPES:  KKBLO\_SELFIELD TYPE SLIS\_SELFIELD.

Example: -

 REPORT YSUBALV.  
 \*----------~~Declaration of variables~~-----------------\*  
TYPE-POOLS: SLIS.  
\* To pass name of the report in function module for ALV  
Data: V\_REPID LIKE SY-REPID.  
\* To pass the overall structure of the ALV report  
Data: STRUCT\_LAYOUT            TYPE SLIS\_LAYOUT\_ALV.  
Data: STRUCT\_LAYOUT1           TYPE SLIS\_LAYOUT\_ALV.  
 \* Internal table to capture various events in ALV  
Data: I\_EVENTS            TYPE SLIS\_T\_EVENT.  
 \* Table for catalog of the fields to be displayed  
Data: I\_FIELDCAT                      TYPE SLIS\_T\_FIELDCAT\_ALV.  
Data: X\_FIELDCAT                     TYPE SLIS\_FIELDCAT\_ALV.  
Data: I\_FIELDCAT1                    TYPE SLIS\_T\_FIELDCAT\_ALV.  
Data: X\_FIELDCAT1       TYPE SLIS\_FIELDCAT\_ALV.  
 \* Internal table to mention the sort sequence  
Data: IT\_SORT TYPE SLIS\_T\_SORTINFO\_ALV.  
Data: X\_SORT   TYPE SLIS\_SORTINFO\_ALV.  
 \* Internal table to display top of page  
Data: i\_list\_top\_of\_page type slis\_t\_listheader.  
\* Structure to display variants  
Data:    i\_variant            like disvariant,  
I\_variant1           like disvariant.  
 \* Internal table to pass data  
DATA: BEGIN OF I\_TAB OCCURS 0,  
Mblnr    like mseg-mblnr ,  
Matnr    like mseg-matnr,  
Maktg   like makt-maktg,  
Charg    like mseg-charg,  
Werks   like mseg-werks,  
Lgort     like mseg-lgort,  
menge like mseg-menge ,  
meins   like mseg-meins ,  
dmbtr    like mseg-dmbtr,  
ebeln    like mseg-ebeln,  
icn(4)    type c ,  
sym(4)   type c ,  
excpt(2) type c ,  
box(1),  
END OF I\_TAB.  
 \*EJECT  
 Data: begin of i\_doc occurs 0.  
INCLUDE STRUCTURE MSEG.  
Data: end of i\_doc.

Step 2: Selecting the variants (Optional)

**SELECTING THE VARIANTS FOR INITIAL LIST DISPLAY (DEFAULT VARIANT)**

The variants in the list display can be both user-specific and general. The user can programmatically set the initial (default) variant for list display. The default variant can be found using the function module 'REUSE\_ALV\_VARIANT\_DEFAULT\_GET'.  
 Sample code:  
 CALL FUNCTION 'REUSE\_ALV\_VARIANT\_DEFAULT\_GET'  
EXPORTING  
i\_save               = variant save condition (A=all, U = user-specific)  
CHANGING  
cs\_variant          = internal table containing the program name (and the default variant---optional)  
EXCEPTIONS  
not\_found = 2.

The user can also choose from the list of existing variants using the function module  'REUSE\_ALV\_VARIANT\_F4'. Example:  
 Initialization.  
v\_repid = sy-repid.  
\* Display default variant  
 PERFORM SUB\_VARIANT\_INIT.  
 AT SELECTION-SCREEN ON P\_VAR.  
\* Once the user has entered variant, check about its existence  
 PERFORM SUB\_CHECK\_PVAR.  
 AT SELECTION-SCREEN ON VALUE-REQUEST FOR P\_VAR.  
\* Display a list of various variants of the report when the User presses F4 key in the variant field  
 PERFORM SUB\_VARIANT\_F4.  
   
 \*&---------------------------------------------------------------------\*  
\*&      Form SUB\_VARIANT\_INIT  
\*&---------------------------------------------------------------------\*  
\* Display default variant  
\*&---------------------------------------------------------------------\*  
Form SUB\_VARIANT\_INIT.  
I\_VARIANT1-REPORT = SY-REPID.  
 \* Search default variant for the report  
  CALL FUNCTION 'REUSE\_ALV\_VARIANT\_DEFAULT\_GET'  
       EXPORTING  
            i\_save     = 'A'  
       CHANGING  
            cs\_variant = i\_variant1  
       EXCEPTIONS  
            not\_found = 2.  
\* If default variant is found, use it as default.  
\* Else, use the variant LAYOUT1.  
  IF sy-subrc = 0.  
    p\_var = i\_variant1-variant.  
  ELSE.  
    p\_var = 'LAYOUT1'.  
  ENDIF.  
endform.                    "SUB\_VARIANT\_INIT  
 \*&---------------------------------------------------------------------------------\*  
\*&      Form SUB\_CHECK\_PVAR  
\*&---------------------------------------------------------------------------------\*  
\* Once the user has entered variant, check about its existence  
\*&---------------------------------------------------------------------------------\*  
 FORM SUB\_CHECK\_PVAR.  
\* If the name of the variable is not blank, check about its existence  
if not p\_var is initial.  
  clear i\_variant.  
  i\_variant-report = sy-repid.  
  i\_variant-variant = p\_var.  
  CALL FUNCTION 'REUSE\_ALV\_VARIANT\_EXISTENCE'  
         EXPORTING  
              I\_SAVE     = 'A'  
         CHANGING  
              CS\_VARIANT = I\_VARIANT.  
\* If no such variant found, flash error message  
     If sy-subrc ne 0 .  
      Message e398(00) with 'No such variant exists'.  
     Else.  
\* If variant exists, use the variant name to populate structure  
\* I\_VARIANT1 which will be used for export parameter: IS\_VARIANT  
\* in the function module : REUSE\_ALV\_GRID\_DISPLAY  
       Clear i\_variant1.  
       Move p\_var to i\_variant1-variant.  
       Move sy-repid to i\_variant1-report.  
     endif.  
 Else.  
   Clear i\_variant.  
 endif.  
ENDFORM.                    "SUB\_CHECK\_PVAR  
 \*&---------------------------------------------------------------------------------\*  
\*&      Form SUB\_VARIANT\_F4  
\*&---------------------------------------------------------------------------------\*  
\* Display a list of various variants of the report when the User presses F4 key in the variant field  
\*&---------------------------------------------------------------------------------\*  
Form SUB\_VARIANT\_F4.  
i\_variant-report = sy-repid.  
\* Utilizing the name of the report, this function module will search for a list of  
\* variants and will fetch the selected one into the parameter field for variants  
CALL FUNCTION 'REUSE\_ALV\_VARIANT\_F4'  
       EXPORTING  
            IS\_VARIANT         = I\_VARIANT  
            I\_SAVE                                     = 'A'  
            I\_DISPLAY\_VIA\_GRID   = 'X'  
       IMPORTING  
            ES\_VARIANT                = I\_VARIANT1  
       EXCEPTIONS  
            NOT\_FOUND                 = 1  
            PROGRAM\_ERROR      = 2  
            OTHERS                                   = 3.  
  IF SY-SUBRC = 0.  
    P\_VAR = I\_VARIANT1-VARIANT.  
 ENDIF.  
endform.                    "SUB\_VARIANT\_F4

Step3 (Defining Output Characteristics)

**DEFININING OUTPUT CHARACTERISTICS: PREPARING DISPLAY FIELDS CATALOG**

 A field catalog is prepared using the internal table (I\_FIELDCAT) of type SLIS\_T\_FIELDCAT\_ALV. Field catalog containing descriptions of the list output fields (usually a subset of the internal output table fields).  
 A field catalog is required for every ALV list output to add desired functionality (i.e. Key, Hotspot, Specific headings, Justify, Col. position etc) to certain fields of the output. If not mentioned specifically, then the defaults are taken. The possible values and defaults are listed below.  
The field catalog for the output table is built-up in the caller's coding. The build-up can be completely or partially automated by calling the  
REUSE\_ALV\_FIELDCATALOG\_MERGE module.  
The minimal field catalog is documented below. This can be done in a routine using a local variable. The user can use the other optional parameters to assign output attributes to different fields in the output, which differ from the default.  
 A field catalog need not be built-up and passed explicitly only under the following conditions:  
1.    The internal table to be output has the same structure as a Data Dictionary structure which is referred to in the internal table declaration using LIKE or INCLUDE STRUCTURE. In this case the attributes of the different fields is taken directly from the table and the attributes (key fields, length, texts etc) need to state explicitly.  
 2.    All fields in this structure are to be output  
 3.    The structure name is passed to ALV in the parameter I\_STRUCTURE\_NAME of the function module REUSE\_ALV\_LIST\_DISPLAY.  
All the values entered in the catalog are specific to the particular field whose name is entered in the fieldname FIELDNAME of the fieldcat structure. The name of the table is also entered in the corr. Fieldname TABNAME of the structure.

 The different possible attributes are:  
 -          **Row\_pos** (row position): Only relevant if the list output is to be multi-line (two or three lines) by default. So, this attribute can be used maintain certain level of alignment in the output.  
                   Value set: 0, 1 - 3  
 -          **Col\_pos** (column position): This parameter is relevant when the fields in the output are to be different from the sequence of the fields in the internal table used for display.       The parameter specifies the relative column position of the field in the list output. The column order can be changed interactively by the user. If this parameter is initial for all field catalog entries, columns appear in the internal table field sequence.  
Value set: 0, 1 - 60  
 -          **Fieldname** (field name): This is the name of the internal table field for which the parameters are passed in the catalog.  
Value set: internal output table field name (required parameter)  
                         
-          **Tabname** (internal output table): Name of the internal output table that contains the field FIELDCAT-FIELDNAME above.  
                   Value set: SPACE, internal output table name.  
 -          **Ref\_fieldname** (reference field name): Name of the Data Dictionary field referred to. This parameter is only used when the internal output table field described by the current field catalog entry has a reference to the Data Dictionary (not a program field), and the field name in the internal output table is different from the name of the field in the Data Dictionary. If the field names are identical, naming the Data Dictionary structure or table in the FIELDCAT-REF\_TABNAME parameter is sufficient.  
    Value set: SPACE, Data Dictionary field name.  
 -          **Ref\_tabname** (reference table/structure field name): Structure or table name of the referred Data Dictionary field. This parameter is only used when the internal output table field described by the current field catalog entry has a Data Dictionary reference (not a program field).  
Value set: SPACE, name of a Data Dictionary structure or table

**Link to currency unit**

-          **Cfieldname** (currency unit field name): This is used for currency fields that have a reference to any unit field. This is only relevant for amount columns with associated unit. This parameter contains the Name of the internal output table field containing the currency unit associated with the amount field FIELDCAT-FIELDNAME. The field in FIELDCAT-CFIELDNAME must have its own field catalog entry.  
Value set: SPACE, output table field name.  
 -          **Ctabname** (internal currency unit field output table): Name of the internal output table containing the FIELDCAT-CFIELDNAME field.  
Value set: SPACE, output table field name.  
             Link to measurement unit  
-          **Qfieldname** (measurement unit field name): Only relevant for quantity columns with unit link. Name of the internal output table field containing the measurement unit associated with the quantity field FIELDCAT-FIELDNAME. The field in FIELDCAT-QFIELDNAME must have its own field catalog entry.  
Value set: SPACE, output table field name.  
 -          **Qtabname** (internal measurement unit field output table): Name of the internal output table containing the FIELDCAT-QFIELDNAME field.  
 Value set: SPACE, output table field name.  
 -          **Outputlen** (column width): This parameter is used if the desired output length for a field is desired to be different from the internal output table field. For fields with a Data Dictionary link this parameter can be left initial. For fields without a Data Dictionary link (program field) the parameter must be given the value of the desired field list output length (column width).  
Initial = column width is the output length of the referred Data Dictionary field (domain).  
N = column width is n characters.  
      Value set: 0 (initial), n.  
 -          **Key** (key column): By default, the system makes some fields in the output as key fields, provided the fields are key fields in their referencing table. Using this parameter, fields other than key fields of the referencing table can be made key field. This parameter is most important if the output needs to contain some field, which are not scrollable or cannot be hidden.  
If the internal output table contains fields that are key fields from different tables, then all those fields in the report output becomes un-scrollable and cannot be hidden. So, the fields in the output internal table should not be referenced from tables in which they are key fields. Instead, they should be referenced to the tables in which they are not key fields, incase they are not desired as key field in the output.  
'X' = key field (key field output in color) and Key fields cannot be interactively hidden. Parameter FIELDCAT-NO\_OUT must be left initial.  
      Value set: SPACE, 'X'.  
 -          **Key\_sel** (hide-able key column): This parameter is only relevant for the fields which are made key fields using FIELDCAT-KEY = 'X'. Using this parameter the Key field can be hidden interactively.  
The key column sequence cannot be changed interactively by the user. The output is controlled by the FIELDCAT-NO\_OUT parameter analogously to non-key fields.  
Value set: SPACE, 'X'.  
 -          **No\_out** (field in field list): This parameter is used to remove certain fields from the output during initial display.  The user can however interactively choose the field for output from the field list in the display variant.  
                   'X' = field is not displayed in the current list.  
        Value set: SPACE, 'X'.  
 -          **Tech** (technical field): This parameter is used to make certain field display only in the field catalog. The fields with this parameter set cannot be output in the list nor can they be displayed interactively from the catalog.  
                    'X' = technical field.  
       Value set: SPACE, 'X'.  
 -          **Emphasize** (highlight columns in color): As name suggests, this field parameter is used to highlight certain field with chosen colors.  
Value set: SPACE, 'X' or 'Cxyz' (x:'1'-'9'; y,z: '0'=off ,'1'=on).  
'X' = column is colored with the default column highlight color.  
                   'Cxyz' = column is colored with a coded color:  
                         -  C: Color (coding must begin with C)  
                         -  X: color number  
                         -  Y: bold  
                         -  Z: inverse  
 -          **Hotspot** (column as hotspot): This parameter is used to make certain field appear as hotspot i.e. a hand is displayed if the cursor is positioned on the field value. Single click on such fields cause the PICK OR F2 events to happen.  
         Value set: SPACE, 'X'.  
                    'X' = column cells are output as hotspots.  
 -          **Fix\_column** (fix column): This parameter is used to fix certain columns in the output. All columns to be fixed must have this flag, starting from the left. If a column without this flag is output, only the columns to the left of this column are fixed. The user can change the column fixing interactively.  
                   Value set: SPACE, 'X'.  
        'X' = column fixed (does not scroll horizontally).  
        
-          **Do\_sum** (sum over column): the user can also call this function interactively.  
             Value set: SPACE, 'X'.  
'X' = a sum is calculated over this internal output table field.  
 -          **No\_sum** (sums forbidden): No sum can be calculated over this field, although the data type of the field would allow summing.  
         Value set: SPACE, 'X'.  
 -          **Icon:** The parameter displays certain column contents as icons. The internal output table column contents must be valid icon strings.  
                    Value set: SPACE, 'X'.  
                     'X' = column contents to be output as an icon.  
 -          **Symbol:** The internal output table column must be a valid symbol character.  
Value set: SPACE, 'X'  
                   'X' = column contents are to be output as a symbol.  
 -          **Just** (justification): This parameter is used for alignment of the contents of the output table. This is only relevant for CHAR or NUMC fields in the output internal table. The justification of the column header always follows the justification of the columns. Independent justification of the column header is not possible.  
Value set: SPACE, 'R', 'L', and 'C'.  
       ' ' = Default justification for this data type  
       'R' = right-justified output  
       'L' = left-justified output  
       'C' = centered output.  
   
-          **Lzero** (leading zeros): By default ALV outputs NUMC fields right-justified without leading zeros. Using this parameter only the NUMC fields can be displayed with leading zeroes.  
                   Value set: SPACE, 'X'.  
                    'X' = output with leading zeros.  
 -          **No\_sign** (no +/- sign): This parameter is used to suppress the signs of the output fields. It is only relevant for the value fields.  
    Value set: SPACE, 'X'.  
                  'X' = value output without +/ sign.  
 -          **No\_zero** (suppress zeros): Only relevant for value fields.  
             Value set: SPACE, 'X'.  
                    'X' = suppress zeros.  
 -          **Edit\_mask** (field formatting): To apply the report output formatting options same as in the WRITE statement in report writing.  
           Value set: SPACE, template.  
   
The following parameters are used for customizing the texts in the heading of the output of the columns. The texts are taken from the Data Dictionary for fields with a Data Dictionary reference. If this is not desired, the text parameters can also be specified. The Data Dictionary texts are then ignored.  
If the user changes the column width interactively, the column header text with the appropriate length is always used.  
The interactive function 'Optimize column width' takes account of both the field contents and the column headers: if all field contents are shorter than the shortest column header, the column width depends on the column header.  
      The 'long field label' is also used in display variant definition,  
        Sort, etc.  Popup.  
 -          **seltext\_l** (long field label)  
 -          **seltext\_m** (medium field label)  
 -          **seltext\_s** (short field label)  
 -          **reptext\_ddic** (header) Analogous to the Data element main header  
 -          **Ddictxt** (specify text): You can specify with values 'L', 'M', and 'S', the keyword that should always be used as column header. If the column width changes, no attempt is made in this case to find an appropriate header for the new output width.  
Value set: SPACE, 'L', 'M', and 'S'.  
 **Sample code:**  
\*&---------------------------------------------------------------------\*  
\*&      Form SUB\_PREPARE\_FIELDCATALOG  
\*&---------------------------------------------------------------------\*  
\* Prepare field catalog for the main report. State the name of the field, name of internal table, various formatting options etc  
\*----------------------------------------------------------------------\*  
Form SUB\_PREPARE\_FIELDCATALOG.  
  X\_FIELDCAT-COL\_POS = 1.                          \* First field to appear in ALV list  
  X\_FIELDCAT-FIELDNAME           = 'SYM'.                 \* Name of the internal table field  
  X\_FIELDCAT-TABNAME              = 'I\_TAB'.              \* Name of the internal table  
  X\_FIELDCAT-SELTEXT\_M           = 'Stat'.                   \* Heading for the field  
  X\_FIELDCAT-SYMBOL = 'X'.                       \* The field is going to contain a symbol  
\* Append the specifications to the internal table for field catalog.  
APPEND X\_FIELDCAT TO I\_FIELDCAT.  
CLEAR X\_FIELDCAT.  
 X\_FIELDCAT-COL\_POS   = 2.                                          \* Second field to appear in ALV list  
X\_FIELDCAT-FIELDNAME             = 'MATNR'.           \* Name of the field in the internal table  
X\_FIELDCAT-TABNAME                = 'I\_TAB'.              \* Name of the internal table  
X\_FIELDCAT-SELTEXT\_M             = 'MatItem'.           \* Heading for the column  
\* It is going to be the key field.The color for this field is going to be different  
X\_FIELDCAT-KEY                             = 'X'.  
X\_FIELDCAT-KEY\_SEL    = 'X'.  
\* Single click on the field will trigger double click event. Also, a hand will appear when the cursor navigates to the field  
X\_FIELDCAT-HOTSPOT = 'X'.  
X\_FIELDCAT-FIX\_COLUMN          = 'X'.                       \* The column and those left to it will not scroll  
X\_FIELDCAT-REF\_TABNAME      = 'MSEG'.               \* F1 help will come as it is referenced to DDIC table  
APPEND X\_FIELDCAT TO I\_FIELDCAT.  
CLEAR X\_FIELDCAT.  
 X\_FIELDCAT-COL\_POS   = 3.  
X\_FIELDCAT-FIELDNAME             = 'MAKTG'.  
X\_FIELDCAT-TABNAME                = 'I\_TAB'.  
X\_FIELDCAT-SELTEXT\_M             = 'Description'.  
\* X\_FIELDCAT-OUTPUTLEN          = 50.  
X\_FIELDCAT-HOTSPOT = space.  
X\_FIELDCAT-JUST                            = 'C'.        \* The field is centre(C for centre, R and L for left and right) justified  
X\_FIELDCAT-KEY                             = 'X'.  
X\_FIELDCAT-FIX\_COLUMN          = 'X'.  
\*X\_fieldcat-no\_out                             = 'X'.  
X\_FIELDCAT-FIX\_COLUMN          = 'X'.  
APPEND X\_FIELDCAT TO I\_FIELDCAT.  
CLEAR X\_FIELDCAT.  
 X\_FIELDCAT-COL\_POS   = 4.  
X\_FIELDCAT-FIELDNAME             = 'CHARG'.  
X\_FIELDCAT-TABNAME                = 'I\_TAB'.  
X\_FIELDCAT-SELTEXT\_M             = 'Batch'.  
\* X\_FIELDCAT-OUTPUTLEN          = 10.  
X\_FIELDCAT-HOTSPOT = space.  
APPEND X\_FIELDCAT TO I\_FIELDCAT.  
CLEAR X\_FIELDCAT.  
 X\_FIELDCAT-COL\_POS   = 5.  
X\_FIELDCAT-FIELDNAME             = 'EBELN'.  
X\_FIELDCAT-TABNAME                = 'I\_TAB'.  
X\_FIELDCAT-SELTEXT\_M             = 'Purchase Order'.  
\* X\_FIELDCAT-OUTPUTLEN          = 14.  
X\_FIELDCAT-EMPHASIZE              = 'C511'.                  \* The field will be colored differently (Cxyz)  
X\_FIELDCAT-NO\_OUT                    = 'X'.                       \* Initially the field will be hidden  
APPEND X\_FIELDCAT TO I\_FIELDCAT.  
CLEAR X\_FIELDCAT.  
 X\_FIELDCAT-COL\_POS   = 6.  
X\_FIELDCAT-FIELDNAME             = 'MBLNR'.  
X\_FIELDCAT-TABNAME                = 'I\_TAB'.  
X\_FIELDCAT-SELTEXT\_M             = 'Document no'.  
\* X\_FIELDCAT-OUTPUTLEN          = 14.  
X\_FIELDCAT\_EMPHASIZE             = 'C711'.  
X\_FIELDCAT-NO\_OUT    = 'X'.  
APPEND X\_FIELDCAT TO I\_FIELDCAT.  
CLEAR X\_FIELDCAT.  
 X\_FIELDCAT-COL\_POS   = 7.  
X\_FIELDCAT-FIELDNAME             = 'WERKS'.  
X\_FIELDCAT-TABNAME                = 'I\_TAB'.  
X\_FIELDCAT-SELTEXT\_M             = 'Plant'.  
\* X\_FIELDCAT-OUTPUTLEN          = 5.  
X\_FIELDCAT-EMPHASIZE              = 'C310'.  
APPEND X\_FIELDCAT TO I\_FIELDCAT.  
CLEAR X\_FIELDCAT.  
 X\_FIELDCAT-COL\_POS   = 8.  
X\_FIELDCAT-FIELDNAME             = 'LGORT'.  
X\_FIELDCAT-TABNAME                = 'I\_TAB'.  
X\_FIELDCAT-SELTEXT\_M             = 'St.Loc'.  
\* X\_FIELDCAT-OUTPUTLEN          = 7.  
\* X\_FIELDCAT-NO\_OUT                 = 'X'.  
APPEND X\_FIELDCAT TO I\_FIELDCAT.  
CLEAR X\_FIELDCAT.  
X\_FIELDCAT-COL\_POS   = 9.  
X\_FIELDCAT-FIELDNAME             = 'MENGE'.  
X\_FIELDCAT-TABNAME                = 'I\_TAB'.  
X\_FIELDCAT-SELTEXT\_M             = 'Quantity'.  
X\_FIELDCAT-OUTPUTLEN             = 12.  
X\_FIELDCAT-DO\_SUM                    = 'X'.                       \* Summation is allowed for this field  
X\_FIELDCAT-REF\_TABNAME      = 'MSEG'.  
APPEND X\_FIELDCAT TO I\_FIELDCAT.  
CLEAR X\_FIELDCAT.  
 X\_FIELDCAT-COL\_POS   = 10.  
X\_FIELDCAT-FIELDNAME             = 'ICN'.  
X\_FIELDCAT-TABNAME                = 'I\_TAB'.  
X\_FIELDCAT-SELTEXT\_M             = ''.  
X\_FIELDCAT-OUTPUTLEN             = 2.  
X\_FIELDCAT-ICON                           = 'X'.  
\* X\_FIELDCAT-NO\_OUT                 = 'X'.  
APPEND X\_FIELDCAT TO I\_FIELDCAT.  
CLEAR X\_FIELDCAT.  
 X\_FIELDCAT-COL\_POS   = 11.  
X\_FIELDCAT-FIELDNAME             = 'MEINS'.  
X\_FIELDCAT-TABNAME                = 'I\_TAB'.  
X\_FIELDCAT-SELTEXT\_M             = 'Unit'.  
X\_FIELDCAT-OUTPUTLEN             = 5.  
X\_FIELDCAT-FIELDNAME             = 'MEINS'.  
APPEND X\_FIELDCAT TO I\_FIELDCAT.  
CLEAR X\_FIELDCAT.  
 X\_FIELDCAT-COL\_POS   = 12.  
X\_FIELDCAT-FIELDNAME             = 'DMBTR'.  
X\_FIELDCAT-TABNAME                = 'I\_TAB'.  
X\_FIELDCAT-SELTEXT\_M             = 'Local curr'.  
X\_FIELDCAT-OUTPUTLEN             = 12.  
X\_FIELDCAT-INTTYPE    = 'P'.  
X\_FIELDCAT-JUST                            = 'R'.  
X\_FIELDCAT-DO\_SUM    = 'X'.  
APPEND X\_FIELDCAT TO I\_FIELDCAT.  
CLEAR X\_FIELDCAT.  
 X\_FIELDCAT-COL\_POS   = 13.  
X\_FIELDCAT-FIELDNAME             = 'EXCPT'.  
X\_FIELDCAT-TABNAME                = 'I\_TAB'.  
X\_FIELDCAT-SELTEXT\_M             = ''.  
X\_FIELDCAT-OUTPUTLEN             = 3.  
APPEND X\_FIELDCAT TO I\_FIELDCAT.  
CLEAR X\_FIELDCAT.  
ENDFORM.                    "SUB\_PREPARE\_FIELDCATALOG

Step 4(Build up events table)

 The next step is to build an event table, which are used for firing both user commands and the system dependent events i.e. top of page, end of page etc.  
A list of possible events is populated into an event table (I\_EVENTS) when this table is passed from the function module REUSE\_ALV\_EVENT\_NAMES\_GET. The return table from this function module contains all the possible events.

The function module contains following import and export parameters.   
IMPORTING PARAMETERS: I\_LIST\_TYPE

This parameter has possible values from 0-4.

The parameter I\_LIST\_TYPE is of TYPE SLIS\_LIST\_TYPE and is DEFAULT 0 .   
 EXPORTING PARAMETERS:  I\_EVENTS table.  This table is of TYPE SLIS\_T\_EVENT and returns to the program the name of all the possible events.  
The table structure contains the fields:  
                I\_EVENTS-NAME: Name of the Callback event.  
I\_EVENTS-FORM: Name of the form routine that should be called in the calling program at the event.  
 Only events with a form routine name are processed.  
 The I\_EVENTS table returns with the following possible constants:  
 **1.** **Slis\_ev\_item\_data\_expand TYPE slis\_formname VALUE 'ITEM\_DATA\_EXPAND'.**   
Only relevant for hierarchical-sequential lists using the layout parameter IS\_LAYOUT-EXPAND\_FIELDNAME of the structure IS\_LAYOUT. Exit for passing item entries (ITEM table) for a header record that was expanded interactively by the user.**2.** **Slis\_ev\_reprep\_sel\_modify TYPE slis\_formname VALUE 'REPREP\_SEL\_MODIFY'.**  
RS\_SELFIELD-TABINDEX contains the header table index for which the item entries are to       be put in the global item output table (T\_OUTTAB\_SLAVE). The Callback is only called if ALV has no items for a header that is to be expanded.

RFLG\_ALL is passed with 'X' if the user shows all items. The application must ensure that    entries are not repeated in the item table.

RS\_SELFIELD is initial in this case.          **3.** **Slis\_ev\_caller\_exit\_at\_start TYPE slis\_formname VALUE 'CALLER\_EXIT'.**  
Is called at the beginning of the function module to make special settings. It is not usually used.           **4.** **Slis\_ev\_user\_command TYPE slis\_formname VALUE 'USER\_COMMAND'.**  
As this is a frequently-used Callback event, the form routine can also be passed        directly in the interface by passing the user command in the IMPORTING parameter           I\_CALLBACK\_USER\_COMMAND.  
**5.** **Slis\_ev\_top\_of\_page TYPE slis\_formname VALUE 'TOP\_OF\_PAGE'.**  
Equivalent to the list processing TOP-OF-PAGE event.  
**6.** **Slis\_ev\_top\_of\_coverpage TYPE slis\_formname VALUE 'TOP\_OF\_COVERPAGE'.**  
The selection information and list status are output together (if they exist) on a separate page by default  
**7.** **Slis\_ev\_end\_of\_coverpage TYPE slis\_formname VALUE 'END\_OF\_COVERPAGE'.**  
Analogously to TOP\_OF\_COVERPAGE the user can add other information

to the information output by ALV (selection information, list status) at this event.**8.** **Slis\_ev\_foreign\_top\_of\_page TYPE slis\_formname VALUE 'FOREIGN\_TOP\_OF\_PAGE'.**  
The Top-of-page event is always processed in ALV and is only passed to the caller via the Callback mechanism. This is still the case if the caller, e.g. by a user action, processes a branch list which was not formatted by ALV (e.g. a popup with additional information about the list record selected and displayed by ALV).

In this case, top-of-page cannot be formatted by ALV analogously to the basic list, it must be handled completely by the caller. The event top-of-page still occurs in ALV. When ALV notices a top-of-page which was not caused by an ALV output, the form routine in FOREIGN\_TOP\_OF\_PAGE is called. **9.** **Slis\_ev\_foreign\_end\_of\_page TYPE slis\_formname VALUE 'FOREIGN\_END\_OF\_PAGE'.**  
The event end-of-page is always processed in ALV and only passed to the caller via callback. This is still the case, e.g. when the caller processes a details list which was not formatted by ALV (e.g. a popup with further information about selected list records which were displayed by ALV).In this case, end-of-page cannot be formatted by ALV analogously to the basic list, it must be handled completely by the caller. The event end-of-page still occurs in ALV. When ALV notices an end-of-page that was not caused by an ALV output, the form routine in FOREIGN\_END\_OF\_PAGE is called.  
**10.** **Slis\_ev\_pf\_status\_set TYPE slis\_formname VALUE 'PF\_STATUS\_SET'.**  
If a user list status is to be set, it must be done in the form routine assigned to this event. The ALV function codes, which must not be active, are in the Parameter RT\_EXTAB. This table must be passed with the SET PF-STATUS command (with inactive user function codes as well, if necessary).

The STANDARD status of the function group SALV should be used as a template for a user-specific status. As this is a frequently used Callback event, its form routine can also be passed directly in the interface in the IMPORTING parameter I\_CALLBACK\_PF\_STATUS\_SET.**11.** **Slis\_ev\_list\_modify TYPE slis\_formname VALUE 'LIST\_MODIFY'.**  
LIST\_MODIFY USING R\_TABNAME TYPE SLIS\_TABNAME

                                         R\_INDEX LIKE SY-TABIX

                                         R\_INDEX\_ITEM LIKE SY-TABIX

                                         R\_INDEX\_SUM LIKE SY-TABIX.**12.** **Slis\_ev\_top\_of\_list TYPE slis\_formname VALUE 'TOP\_OF\_LIST'.**   
Information output at the start of the list\*     \*  
**13.** **Slis\_ev\_end\_of\_page TYPE slis\_formname VALUE 'END\_OF\_PAGE'.**  
Information output at the end of a page. This is only called for printing.**14.** **Slis\_ev\_end\_of\_list TYPE slis\_formname VALUE 'END\_OF\_LIST'.**  
Information output at the end of the list  
**15.** **Slis\_ev\_after\_line\_output TYPE slis\_formname VALUE 'AFTER\_LINE\_OUTPUT'.**  
Output information after each output line. Should only be used in justified cases because it costs a lot of performance.  
**16.** **Slis\_ev\_before\_line\_output TYPE slis\_formname VALUE** **'BEFORE\_LINE\_OUTPUT'.**  
Output information before each output line. Should only be used in justified cases because it costs a lot of performance.  
**17.** **Slis\_ev\_subtotal\_text TYPE slis\_formname VALUE** **'SUBTOTAL\_TEXT'.**  
This event table (I\_EVENTS) is now checked with the desired constants. If the desired constant is found, then the corresponding field for the FORM NAME is populated with the name of the routine containing the corresponding event. **Sample code :**  
 FORMNAME\_TOP\_OF\_PAGE TYPE SLIS\_FORMNAME VALUE 'TOP\_OF\_PAGE',  
FORMNAME\_END\_OF\_PAGE TYPE SLIS\_FORMNAME VALUE 'END\_OF\_PAGE', FORMNAME\_USER\_COMMAND TYPE SLIS\_FORMNAME VALUE 'USER\_COMMAND'.  
 DATA: L\_I\_EVENT TYPE SLIS\_ALV\_EVENT.  
   CALL FUNCTION 'REUSE\_ALV\_EVENTS\_GET'  
       EXPORTING  
           I\_LIST\_TYPE = 0  
       IMPORTING  
            ET\_EVENTS   = I\_EVENTS.  
   READ TABLE I\_EVENTS WITH KEY NAME = SLIS\_EV\_TOP\_OF\_PAGE INTO L\_I\_EVENT.  
  IF SY-SUBRC = 0.  
    MOVE FORMNAME\_TOP\_OF\_PAGE TO L\_I\_EVENT-FORM.  
    APPEND L\_I\_EVENT TO I\_EVENTS.  
  ENDIF.  
   READ TABLE I\_EVENTS WITH KEY NAME = SLIS\_EV\_END\_OF\_PAGE INTO L\_I\_EVENT.  
  IF SY-SUBRC = 0.  
    MOVE FORMNAME\_END\_OF\_PAGE TO L\_I\_EVENT-FORM.  
    APPEND L\_I\_EVENT TO I\_EVENTS.  
  ENDIF.  
  CLEAR L\_I\_EVENT.  
  READ TABLE I\_EVENTS WITH KEY NAME = SLIS\_EV\_USER\_COMMAND INTO L\_I\_EVENT.  
  IF SY-SUBRC = 0.  
    MOVE FORMNAME\_USER\_COMMAND TO L\_I\_EVENT-FORM.  
    APPEND L\_I\_EVENT TO I\_EVENTS.  
  ENDIF.  
 This will prepare the events table for the report.  
The report will contain three forms for the above events:  
1.        FORM TOP\_OF\_PAGE: This form will contain the top of page event for the report i.e. header etc  
Using the function module 'REUSE\_ALV\_COMMENTARY\_WRITE', the internal table containing the headings for top of page event can be passed to the list output. Also, any logo specific to the report can be passed to the function module.  
2.        FORM END\_OF\_PAGE: This form will contain the end of page event for the report i.e. footer etc  
3.        FORM USER\_COMMAND: This form will contain the desired user command i.e. pick/line selection

Step 5(Report Output list description)

 A layout is build for the report output list description USING the internal table declared above (I\_LAYOUT).  
Output list description structure.  
     The parameters are described under the following heads:  
-          Display options  
-          Exceptions  
-          Totals  
-          Interaction  
-          Detail screen  
-          Display variants (only for hierarchical-sequential lists)  
-          Color  
-          Other  
The layout table is of type slis\_layout\_alv\_spec and has the following fields:  
    
 Display options                                                   
1. **Colwidth\_optimize** (1) TYPE c: This parameter optimizes the length of the different columns in the output. The width of the different col. now depends on the max. Length of the data in the column.  
Value set: SPACE, 'X'  
'X' = optimizes the column width so that all contents are displayed completely.                     
 2. **No\_colhead** (1) TYPE c: This parameter suppresses the column headings                      
Value set: SPACE, 'X'.  
'X' = column headers are not output  
 3\*. No\_hotspot\*(1) TYPE c :  The headings of the report output are not output as hotspot.         
Value set: SPACE, 'X'.  
'X' = column headers are not output as hotspot  
 4\*. Zebra\*(1) TYPE c : The report is output in the striped pattern.               
Value set: SPACE, 'X'.  
'X' = striped pattern (e.g. for wide lists)  
 5. **No\_vline** (1) TYPE c: The report output contains columns only separated by space and no lines.  It is not relevant for: hierarchical-sequential lists and multiple-line lists.  
Value set: SPACE, 'X'.  
'X' = columns separated by SPACE  
 6. **No\_min\_linesize**(1) TYPE c : The report  line size is equal to the width of the list. It is not relevant for block lists.  
Value set: SPACE, 'X'.  
'X' = line size depends on list width  
'  ' = Line size is set to 80 or MIN\_LINESIZE (if > 0) .  
7. **Min\_linesize** LIKE sy-linsz: The report output contains a minimum possible length of line. If initial min\_linesize is set to 80 by default, then this parameter is used to customize it. The prerequisite for this is that the parameter no\_min\_linesize should be ' '.  
Value set: 0, 10 - 250  
If the list is wider, the format uses the list width (maximum 250 or MAX\_LINESIZE (if > 0)).  
 8. **Max\_linesize** LIKE sy-linsz: The default max. Linesize is 250. To change this default value, this parameter can interactively-define the maximum list width setting.  
Value set: 0, 80 - 1020  
 9. **Window\_titlebar** LIKE rsmpe-tittext: To set the titlebar on the report output.                              
10. **No\_uline\_hs**(1) TYPE c.  
                                    
Exceptions                                                   
11. **Lights\_fieldname** TYPE slis\_fieldname: Internal output table field containing the codes of exceptions to be output.  
  Output table field code: '1' = red traffic light  
                                                '2' = yellow traffic light  
                                                 '3' = green traffic light Fieldname for exception   
Value set: SPACE, internal output table field name.  
         
 12. **Lights\_tabname** TYPE slis\_tabname: Name of the internal output table that contains the field in the parameter LIGHTS\_FIELDNAME. If LIGHTS\_FIELDNAME is not empty, this field must also be filled for hierarchical-sequential lists. Only relevant for hierarchical-sequential lists.  
Value set: SPACE, internal output table name.  
 13. **Lights\_rollname** LIKE dfies-rollname: The documentation of this data element is displayed when you call F1 help for an exception column.  
Value set: SPACE, data element name.  
 14. **Lights\_condense** (1) TYPE c : If a list record is output with 'red traffic light', each  Subtotal that includes this record is also output with 'red traffic light'.  
Value set: SPACE, 'X'  
 'X' = the 'maximum' exception of the items in the subtotal is output at subtotal level.  
 Sums                                                         
15\*. No\_sumchoice\* (1) TYPE c: This parameter allows the choice for summing up Only by field catalog.  
 Value set: SPACE, 'X'  
 'X' = fields which are to be summed, passed by the calling program (FIELDCAT-DO\_SUM = 'X'). The user should not be able to change this value interactively.  
 16. **No\_totalline**(1) TYPE c : Removes the option of having totals after sub-totals.  
Value set: SPACE, 'X'  
'X' = no total record is to be output. Subtotals can still be calculated and output. The fields in the subtotals are flagged DO\_SUM = 'X' in the field list.  
 17. **No\_subchoice**(1) TYPE c : Does not allow the user to interactively change the field chosen for subtotals.  
Value set: SPACE, 'X'  
'X' = value whose change triggers subtotals, provided by the calling program. The user should not be able to change this value interactively.  
           
18. **No\_subtotals**(1) TYPE c : No subtotals possible            
Value set: SPACE, 'X'  
'X' = no subtotals.  
 19\*. Numc\_sum\*(1)  TYPE c : Totals only possible for NUMC-Fields.  
 20. **No\_unit\_splitting** TYPE c: No separate total lines by inh.units     
 21.**totals\_before\_items** TYPE c: Display totals before the items     
 22. **Totals\_only**(1) TYPE c :  Show only totals        
Value set: SPACE, 'X'  
'X' = only total records are output.  
 23. **Totals\_text**(60) TYPE c : Text for 1st col. in totals     
Value set: SPACE, string (max.60)  
' ' = The first column in the total record contains an appropriate number of '\*'s to indicate the total by default. If the first column is wide enough, the string 'Total' is output after the asterisks.  
'String' = The string passed is output after the total indicated by '\*', if the column is wide enough.  
 24. **Subtotals\_text**(60) TYPE c : Texts for subtotals  
Value set: SPACE, string (max.60)  
' ' = In the first column of subtotal records, the subtotal is indicated by an appropriate number of '\*' by default. If the first column is not a subtotal criterion, the string 'Total' is output after the asterisks, if the column is wide enough.  
'String' = the string passed is output after the subtotal indicated by '\*', if the column is wide enough and the first column is not a subtotal criterion. If it is a subtotal criterion, its value is repeated after the total, if the column is wide enough.  
 Interaction                                                        
25. **Box\_fieldname** TYPE slis\_fieldname: Fieldname for checkbox in the report output. If the list has checkboxes at the start of records (for selecting several records), this parameter contains the internal output table field name indicated by the checkbox selection column. The field is a checkbox at the start of list records without a list header.  
Value set: SPACE, internal output table field name  
      
26. **Box\_tabname** TYPE slis\_tabname: Name of the internal output table that contains the field in the parameter BOX\_FIELDNAME. If BOX\_FIELDNAME is not empty, this field must also be filled for hierarchical-sequential lists.  
Value set: SPACE, internal output table name.  
 27. **Box\_rollname** LIKE dd03p-rollname: rollname for checkbox            
 28. **Expand\_fieldname** TYPE slis\_fieldname: fieldname flag 'expand'. The user can show or hide the items by clicking on the folder symbol (hotspot). If the items for a header entry are only to be read by the calling program and passed to ALV when a header has been expanded interactively, this can be controlled via the CALLBACK event        'ITEM\_DATA\_EXPAND'.  
 29. **Hotspot\_fieldname** TYPE slis\_fieldname: Used to make the fieldname flag hotspot.  
 30. **No\_input**(1) TYPE c : The fields are only display fields.  
Value set: SPACE, 'X'  
'X' = all ready-for-input fields in a list are displayed as not ready-for-input. (Record selection checkboxes and fields which can be made ready-for-input via the field list parameter FIELDCAT-INPUT = 'X')  
 31. **F2code** LIKE sy-ucomm: To assign an ALV standard function code to double-click (F2), assign the function code to this parameter. Ex.: to assign the ALV standard function 'Detail' ('&ETA') to F2.  
=> LAYOUT-F2CODE = '&ETA'.  
Value set: SPACE, function code  
   
32. **Confirmation\_prompt**: confirm. Prompt when leaving         
Value set: SPACE, 'X'  
 'X' = if one of the functions 'Back (F03)', 'Exit (F15)' or 'Cancel (F12)' occurs, a confirmation prompt appears.  
 33. **Key\_hotspot**(1) TYPE c : The key fields are displayed as hotspot. The columns defined in the field catalog as key fields (FIELDCAT-KEY = 'X') are output as hotspots, i.e. clicking on a key column (highlighted in color in the list) calls the function under F2.  
Value set: SPACE, 'X'.  
 34. **Reprep**(1) TYPE c :  report report interface active.       
 35. **Group\_buttons** (1) TYPE c :  group-buttons for COL1 - COL5 . Group output fields via FIELDCAT-SP\_GROUP in the field list, and pass the group name to the list module in the interface parameter IT\_SPECIAL\_GROUPS.             
Value set: SPACE, 'X'.  
 36. **No\_keyfix**(1) TYPE c : Used to make the key fields scrollable.  
Value set: SPACE, 'X'.  
' ' = The key columns defined in the field catalog by FIELDCAT-KEY = 'X' are fixed in the list output. These columns do not scroll horizontally. The item table key columns are fixed in hierarchical-sequential lists. The header table key fields are not considered here.  
'X' = key columns not fixed  
 37. **Get\_selinfos**(1) TYPE c : To read selection screen.  
Value set: SPACE, 'X'.  
If the calling program is a report with an ABAP/4 selection screen, setting this parameter makes ALV read the selection screen again. If the selections are read successfully, a pushbutton, via which the user can call a popup which lists the report selections in a simple form, becomes active on the results list output by ALV.  
 38.   **group\_change\_edit**(1) TYPE c :  Settings by user for new group       
Value set: SPACE, 'X'  
'X' = the user can enter a format option for each sort criterion in the sort/subtotal popup, for the list format when this value changes (e.g. new page or underline).  
 39.   **No\_scrolling**(1) TYPE c : Does not allow scrolling of the list to the right.  
Value set: SPACE, 'X'.  
 40. **Expand\_all**(1) TYPE c : Expand all positions                     
 Detailed screen                                                      
40.   **Detail\_popup**(1) TYPE c : show detail in popup.  
Value set: SPACE, 'X'  
'  '  = List record detail display in full-screen mode, with top-of-page.  
'X'  = list record detail display in popup (without top-of-page).  
 41. **Detail\_initial\_lines**(1) TYPE c : show also initial lines     
Value set: SPACE, 'X'  
' ' = Only fields whose contents are not initial are output in the detail view.  
'X' = initial field contents are also output in detail.  
 41.   **detail\_titlebar**(30) type c : Titlebar for detail screen  
Value set: SPACE, string (max.30)  
' ' = ' Detail: Display' is output as the title of the detail window.  
'String' = the string passed is output as the title of the detail window.   
              
Display variants                                                     
42. **Header\_text** (20) TYPE c: Text for header button. Only relevant for hierarchical-sequential lists. You can toggle between display field and field list views via pushbuttons in the display variant definition popup for hierarchical-sequential lists. The views refer to the hierarchy level of the fields. This is technically a toggle between the header table and item table fields.     
Value set: SPACE, CHAR (20)  
' ' = The header table field pushbutton text is 'Header' by default.  
CHAR (20) = header table field pushbutton text.  
 43.**item\_text**(20) TYPE c : Text for item button. Only relevant for hierarchical-sequential lists. You can toggle the view between the display fields and the field list via pushbuttons in the display variant definition popup for hierarchical-sequential lists. The views refer to the hierarchy level of the fields. This is technically a toggle between the header table and item table fields.          
Value set: SPACE, CHAR (20)  
' ' = The pushbutton text for the item table fields is 'Item' by default.  
CHAR (20) = item table field pushbutton text.  
 44.**default\_ item**(1) TYPE c : Items as default. Only relevant for hierarchical-sequential lists.  
Value set: SPACE, 'X'  
' ' = The header table fields are displayed by default in the display variant definition popup. The user can switch to the item table fields interactively.  
'X' = the item table fields are displayed by default in the display variant Definition Popup.    The user can switch to the header table fields interactively.  
 Colour                                                         
45. **Info\_fieldname** TYPE slis\_fieldname: infofield for listoutput. A whole list record can be colored individually using a color code in a column of the internal output table for the record. Assign the name of the field containing the color code to this parameter.  
Value set: SPACE, internal output table field name  
The internal output table field must be of type CHAR(3).  
The code must have the following syntax:  'Cxy':  
        C = color (all codes must start with 'C')  
        X = color number ('1'-'9')  
        Y = bold ('0' = off, '1' = on)  
 46. **Coltab\_fieldname** TYPE slis\_fieldname: Cells can be colored individually using a color code which is contained in a column of the internal output table for the record containing the cell. Assign the name of the field to this parameter.          
 Others                                                  
47. **List\_append**(1) TYPE c :  no call screen. It is only useful to output block-lists without specifying the above modules if the number of list blocks exceeds, or may exceed, the maximum number specified in the block module documentation. These operations are not possible for user-defined block lists.  
 **Example code :**  
        I\_LAYOUT-f2code       = ws\_fcode.  
       I\_LAYOUT-zebra        = 'X'.  
 I\_LAYOUT-colwidth\_optimize = 'X'.  
 I\_LAYOUT-no\_keyfix = 'X'.  
 I\_LAYOUT-get\_selinfos = 'X'.  
 I\_LAYOUT-no\_hotspot = 'X'.  
       I\_LAYOUT-no\_input = 'X'.  
 I\_LAYOUT-hotspot\_fieldname = FIELDNAME.  
 I\_LAYOUT-no\_input          = 'X'.  
 I\_LAYOUT-no\_vline          = `X'.  
 I\_LAYOUT-no\_colhead        = ' '.  
 I\_LAYOUT-lights\_condense   = ` `.  
 I\_LAYOUT-totals\_text       = ` `.  
 I\_LAYOUT-subtotals\_text    = ` `.  
 I\_LAYOUT-totals\_only       = ` `.  
 I\_LAYOUT-key\_hotspot       = 'X'.  
 I\_LAYOUT-detail\_popup      = 'X'.  
 I\_LAYOUT-group\_change\_edit = 'X'.  
 I\_LAYOUT-GROUP\_BUTTONS     = 'X'.

Step 6(Pass Selection-screen Information)

This step is required to get the selection screen information in the report output.   
The prerequisite for this is to set the parameter LAYOUT-GET\_SELINFOS of the IMPORTING structure.

The parameters to be passed in the IS\_SEL\_HIDE table are:

o mode:              'R' = only the entries passed in the internal table IS\_SEL\_HIDE-T\_ENTRIES

Are output in the pop up. Selection info, which the list tool read in the selection screen (when called by a report with a selection screen), is replaced by the values passed.

'S' = the selection info which the list tool read in the selection screen of the calling report are modified by the entries in the table IS\_SEL\_HIDE-T\_ENTRIES.   
v      t\_entries:         Selection info table   
v      t\_entries-mode:   'A' = output the selection info for the current table record in the info popup.

§         'D' = do not output select option or SELNAME parameter selection info in   the popup.   
v      t\_entries-selname: (only used in t\_entries-mode = 'D') : Name of the select option or parameter. The following table fields are only used in t\_entries-mode = 'A'. They contain the selection information to be added.

-          t\_entries-field:  DDIC field name of the field for which selection information is to be output.

-          t\_entries-table:  DDIC table names of t\_entries-field.

-          t\_entries-stext:  Field name in info popup.

-          If t\_entries-field and t\_entries-table have been entered, this text is taken from DDIC.

-          t\_entries-valuf:  Selection condition 'from' value (external format)

-          t\_entries-valut:  Selection condition 'to' value (external format)

-          t\_entries-sign0:  (I)nclusive (E)xclusive

-          t\_entries-option:  All values of the select options Option field allowed.

Step 7(Deciding Sort Criteria)

The Table IT\_SORT is populated with the sort criteria for the different fields.  
The caller specifies the sorting and/or subtotaling of the basic list in the internal table IT\_SORT.  
 This internal table has the following fields:  
-          spos :  Sort sequence  
-          fieldname :  Internal output table field name  
-          tabname : Only relevant for hierarchical-sequential lists. Name of the internal output table.  
-          up : 'X' = sort in ascending order  
-          down : 'X' = sort in descending order  
-          subtot : 'X' = subtotal at group value change  
-          group : '\* ' = new page at group value change ,'UL' = underline at group value change

Step 8(Final Step)

The final step in the output of the report is the use of two ALV functions modules.1.       REUSE\_ALV\_FIELDCATALOG\_MERGE  
2.       REUSE\_ALV\_LIST\_DISPLAY  
 The first function module is used to pass the field catalog to the report output and merge it with the internal output table.  
   
FUNCTION reuse\_alv\_fieldcatalog\_merge.

\*"---------------------------------------------------------------------

\*"\*"Lokale Schnittstelle:

\*" IMPORTING

\*"            VALUE(I\_PROGRAM\_NAME)                        LIKE  SY-REPID OPTIONAL

\*"            VALUE(I\_INTERNAL\_TABNAME)                TYPE  SLIS\_TABNAME OPTIONAL

\*"            VALUE(I\_STRUCTURE\_NAME)     LIKE  DD02L-TABNAME OPTIONAL

\*"            VALUE(I\_CLIENT\_NEVER\_DISPLAY)           TYPE  SLIS\_CHAR\_1 default 'X'

\*"            VALUE(I\_INCLNAME)                                      LIKE  TRDIR-NAME OPTIONAL

\*"  CHANGING

\*"            VALUE(CT\_FIELDCAT) TYPE  SLIS\_T\_FIELDCAT\_ALV

\*"   EXCEPTIONS

\*"            INCONSISTENT\_INTERFACE

\*"            PROGRAM\_ERROR

\*"---------------------------------------------------------------------

Import parameters

I\_PROGRAM\_NAME: Program in which the internal output table is declared and populated

I\_INTERNAL\_TABNAME: Internal output table name

I\_STRUCTURE\_NAME: Structure name (structure, table, and view)

I\_CLIENT\_NEVER\_DISPL: Hide client fields default 'X'

I\_INCLNAME: Data declarations include name  CHANGING parameter

CT\_FIELDCAT: Field catalog with field descriptions   
The variant based on a program-internal table should only be used for rapid prototyping since the following restrictions apply:

1.   Performance is affected since the code of the table definition must always be read and interpreted at runtime.

2.   Dictionary reference are only considered if the keywords LIKE or  INCLUDE STRUCTURE (not TYPE) are used.   
 **Step 8(Display Internal Output Table)**  
   
The other function module is used to display the internal output table with the contents

FUNCTION reuse\_alv\_list\_display.

\*"----------------------------------------------------------------------

\*"\*"Lokale Schnittstelle:

" IMPORTING

\*"            VALUE(I\_INTERFACE\_CHECK) DEFAULT SPACE

\*"            VALUE(I\_CALLBACK\_PROGRAM) LIKE  SY-REPID DEFAULT SPACE

\*"            VALUE(I\_CALLBACK\_PF\_STATUS\_SET) TYPE  SLIS\_FORMNAME DEFAULT SPACE

\*"            VALUE(I\_CALLBACK\_USER\_COMMAND) TYPE  SLIS\_FORMNAME DEFAULT SPACE

\*"            VALUE(I\_STRUCTURE\_NAME) LIKE  DD02L-TABNAME OPTIONAL

\*"            VALUE(IS\_LAYOUT) TYPE  SLIS\_LAYOUT\_ALV OPTIONAL

\*"            VALUE(IT\_FIELDCAT) TYPE  SLIS\_T\_FIELDCAT\_ALV OPTIONAL

\*"            VALUE(IT\_EXCLUDING) TYPE  SLIS\_T\_EXTAB OPTIONAL

\*"            VALUE(IT\_SPECIAL\_GROUPS) TYPE  SLIS\_T\_SP\_GROUP\_ALV OPTIONAL

\*"            VALUE(IT\_SORT) TYPE  SLIS\_T\_SORTINFO\_ALV OPTIONAL

\*"            VALUE(IT\_FILTER) TYPE  SLIS\_T\_FILTER\_ALV OPTIONAL

\*"            VALUE(IS\_SEL\_HIDE) TYPE  SLIS\_SEL\_HIDE\_ALV OPTIONAL

\*"            VALUE(I\_DEFAULT) DEFAULT 'X'

\*"            VALUE(I\_SAVE) DEFAULT SPACE

\*"            VALUE(IS\_VARIANT) LIKE  DISVARIANT STRUCTURE  DISVARIANT DEFAULT SPACE

\*"            VALUE(IT\_EVENTS) TYPE  SLIS\_T\_EVENT OPTIONAL

\*"            VALUE(IT\_EVENT\_EXIT) TYPE  SLIS\_T\_EVENT\_EXIT OPTIONAL

\*"            VALUE(IS\_PRINT) TYPE  SLIS\_PRINT\_ALV OPTIONAL

\*"            VALUE(IS\_REPREP\_ID) TYPE  SLIS\_REPREP\_ID OPTIONAL

\*"            VALUE(I\_SCREEN\_START\_COLUMN) DEFAULT 0

\*"            VALUE(I\_SCREEN\_START\_LINE) DEFAULT 0

\*"            VALUE(I\_SCREEN\_END\_COLUMN) DEFAULT 0

\*"            VALUE(I\_SCREEN\_END\_LINE) DEFAULT 0

"       EXPORTING

\*"            VALUE(E\_EXIT\_CAUSED\_BY\_CALLER)

\*"            VALUE(ES\_EXIT\_CAUSED\_BY\_USER) TYPE  SLIS\_EXIT\_BY\_USER

"       TABLES

\*"            T\_OUTTAB

"       EXCEPTIONS

\*"            PROGRAM\_ERROR Import parameters

I\_INTERFACE\_CHECK:                     Interface consistency check log output.

I\_CALLBACK\_PROGRAM:              Name of the calling program

I\_CALLBACK\_PF\_STATUS\_SET: Set EXIT routine to status.

I\_CALLBACK\_USER\_COMMAND: EXIT routine for command handling

I\_STRUCTURE\_NAME:                    Internal output table structure name

IS\_LAYOUT:                                        List layout specifications

IT\_FIELDCAT:                                     Field catalog with field descriptions

IT\_EXCLUDING:                                 Table of inactive function codes

IT\_SPECIAL\_GROUPS:                      Grouping fields for column selection

IT\_SORT:                                              Sort criteria for first list display

IT\_FILTER:                                           Filter criteria for first list output

IS\_SEL\_HIDE:                                     Selection information modification

I\_DEFAULT:                                        Initial variant active/inactive logic

I\_SAVE:                                                 Variants can be saved

IS\_VARIANT:                                      Variant information

IT\_EVENTS:                                         Table of events to perform

IT\_EVENT\_EXIT:                                Standard fcode exit requests table

IS\_PRINT:                                             Print information

IS\_REPREP\_ID:                                    Initialization keys for Re/Re interface

I\_SCREEN\_START\_COLUMN:        Coordinates for list in dialog box

I\_SCREEN\_START\_LINE:                 Coordinates for list in dialog box

I\_SCREEN\_END\_COLUMN:             Coordinates for list in dialog box

I\_SCREEN\_END\_LINE:                      Coordinates for list in dialog box

IT\_EVENT\_EXIT:                                Standard fcode exit requests table

IS\_PRINT:                                             Print information

IS\_REPREP\_ID:                                    Initialization keys for Re/Re interface

I\_SCREEN\_START\_COLUMN:        Coordinates for list in dialog box

I\_SCREEN\_START\_LINE:                 Coordinates for list in dialog box

I\_SCREEN\_END\_COLUMN:             Coordinates for list in dialog box

I\_SCREEN\_END\_LINE:                      Coordinates for list in dialog box    Export parameters

E\_EXIT\_CAUSED\_BY\_CALLER:    Delete list in CALLBACK\_USER\_COMMAND

ES\_EXIT\_CAUSED\_BY\_USER:                       How the user left the list Tables

T\_OUTTAB:                                                         Table with data to be displayed ---mandatory

Documentation on function module: REUSE\_ALV\_GRID\_DISPLAY

The function module outputs an internal table with whatever structure in the form of a formatted single- or multi-line list.  **Process:**  
   
Ø       Passing an internal table with the set of information to be output

Ø       Passing a structure with general layout specifications for list   layout

Ø       Passing a field catalog in the form of an internal table   
  The field catalog describes the fields to be output in the list.

Notes

v      All interactions performed on the list refer directly to the internal output table. Sorting the list, for example, also involves a resorting of the internal output table passed (since it was passed by reference).   
v      An important factor determining the usability of the tool or of various generic functions (totals, subtotals) is the expected amount of data to be displayed.

**Parameters :**

-          I\_INTERFACE\_CHECK

-          I\_BYPASSING\_BUFFER

-          I\_BUFFER\_ACTIVE

-          I\_CALLBACK\_PROGRAM

-          I\_CALLBACK\_PF\_STATUS\_SET

-          I\_CALLBACK\_USER\_COMMAND

-          I\_CALLBACK\_TOP\_OF\_PAGE

-          I\_CALLBACK\_HTML\_TOP\_OF\_PAGE

-          I\_CALLBACK\_HTML\_END\_OF\_LIST

-          I\_STRUCTURE\_NAME

-          I\_BACKGROUND\_ID

-          I\_GRID\_TITLE

-          I\_GRID\_SETTINGS

-          IS\_LAYOUT

-          IT\_FIELDCAT

-          IT\_EXCLUDING

-          IT\_SPECIAL\_GROUPS

-          IT\_SORT

-          IT\_FILTER

-          IS\_SEL\_HIDE

-          I\_DEFAULT

-          I\_SAVE

-          IS\_VARIANT

-          IT\_EVENTS

-          IT\_EVENT\_EXIT

-          IS\_PRINT

-          IS\_REPREP\_ID

-          I\_SCREEN\_START\_COLUMN

-          I\_SCREEN\_START\_LINE

-          I\_SCREEN\_END\_COLUMN

-          I\_SCREEN\_END\_LINE

-          IT\_ALV\_GRAPHICS

-          IT\_ADD\_FIELDCAT

-          IT\_HYPERLINK

-          E\_EXIT\_CAUSED\_BY\_CALLER

-          ES\_EXIT\_CAUSED\_BY\_USER

**I\_CALLBACK\_PROGRAM:** **Name of the calling program**  
 Program from which the function module is called and that contains the exit routines. The program should always be a report, function group, module pool or form routine pool (it should not be an include).  
 **Caution:** Never pass SY-REPID directly at the interface. If field SY-REPID contains the desired program name, you must absolutely assign this name to an auxiliary variable and pass this variable to the   interface.

**I\_CALLBACK\_PF\_STATUS\_SET: Set EXIT runtime to status**

Passing an EXIT routine indicates to the ALV that the caller wants to  set a self-defined user status.  As a result, the default status of the ALV is not set.  The interface of the form routine specified must be defined as follows:   
**FORM** set\_pf\_status **USING** rt\_extab **TYPE** slis\_t\_extab Table RT\_EXTAB contains the function codes that would be hidden on the standard user interface.  
   
If the caller wants to use a self-defined user interface (for example, in order to provide additional list functions or use existing functions), we recommend that you copy standard status STANDARD from function group SALV and modify it accordingly. ALV standard function codes always start with '&'.   
See also the documentation on parameter  I\_CALLBACK\_USER\_COMMAND.   
If a self-defined user interface is used that includes function codes of the standard user interface, the function codes of the excluding table  passed should be taken into account.   
This means that the user status should generally be set as follows:

**SET PF-STATUS** user status **EXCLUDING** rt\_extab.

Application functions can be added to excluding table rt\_extab if they  are to be disabled.

The routine is called whenever the standard user interface would be set  with SET PF-STATUS.

**Default**

 If no EXIT routine is specified, the ALV sets a status that corresponds  to status STANDARD of function group SALV.

**I\_CALLBACK\_USER\_COMMAND: EXIT routine for command handling**

Description   
Passing an EXIT routine indicates to the ALV that the application wants  to respond to certain function codes.   
Generally, these are function codes that are unknown to the ALV (that  is, are not standard ALV functions) and that were defined and set by a user status.   
See also the documentation on parameter  I\_CALLBACK\_PF\_STATUS\_SET.   
The interface of the form routine specified must be defined as follows:   
**FORM** user\_command  **USING** r\_ucomm **LIKE** sy-ucomm rs\_selfield **TYPE** slis\_selfield.   
     Parameter R\_UCOMM contains the function code triggered.   
     Structure RS\_SELFIELD contains the following information:

o        **tabname** **:**      Name of the internal output table   
o        **tabindex :**       Index of the internal output table   
o        **fieldname:**      Field name   
o        **endsum** **:**     Cursor is located on the totals line   
o        **sumindex :**     If >0, the cursor is located on a subtotals line   
o        **value** **:**        Value of the field on the list   
o        **refresh** **:**       (Exporting) List should be set up again   
o        **col\_stable:**   (Exporting) Keep column position when list is set up    again   
o        **row\_stable:**   (Exporting) Keep row position when list is set up again   
o        **exit** **:**        (Exporting) Exit list (and ALV)   
o   **before\_action:** Call before standard action execution   
o   **after\_action :** Call after standard action execution, before list    setup   
o   **ignore\_multi :** Internal use   
o   **sel\_tab\_field:** Internal use   
The EXIT routine is called whenever a function unknown to the ALV is   triggered or if the routine call before/after the execution of a  standard function code has been defined by interface parameter

    IT\_EVENT\_EXIT.   
See also the documentation on parameter IT\_EVENT\_EXIT.   
The function code and the current cursor position are then passed on to the calling program through the EXIT routine.   
If the user has selected multiple rows by selecting checkboxes, the output table field designated as the checkbox contains the current state of the checkbox in the list. **I\_CALLBACK\_TOP\_OF\_PAGE**  
   
EXIT routine for handling TOP-OF-PAGE   
  Description   
    If the caller specifies an EXIT routine, this routine must have the   following form:   
FORM top\_of\_page.   
Module REUSE\_ALV\_COMMENTARY\_WRITE can then be called within the EXIT routine. This module is responsible for formatting the header information and also ensures online HTML formatting. In the print preview or in batch mode, the text passed is then output in the normal format.   
If module REUSE\_ALV\_COMMENTARY\_WRITE cannot be used, you must use two parameters instead. In I\_CALLBACK\_TOP\_OF\_PAGE you pass the form routine that is responsible for normal formatting in batch mode or in the print preview mode. The form routine that is responsible for online formatting, is passed in parameter I\_CALLBACK\_HTML\_TOP\_OF\_PAGE. If one of these parameters is not filled, top-of-page is not output in the respective mode.

**I\_CALLBACK\_HTML\_TOP\_OF\_PAGE**

EXIT routine for HTML TOP-OF-PAGE   
  Description   
    If function module REUSE\_ALV\_COMMENTARY\_WRITE is not used in the form   for CALLBACK\_TOP\_OF\_PAGE, the form routine must be passed in parameter

    I\_CALLBACK\_HTML\_TOP\_OF\_PAGE for the online mode. The form should then

    have the following format:   
    form top\_of\_page using cl\_dd type ref to cl\_dd\_document.   
    In the form, you can, for example, use methods of class CL\_DD\_DOCUMENT

    to display text in HTML format. **I\_CALLBACK\_HTML\_END\_OF\_LIST**  
   
EXIT routine for HTML END-OF-LIST   
    Description   
In this parameter, you can pass a form for the online handling of   end-of-list. The form must have the followiong format:

Form End\_of\_List using Cl\_dd type ref to cl\_dd\_document.**I\_STRUCTURE\_NAME**  
   
Internal output table structure name   
Description   
    If the internal output table is defined through an ABAP Dictionary   structure (INCLUDE STRUCTURE struct or LIKE struct), you can  automatically set up the field catalog by passing the structure name.   
    The field catalog is then internally set up for this structure as   follows:   
    o   All fields are on the list (NO\_OUT = SPACE) except fields of data   type CLNT.   
    o   The key fields of the Dictionary structure are adopted in the field  catalog as key fields.   
    o   References to unit fields stored in the Dictionary are adopted provided that the reference fields are contained in the structure.   
    o   If you additionally pass a field catalog as parameter, the structure information is merged with this field catalog.   
    For more information on how to set up the field catalog automatically,  see the documentation on function module  REUSE\_ALV\_FIELDCATALOG\_MERGE. **I\_GRID\_TITLE**  
   
Control title   
  Description :   Specifies the title of the control. This text is displayed above the   grid. **I\_GRID\_SETTINGS**  
   
Grid settings   
  Description   
If Top-of-Page or End-of-List are output online, these areas are   displayed in a splitter above or below the list. Using I\_GRID\_SETTINGS  you can reduce the default size to 0%. To do this, you use two fields:   
    COLL\_TOP\_P: Sets Top-of-Page to 0%   
    COLL\_END\_L: Sets End-of-List to 0% **IS\_LAYOUT**  
   
List layout specifications   
  Description   
    Structure for describing the list to be output.   
    The parameters are described and grouped based on the following

    categories:

* Display options
* Exceptions
* Totals
* Interaction
* Detail screen
* Color
* Other   
      Note the section on pre-defined settings.     **Display options**
* **colwidth\_optimize**     Value range: SPACE, 'X'  
       X' = Optimizes the column width to ensure that the content is displayed completely.
* **no\_colhead**Value range: SPACE, 'X'  
  'X' = Do not output column headings.
* **zebra**Value range: SPACE, 'X'  
  'X' = Striped pattern (for wide lists, for example)
* **no\_vline**Value range: SPACE, 'X'  
  'X' = Separate columns by SPACE.

**Exceptions**  
**lights\_fieldname**

Value range: SPACE, field name of the internal output table field of the internal output table that contains the coding of the exceptions to be output .   
Coding in the field of the output table:

'1' = red traffic light

'2' = yellow traffic light

'3' = green traffic light **lights\_tabname**

Value range: SPACE, table name of the internal output table Table name of the internal output table that contains the specified field in parameter LIGHTS\_FIELDNAME. **lights\_rollname**

 Value range: SPACE, data element name

The documentation defined for this data element is displayed when the F1 help for an exception column is called. **lights\_condense**

        Value range: SPACE, 'X'

        'X' = The system outputs the 'maximum' exception of the items included in the total at subtotal level.

Example: If a list row is output with a 'red traffic light', each subtotal included in this list row is also displayed with a 'red traffic light'. **Totals**

* **no\_sumchoice**

        Value range: SPACE, 'X'

        'X' = Value fields for which totals are calculated are communicated  by the calling program (FIELDCAT-DO\_SUM = 'X'). The user should not  be allowed to change this pre-defined setting interactively.

* **no\_totalline**

        Value range: SPACE, 'X'

        'X' = No overall totals line should be displayed. If required, subtotals can nevertheless be calculated and displayed. The fields which are used for calculating subtotals are to be marked with

        DO\_SUM = 'X' in the field catalog.-         **no\_subchoice**

Value range: SPACE, 'X'

'X' = Characteristics at whose control level subtotals should be calculated are communicated by the calling program.  The user should not be allowed to change this pre-defined setting interactively.   
See also the documentation on IMPORTING parameter IT\_SORT.

* **no\_subtotals**

        Value range: SPACE, 'X'

        'X' = Calculating subtotals should not be allowed.

* **totals\_only**

        Value range: SPACE, 'X'

        'X' = Data is output in compressed format only at totals line level.

Prerequisite: IMPORTING parameter IT\_SORT is filled accordingly with the sort criteria and the subtotals indicator.

See also the documentation on IMPORTING parameter IT\_SORT.

-         **totals\_text**

        Value range: SPACE, string (not more than 60)

        ' ' = In the first column, the standard system indicates the totals  level by displaying an adequate number of '\*' for the overall total.After the asterisks, the system displays the string 'total' provided

        that the column width of the first output column is large enough. If  the column width is not sufficient, only the asterisks are   displayed.

        'string' = After the totals level indicated visually by means of  '\*', the system displays the string passed provided that the column   width is sufficient.

* **subtotals\_text**

        Value range: SPACE, string (not more than 60) ' ' = In the first column, the standard system indicates the totals  level by displaying an adequate number of '\*' for the subtotals  line. After the asterisks, the system displays the string \*total\*  provided that the column width of the first output column is large enough and the characteristic of the first column is not a subtotal criterion. If the column width is not sufficient, only the asterisks  are displayed.   
'string' = After the totals level indicated visually by means of  '\*', the system displays the string passed provided that the column  width is sufficient and the characteristic of the first column is   not a subtotal criterion.   
 If the characteristic is a subtotal criterion, the system repeats  the characteristic value for which subtotals were calculated after the totals level provided that the column width is sufficient.

* **numc\_sum**

Value range: SPACE, 'X'

' ' = In the standard system, it is not possible to calculate totals     for NUMC fields.

'X' = It is generally possible to calculate totals for NUMC fields. If this indicator is set, you can use parameter FIELDCAT-NO\_SUM to control for each NUMC column whether totals can be calculated or  not. **Interaction**  
-         **box\_fieldname**

Value range: SPACE, field name of the internal output table.  If the list should have checkboxes at the beginning of each list row  (to allow the user to select multiple rows at once), you must fill   this parameter with the field name of the internal output table that  represents the selection column for selecting rows with the help of  checkboxes. The field is always displayed as a checkbox at the beginning of each list row without a list heading.

* **box\_tabname**

        Value range: SPACE, table name of the internal output table

* **f2code**

        Value range: SPACE, function code

        Meaning when the ALV standard interface is used:

        If you want to assign a standard ALV function code to a double-click (F2), you must assign this function code to this parameter.

    Example: You want to assign the standard ALV function 'Detail'

        ('&ETA') to F2.

          => LAYOUT-F2CODE = '&ETA'

        Meaning if a self-defined interface is used: **Case 1:**

 You leave the standard ALV function code for F2 '&IC1' in the copied  interface of the application. However, you want to have a function  executed with F2 that is not assigned to F2 in the interface

 (standard ALV function or application function). You must    communicate this function code to the ALV using parameter F2CODE.**Case 2:**

  You remove the standard ALV function code for F2 '&IC1' from the   interface of the application and use another function code instead  (standard ALV function or application function). You must

 communicate this function code to the ALV using parameter F2CODE.   
        This is required if you want to allow columns to be selected.

* **confirmation\_prompt**

Value range: SPACE, 'X'

'X' = If one of the functions 'Back(F03)', 'Exit(F15)' or   'Cancel(F12)' is triggered, the system asks the user if he wants to   leave the list.

* **key\_hotspot**

Value range: SPACE, 'X'

The columns defined as key fields in the field catalog  (FIELDCAT-KEY = 'X') are output as a hotspot. This means that    single-clicking a key field (highlighted in color in the list) triggers the function assigned to F2.

* **reprep**

       Value range: SPACE, 'X'

       'X' = Enable report/report interface

       Prerequisite: Application system (=> report RKKBRSTI  exists).

       The list module acts as a potential sender in the report/report   interface (interface initialization, if required).  
The calling report/module pool entered in I\_CALLBACK\_PROGRAM is  declared to the report/report interface as the sender report with type RT=Report.  If the sender report is assigned to receiver reports in table TRSTI, function code BEBx is set to active. ( x = function code class)       **Example:**

       If sender RKTFGS15 has a receiver assignment for Report Writer  report group 7KOI with function code class '3' (SAP setting), this receiver report group is called through the report/report interface

       at function code 'BEB3'. The selections passed to the report/report  interface are the report selections and the key information of the  selected row.   
 For more information on the report/report interface, see the documentation on function group 'RSTI'.**Detail screen**

* **detail\_initial\_lines**

         Value range: SPACE, 'X'

         ' ' = In the detail view, the system displays only fields whose content is not set to initial.

         'X' = Initial field contents are also displayed on the detail screen.

* **detail\_titlebar**

         Value range: SPACE, string (not more than 30)

         ' ' = The system displays 'Detail: Display' as the title of the detail screen.

         'string' = The system displays the string passed as the title of the detail screen.  **Color**

* **info\_fieldname**

         Value range: SPACE, field name of the internal output table.

You can color an entire list row individually by using a color code  that is set for each row in a column of the internal output table.  You must assign the field name of the field with the color code to

 this parameter.   
         The field of the internal output table must be of type CHAR(3).

         The code must comply with the following syntax:     'Cxy':

         C = Color (each code must begin with 'C')

         x = Color number ('1'-'9')

         y = Intensified ('0' = off, '1' = on)

         Note: The color of the key column is not affected. If you also want

         to color the key column at row or cell level, you can use complex

         coloring which is described below for parameter COLTAB\_FIELDNAME.

         For information on coloring columns, see the documentation on field   catalog parameter FIELDCAT-EMPHASIZE of IMPORTING parameter   IT\_FIELDCAT.

* **coltab\_fieldname**

Value range: SPACE, field name of the internal output table

You can color cells individually by using a color code that is set for the row of the cells in a column of the internal output table.  You must assign the field name of the field with the color code to this parameter.   
 The field of the internal output table must be of type   SLIS\_T\_SPECIALCOL\_ALV.

 Principle: The field for the color code is filled in the row in which the cells to be colored are located. The field then contains an internal table of the above structure that includes the field names of the cells to be colored with the color code. The cell coordinates are therefore derived from the row position in which the   color code is written and the column information contained in the color table. The row structure of the internal color table of type SLIS\_T\_SPECIALCOL\_ALV is as follows:   
        Farbtabelle-NAME      = Field name of cell to be colored

        Farbtabelle-COLOR-COL = Color number (1 - 9)

        Farbtabelle-COLOR-INT = Intensified (0 = off, 1 = on)

        Farbtabelle-COLOR-INV = Inverse (0 = off, 1 = on)

        Farbtabelle-NOKEYCOL  = Ignore key coloring ('X' = yes, ' ' = no)   
If parameter Farbtabelle-NAME is not filled, all color specifications refer to all fields. As a result, the entire row is colored.   
  Default   
In many cases, the default layout settings can be kept so that you frequently do not need to pass this structure with modified flags. **IT\_FIELDCAT**

 Field catalog with field descriptions  **Description**  
   
     Field catalog containing the field descriptions of the fields to be considered for the list output (usually, this is a subset of the fields  in the internal output table).   
     Basically, you need a field catalog for each list output that uses the  ALV.   
     The field catalog associated with the output table is generated in the code of the caller. You can generate the field catalog automatically or semi-automatically by calling function module

     REUSE\_ALV\_FIELDCATALOG\_MERGE.   
See also the documentation on function module : REUSE\_ALV\_FIELDCATALOG\_MERGE.

The minimum values required for the field catalog are documented in the 'Default' section. The caller can optionally use all other parameters to  assign non-standard output attributes to a field.

 It is only in the following cases that you are not required to generate  the field catalog and pass it explicitly:

* The structure of the internal table to be output corresponds to a   structure stored in the Data Dictionary and is referenced with LIKE   or INCLUDE STRUCTURE in the declaration of the internal table.
* All fields of this structure should be output in the list.
* The structure name is declared to the ALV using parameter I\_STRUCTURE\_NAME.

         See also the documentation on IMPORTNG parameter  I\_STRUCTURE\_NAME.     Positioning  
-         **col\_pos (column position)**  
        Value range: 0, 1 - 60  
        Only relevant if the relative column positions should by default not   be identical to the sequence of the fields in the field catalog.  
         The parameter determines the relative column position of the field   in the list output. The column sequence can interactively be changed by the user. If this parameter is set to its initial value for each field catalog entry, the columns are arranged in the order of the   fields in the field catalog.  
     Identification  
-         fieldname (field name)

        Value range: Field name of the internal output table (required parameter)

        Field name of the field in the internal output table that is   described by this field catalog entry.   
   Reference to the Data Dictionary **ref\_fieldname (field name of the reference field)**  
       Value range: SPACE, name of a field in the Data Dictionary Name of the referenced field in the Data Dictionary.  
       This parameter is only required if the field in the internal output table that is described by the current entry in the field catalog has a reference to the Data Dictionary (that is, is not a program field) and if the field name in the internal output table is not identical to the field name of the field in the Data Dictionary. If both field names are identical, it is sufficient to specify the Data   Dictionary structure or table in parameter FIELDCAT-REF\_TABNAME.  
-         ref\_tabname (field name of the reference table/structure)  
Value range: SPACE, name of a structure or table in the Data Dictionary Structure or table name of the referenced field in the Data Dictionary.  
This parameter is only required if the field in the internal output table that is described by the current entry in the field catalog has a reference to the Data Dictionary (that is, is not a program field).  
     Reference to fields with currency units/units of measure  
Each amount or quantity field of the internal output table whose decimal places are to be displayed with the proper unit in the list output, must comply with the following conventions:  
        -   The field is of data type QUAN or CURR (internal type P).  
           (Physically, the field must actually belong to this data type.  
           Overriding the physical data type with parameter FIELDCAT-DATATYPE has no effect.)  
          -   There is one field in the internal output table that contains the relevant unit.  
          -   There is also an entry for the unit field in the field catalog.  
             (If the unit should not be displayed as a column in the list and the user should not be able to show the unit interactively, for example, because the unit is always unique and therefore explicitly output by the caller in the list header, then you can assign parameter FIELDCAT-TECH  = 'X' to the field catalog entry for the unit field.  
          If a value field has a reference to a unit, this has the following effects when the list is output:  
          -   The decimal places are displayed with the proper unit.  
          -   An initial value field with reference to a non-initial unit is displayed as '0' (provided that FIELDCAT-NO\_ZERO is initial). If unit-specific totals are calculated for this value field, the unit is considered in the analysis of whether homogeneous units exist.  
          -   An initial value field with reference to an initial unit is displayed as SPACE. If unit-specific totals are calculated for this value field, the unit SPACE has no effect on the homogeneity of the unit if the value field is initial.  
          -   For non-initial value fields with initial unit, the unit SPACE is considered as a unit when unit-specific totals are calculated.  
    Reference to the currency unit  
-         cfieldname (field name of the currency unit field)  
       Value range: SPACE, name of a field of the output table Only relevant to amount columns with unit reference. Field name of the field in the internal output table that contains the currency unit for the amount field FIELDCAT-FIELDNAME.  
There must be a separate field catalog entry for the field specified in FIELDCAT-CFIELDNAME.  
    Reference to the unit of measure  
 -         qfieldname (field name of the unit of measure field)  
       Value range: SPACE, name of a field of the output table  
       Only relevant to quantity columns with unit reference.  
Field name of the field in the internal output table that contains the unit of measure for the amount field FIELDCAT-FIELDNAME.  
There must be a separate field catalog entry for the field specified in FIELDCAT-QFIELDNAME.  
    Output options for a column  
 -         **outputlen (column width)**  
Value range: 0 (initial), n  
For fields with reference to the Data Dictionary you can leave this parameter set to initial.  
For fields without reference to the Data Dictionary (program fields) you must set the parameter to the desired field output length on the list (column width).  
initial = The column width is derived from the output length of the referenced field (domain) in the Data Dictionary.  
       n = The column width is n characters.  
 -         key (key column)  
         Value range: SPACE, 'X'  
         'X' = Key field (colored output for key fields)  
         Key fields cannot be hidden interactively by the user.  
         Parameter FIELDCAT-NO\_OUT must be left set to initial.  
         For exceptions, see the documentation on parameter FIELDCAT-KEY\_SEL.  
 -         key\_sel (key column that can be hidden)  
         Value range: SPACE, 'X'  
         Only relevant if FIELDCAT-KEY = 'X'  
         Key field that can be hidden interactively by the user.  
         The user cannot interactively change the sequence of the key columns.  
         As with non-key fields, output control is performed using parameter FIELDCAT-NO\_OUT.  
 -         no\_out (field in the available fields list)  
         Value range: SPACE, 'X'  
         'X' = Field is not displayed on the current list.  
         The field is available to the user in the field list and can be interactively selected as a display field.  
         At row level, the user can use the detail function to display the content of these fields.  
         See also the documentation on the 'Detail screen' section of parameter IS\_LAYOUT.  
 -         tech (technical field)  
         Value range: SPACE, 'X'  
         'X' = Technical field  
         The field cannot be output on the list and cannot be shown interactively by the user.  
         The field may only be used in the field catalog (not in IT\_SORT, ...).  
 -         emphasize (highlight column in color)  
        Value range: SPACE, 'X' or 'Cxyz' (x:'1'-'9'; y,z: '0'=off '1'=on)  
        'X' = The column is highlighted in the default color for color highlighting.  
        'Cxyz' = The column is highlighted in the coded color:  
         -   C: Color (coding must start with C)  
         -   x: Color number  
         -   y: Intensified  
         -   z: Inverse  
 -         hotspot (column as hotspot)  
        Value range: SPACE, 'X'  
        'X' = The cells of the column are output as a hotspot.  
 -         do\_sum (calculate totals for column)  
        Value range: SPACE, 'X'  
        'X' = Totals are calculated for this field of the internal output table.  
        This function can also be used interactively by the user.  
 -         no\_sum (totals calculation not allowed)  
        Value range: SPACE, 'X'  
        'X' = No totals may be calculated for this field although the data type of the field allows totalling.  
     Formatting column contents  
 -         icon (icon)  
         Value range: SPACE, 'X'  
         'X' = The column contents are displayed as an icon.  
         The column contents of the internal output table must consist of valid icon strings ().  
         The caller should consider the problem of printing icons.  
 -         symbol (symbol)  
         Value range: SPACE, 'X'  
         'X' = The column contents are output as a symbol.  
         The column contents of the internal output table must consist of valid symbol characters.  
         The caller should consider the problem of printing symbols.  
Although symbols can generally be printed, they are not always shown correctly depending on the printer configuration.  
 -         just (justification)  
         Value range: SPACE, 'R', 'L', 'C'  
         Only relevant to fields of data type CHAR or NUMC  
         ' ' = Default justification according to data type  
         'R' = Right-justified output  
          'L' = Left-justified output  
         'C' = Centered output  
         The justification of the column header depends on the justification of the column contents. You cannot justify the column header independently of the column contents.  
 -         lzero (leading zeros)  
         Value range: SPACE, 'X'  
         Only relevant to fields of data type NUMC  
         By default, NUMC fields are output in the ALV right-justified without leading zeros.  
         'X' = Output with leading zeros  
 -         no\_sign (no +/- sign)  
         Value range: SPACE, 'X'  
         Only relevant to value fields  
         'X' = Value output without +/- signs.  
 -         no\_zero (suppress zeros)  
         Value range: SPACE, 'X'  
         Only relevant to value fields  
         'X' = Supress zeros  
 -         edit\_mask (field formatting)  
         Value range: SPACE, mask  
         mask = See documentation on the WRITE formatting option  
         USING EDIT MASK mask  
         Using mask = '== conv' you can force an output conversion conv.  
      Texts  
         The following parameters for texts are always required for program fields without reference to the Data Dictionary.  
         For fields with reference to the Data Dictionary, the texts are retrieved from the Data Dictionary. If you do not want this, you can fill the text parameters also for fields with reference to the Data Dictionary. If you do this, the corresponding texts from the Data Dictionary will be ignored.  
          If the user changes the column width interactively, the text with the appropriate length is always used as the column header.  
         If the user optimizes the column width interactively, both the field contents and the column headings are considered for the list output:  
        If all field contents are shorter than the shortest column heading, the column width is set based on the column heading.  
         The long field label is also used in the dialog boxes for defining the display variant, the sort order, and so on.  
 -         seltext\_l (long field label)  
-         seltext\_m (medium field label)  
-         seltext\_s (short field label)  
-         reptext\_ddic (heading)  
Same as the 'heading' for data element maintenance.  
When the list is output, the system does not necessarily retrieve the text stored here, but uses the text that fits best.  
-         ddictxt (determine text)  
        Value range: SPACE, 'L', 'M', 'S', 'R'  
        Using possible values 'L', 'M', 'S', 'R' you can predefine the keyword that should always be retrieved as the column header. If the column width is changed, the system tries to find a heading that fits the new output width.  
     Parameter for program fields without reference to the Data Dictionary  
        See also the parameter in the 'Texts' section.  
 -         datatype (data type)  
        Value range: SPACE, data type from the Data Dictionary (CHAR, NUMC,...)  
        Only relevant to fields without reference to the Data Dictionary. Data type of program field  
 -         ddic\_outputlen (external output length)  
        Value range: 0 (initial), n  
        Only relevant to fields without reference to the Data Dictionary whose output should nevertheless be modified using a conversion exit.  
        Prerequisites:  
        -   FIELDCAT-EDIT\_MASK = '==conv'  
            See also the documentation on parameter FIELDCAT-EDIT\_MASK  
         -   FIELDCAT-INTLEN = n  
            See documentation on parameter FIELDCAT-INTLEN  
        n = Field output length of the external display  
        The column width FIELDCAT-OUTPUTLEN must not be equivalent to the output length of the external display (FIELDCAT-DDIC\_OUTPUTLEN).  
     o   intlen (internal output length)  
        Value range: 0 (initial), n  
        Only relevant to fields without reference to the Data Dictionary whose output should nevertheless be modified using a conversion exit.  
        Prerequisites:  
         -   FIELDCAT-EDIT\_MASK = '==conv'  
            See also the documentation on parameter FIELDCAT-EDIT\_MASK  
         -   FIELDCAT-DDIC\_OUTPUTLEN = n  
            See also the documentation on parameter FIELDCAT-DDIC\_OUTPUTLEN  
        n = Field output length of the internal display  
    o   rollname (data element)  
       Value range: SPACE, name of a data element from the Data Dictionary  
       You can use this parameter to provide an F1 help for a program field without reference to the Data Dictionary or to provide an F1 help other than that of the Data Dictionary for a field with reference to the Data Dictionary.  
       When the F1 help is called for this field, the documentation for the data element assigned is displayed.  
       If, for fields with reference to the Data Dictionary,  
       FIELDCAT-ROLLNAME is initial, the documentation for the data element of the referenced field in the Data Dictionary is displayed.  
    Other  
   o   sp\_group (field group key)  
       Value range: SPACE, CHAR(1)  
       Key for grouping fields  
       You assign the key to the group description using parameter  
       IT\_SPECIAL\_GROUPS (see also the documentation on parameter IT\_SPECIAL\_GROUPS).  
       If you define such an assignment in the field catalog using  
       IT\_SPECIAL\_GROUPS, the fields in the field list of the display variant dialog box are grouped accordingly.  
    o   reprep (selection criterion of the report/report interface)  
       Value range: SPACE, 'X'  
       Prerequisites:  
        -   The report/report interface exists in the system.  
           (function group RSTI, table TRSTI)  
        -   Parameter LAYOUT-REPREP = 'X'  
           (See also the documentation on parameter  
           LAYOUT-REPREP of IMPORTING parameter IS\_LAYOUT)  
         'X' = If the report/report interface is called, the value of this field is passed as a selection criterion in the selected branch line of the interface.  
   
   Default   
     o   For internal table fields with reference to a field defined in the

         Data Dictionary, it is normally sufficient to make the following specifications:          -   fieldname  
         -   ref\_tabname  
      Note:  
      All fields not explicitly mentioned here are either not relevant in this context or are not released!  
      All other information is retrieved by the ALV from the Data Dictionary.  
      If you do not specify the relative column position (COL\_POS), the fields are output in the list in the order in which they were added to the field catalog.  
      REF\_FIELDNAME must only be specified if the field name of the internal table field is not identical to the field name of the referenced field in the Data Dictionary.  
      Priority rule:  
     Specifications made in the field catalog take priority over specifications in the Data Dictionary.

* For internal table fields without reference to the Data Dictionary (program fields), it is normally sufficient to make the following specifications:       -   fieldname  
        -   outputlen  
        -   datatype  (without data type, character is the default)  
        -   seltext\_s  
        -   seltext\_l  
    Note:  
  If you assign a data element to parameter ROLLNAME, you can, for example, implement an F1 help for program fields.  
   **IT\_EXCLUDING**  
  Table of inactive function codes  
      Description  
   Optional IMPORTING parameter IT\_EXCLUDING is an internal table. You must only fill this table if the caller uses the standard interface of the list tool but does not need certain interface functions and therefore wants to disable them.  
   In this case, you must enter the function codes of these standard functions into the table.  
  **IT\_SORT**  
   Sort criteria for first list display  
      Description  
        Using internal table IT\_SORT, the caller determines the sort order and/or the subtotalling of the basic list.  
        The following fields of this internal table must be filled:  
        o   spos :  Sort order  
        o   fieldname : Field name in the internal output table  
        o   up :  'X' = Sorted in ascending order  
        o   down :  'X' = Sorted in descending order  
        o   subtot : 'X' = Subtotals for control level changes  
        o   comp (INTERNAL USE ONLY)  
        o   expa  
           Perquisites:  
           IT\_SORT-SUBTOT = 'X', that is, the sort criterion is also the subtotals criterion.  
   If no complete breakdown but only a breakdown to totals level n that can be further expanded by the user should be displayed when the list is output for the first time, you must set the indicator for the totals level criterion of level n.  
   **IT\_FILTER**  
   Filter criteria for first list output  
    Description  
        Table with filter criteria  
   Using this parameter, you can pass on filter criteria resulting from explicitly loading a display variant in advance, for example, to list output.  
   This table should never be set up 'manually'.

**IS\_SEL\_HIDE**  
 Selection information modification  
    Description  
      This parameter is currently not supported!  
      Only relevant if layout parameter LAYOUT-GET\_SELINFOS of IMPORTING structure IS\_LAYOUT is set.  
      Complex type for modifying information displayed on the selection dialog box:  
      o   mode:              'R' = Only entries passed in internal table IS\_SEL\_HIDE-T\_ENTRIES are output on the dialog box. Selection information obtained by the list tool by reading the selection screen again (only if the report is called with selection screen) are replaced by the entries passed.  
                             'S' = The selection information obtained by the list tool by reading the selection screen of the calling report again, are modified by the entries of table IS\_SEL\_HIDE-T\_ENTRIES.  
      o   t\_entries:         Table with selection information  
      o   t\_entries-mode:   'A' = Display selection information of the current table row on the information dialog box.  
                           'D' = Do not display selection information of the Select option or of parameter SELNAME  on the dialog box.  
    o   t\_entries-selname: (required only if t\_entries-mode = 'D') name of Select option or parameter  
    The following table fields are only required if t\_entries-mode = 'A'.  
   They contain the selection information to be added.  
    o   t\_entries-field:  DDIC field name of the field for which selection information is to be displayed  
    o   t\_entries-table:  DDIC table name of t\_entries-field.  
    o   t\_entries-stext:  Field description on the information dialog box.  
              If t\_entries-field and t\_entries-table were filled, this text is taken from the DDIC.  
    o   t\_entries-valuf:  Selection condition from-value (external format)  
    o   t\_entries-valut:  Selection condition to-value (external format)  
    o   t\_entries-sign0:  (I)nclusive (E)xclusive  
    o   t\_entries-optio:  All values of the option field of the Select option are allowed.  
    The remaining fields are used internally and are irrelevant to the caller.

**I\_DEFAULT**  
 Initial variant active/inactive logic  
  Description  
      Initial variant maintenance active/inactive. .  
      Prerequisite:        Parameter IS\_VARIANT is filled accordingly.  
         See also the documentation on the IMPORTING parameter IS\_VARIANT.  
  Value Range  
      SPACE = Definition of initial variants not allowed  
      'X' =   Definition of initial variants allowed  
  Default :      SPACE  
**I\_SAVE**  
  Variants can be saved  
  Description  
      Controls the save mode  
         Prerequisite:  
         Parameter IS\_VARIANT is filled accordingly.  
         See also the documentation on IMPORTING parameter IS\_VARIANT.  
  Value Range  
      o   ' ' = Display variants cannot be saved  
         Defined display variants (such as delivered display variants) can be selected for presentation regardless of this indicator. However, changes cannot be saved.  
      o   'X' = Standard save mode  
         Display variants can be saved as standard display variants.  
         Saving display variants as user-specific is not possible.  
      o   'U' = User-specific save mode  
         Display variants can only be saved as user-specific.  
      o   'A' = Standard and user-specific save mode  
         Display variants can be saved both as user-specific and as standard variants. Users make their choice on the dialog box for saving the display variant.  
  Default : SPACE .  
 **IS\_VARIANT**  
  Variant information  
  Description  
      This structure is only relevant if display variants are to be saved and/or read.  
     Variant information including the name of the list variant that is used to output the list.  
      To allow display variants to be read within the ALV, you must specify the access path using fields REPORT (required field), HANDLE (optional field) and/or LOG\_GROUP (optional field).  
      If you also want to allow display variants to be saved, you must additionally fill parameter I\_SAVE accordingly.  
      See also the documentation on IMPORTING parameter I\_SAVE.  
      A variant is uniquely described through:  
      o   The program to which the variant is assigned (REPORT)  
      o   The handle (HANDLE), if, for example, multiple lists with different structures and data are called in a program (I\_CALLBACK\_PROGRAM).  
         The handle is a CHAR(4) field that must be uniquely defined and describes the assignment of the call to the current structure of the internal output table.  
         Example:  
         Depending on the user interaction, several types of lists can be output in program x.  
         The user should be able to define display variants for each type of list.  
You provide this function to the user by assigning a HANDLE to each list. If variants for the program and the handle are saved, the handle must not be modified any more.  
      o   The logical group, if, for example, the same list is created with different settings through various transactions (LOG\_GROUP).  
The logical group is a CHAR(4) that must be uniquely defined and specifies the assignment.  
         Example:  
Program x is called through transactions T1 and T2. Depending on the transaction code, the fields available to the user through the field catalog differ in their assignment to different logical groups.  
If variants for the program and the logical group are saved, the logical group must not be modified any more.  
      o   The user name, if user-specific variants are saved (USERNAME).  
         You do not have to fill this parameter manually since the variant name is unique.  
      o   The variant name (VARIANT).  
         You only have to fill this parameter if this structure is to be used to read a concrete variant and the list is to be output with this variant.  
  Value Range  
      To call a list with a variant, you must specify the above fields.  
  Default  
 If the structure is initial, but saving is active (I\_SAVE is not initial), then IS\_VARIANT-REPORT = I\_CALLBACK\_PROGRAM is set.  
 For a possible entries help for variants, function module REUSE\_ALV\_VARIANT\_F4 is available.

|  |  |
| --- | --- |
| Only required if the list is to be output in a dialog box. | **I\_SCREEN\_START\_COLUMN** **I\_SCREEN\_START\_LINE** **I\_SCREEN\_END\_COLUMN**  **I\_SCREEN\_END\_LINE** |

 **Example Code**  
 WS\_REPNAME = SY-REPID.  
   CALL FUNCTION 'REUSE\_ALV\_FIELDCATALOG\_MERGE'  
       EXPORTING  
            I\_PROGRAM\_NAME     = WS\_REPNAME  
            I\_INTERNAL\_TABNAME = Internal output table field name  
            I\_INCLNAME         = WS\_REPNAME  
       CHANGING  
            CT\_FIELDCAT        = I\_FIELDTAB.  
   IF SY-SUBRC <> 0.  
    WRITE: 'SY-SUBRC: ', SY-SUBRC, 'REUSE\_ALV\_FIELDCATALOG\_MERGE'.  
  ENDIF.  
 CALL FUNCTION 'REUSE\_ALV\_LIST\_DISPLAY'  
       EXPORTING  
I\_CALLBACK\_PROGRAM = WS\_REPNAME  
I\_STRUCTURE\_NAME     = Internal output table field name  
IS\_LAYOUT        = I\_LAYOUT  
IT\_FIELDCAT                      = I\_FIELDTAB  
I\_DEFAULT                         = 'A'  
I\_SAVE                                  = 'A'  
IS\_VARIANT                       = 'X'  
IT\_EVENTS                          = I\_EVENTS[]  
IT\_SORT                               = I\_SORT  
                IS\_SEL\_HIDE        = I\_SELINFO  
       TABLES  
            T\_OUTTAB           = Internal output table field name.  
   
  IF SY-SUBRC <> 0.  
    WRITE: 'SY-SUBRC: ', SY-SUBRC, 'REUSE\_ALV\_LIST\_DISPLAY'.  
  ENDIF  
   
Using other function module 'REUSE\_ALV\_GRID\_DISPLAY' can help us get list output in the form of a grid and also attach logos to the report output.

Sample code

1 **Simple list output**:  
 REPORT Y\_DEMO\_ALV NO STANDARD PAGE HEADING.  
\* Data to be displayed  
DATA: I\_SFLIGHT TYPE TABLE OF SFLIGHT.  
 \*---------------------------------------------------------------------\*  
\* Selection  
  SELECT \* FROM SFLIGHT INTO TABLE I\_SFLIGHT.  
 \* Call ABAP List Viewer (ALV)  
  CALL FUNCTION 'REUSE\_ALV\_LIST\_DISPLAY'  
       EXPORTING  
            I\_STRUCTURE\_NAME   = 'SFLIGHT'  
       TABLES  
            T\_OUTTAB           = I\_SFLIGHT.  
 2.**Simple grid output:**  
 REPORT Y\_DEMO\_ALV\_1.  
\*  
\* Data to be displayed  
DATA: I\_SFLIGHT TYPE TABLE OF SFLIGHT.  
\*---------------------------------------------------------------------\*  
 \* Selection  
  SELECT \* FROM SFLIGHT INTO TABLE I\_SFLIGHT.  
 \* Call ABAP List Viewer (ALV)  
  CALL FUNCTION 'REUSE\_ALV\_GRID\_DISPLAY'  
       EXPORTING  
            I\_STRUCTURE\_NAME   = 'SFLIGHT'  
       TABLES  
            T\_OUTTAB           = I\_SFLIGHT.  
 **3.** **Demo for 'REUSE\_ALV\_POPUP\_TO\_SELECT'**  
 REPORT y\_demo\_alv\_3.  
 TYPE-POOLS: slis.  
 DATA: BEGIN OF i\_outtab OCCURS 0.  
        INCLUDE STRUCTURE sflight.  
DATA:   w\_chk TYPE c.                  "For multiple selection  
DATA: END OF i\_outtab.  
 \*       I\_OUTTAB TYPE SFLIGHT OCCURS 0,  
 DATA:   i\_private TYPE slis\_data\_caller\_exit,  
i\_selfield TYPE slis\_selfield,  
W\_exit(1) TYPE c.  
 PARAMETERS: p\_title TYPE sy-title.  
 START-OF-SELECTION.  
   SELECT \* FROM sflight INTO TABLE i\_outtab.  
   CALL FUNCTION 'REUSE\_ALV\_POPUP\_TO\_SELECT'  
       EXPORTING  
            i\_title                                                          = p\_title  
            i\_selection                                                 = 'X'  
            i\_zebra                                                       = 'X'  
\*           I\_SCREEN\_START\_COLUMN           = 0  
\*           I\_SCREEN\_START\_LINE                    = 0  
\*           I\_SCREEN\_END\_COLUMN                 = 0  
\*           I\_SCREEN\_END\_LINE          = 0  
            i\_checkbox\_fieldname                             = 'W\_CHK'  
\*           I\_LINEMARK\_FIELDNAME               =  
\*           I\_SCROLL\_TO\_SEL\_LINE                   = 'X'  
            i\_tabname                                  = 'I\_OUTTAB'  
            i\_structure\_name                                     = 'SFLIGHT'  
\*           IT\_FIELDCAT             =  
\*           IT\_EXCLUDING            =  
\*           I\_CALLBACK\_PROGRAM      =  
\*           I\_CALLBACK\_USER\_COMMAND =  
\*            IS\_PRIVATE             = I\_PRIVATE  
     IMPORTING  
            es\_selfield             = i\_selfield  
            e\_exit                  = w\_exit  
       TABLES  
            t\_outtab                = i\_outtab  
       EXCEPTIONS  
            program\_error           = 1  
            OTHERS                  = 2.  
   IF sy-subrc <> 0.  
\*    MESSAGE i000(0k) WITH sy-subrc.  
  ENDIF.  
 \*\*\*\*\*the internal table is modified with a cross sign for marking the rows selected  
   LOOP AT i\_outtab WHERE w\_chk = 'X'.  
    WRITE: /  i\_outtab-carrid, i\_outtab-price.  
  ENDLOOP.  
 

**5.** **Whole Sample Code**  
 REPORT YSUBALV.  
 \*----------~~Declaration of variables~~-----------------\*  
TYPE-POOLS: SLIS.  
\* To pass name of the report in function module for ALV  
DATA: V\_REPID LIKE SY-REPID .  
\* To pass the overall structure of the ALV report  
DATA:  STRUCT\_LAYOUT    TYPE SLIS\_LAYOUT\_ALV.  
DATA:  STRUCT\_LAYOUT1   TYPE SLIS\_LAYOUT\_ALV.  
 \* Internal table to capture various events in ALV  
DATA : I\_EVENTS TYPE SLIS\_T\_EVENT.  
 \* Table for catalog of the fields to be displayed  
DATA: I\_FIELDCAT TYPE  SLIS\_T\_FIELDCAT\_ALV.  
data : x\_fieldcat TYPE SLIS\_FIELDCAT\_ALV.  
 DATA:  I\_FIELDCAT1 TYPE  SLIS\_T\_FIELDCAT\_ALV.  
data : x\_fieldcat1 TYPE SLIS\_FIELDCAT\_ALV.  
 \* Internal table to mention the sort sequence  
DATA : IT\_SORT TYPE SLIS\_T\_SORTINFO\_ALV.  
DATA : X\_SORT TYPE SLIS\_SORTINFO\_ALV.  
 \* Internal table to display top of page  
data : i\_list\_top\_of\_page type slis\_t\_listheader.  
 \* Structure to display variants  
 data : i\_variant like disvariant,  
        i\_variant1 like disvariant.  
 \* Internal table to pass data  
DATA: BEGIN OF I\_TAB OCCURS 0,  
        mblnr like mseg-mblnr ,  
        matnr like mseg-matnr,  
        maktg like makt-maktg ,  
        charg like mseg-charg ,  
        werks like mseg-werks,  
        lgort like mseg-lgort,  
        menge like mseg-menge ,  
        meins like mseg-meins ,  
        dmbtr like mseg-dmbtr,  
        ebeln like mseg-ebeln,  
        icn(4) type c ,  
        sym(4) type c ,  
        excpt(2) type c ,  
        box(1),  
    END OF I\_TAB.  
 \*EJECT  
 DATA : begin of i\_doc occurs 0 .  
        INCLUDE STRUCTURE MSEG.  
 DATA : end of i\_doc.  
 \* --------~~End of Data Declaration~~--------------\*  
  PARAMETERS : P\_VAR LIKE DISVARIANT-VARIANT.  
 initialization.  
v\_repid = sy-repid.  
\* Display default variant  
 PERFORM SUB\_VARIANT\_INIT.  
 AT SELECTION-SCREEN ON P\_VAR.  
\* Once the user has entered variant, check about its existence  
 PERFORM SUB\_CHECK\_PVAR.  
 AT SELECTION-SCREEN ON VALUE-REQUEST FOR P\_VAR.  
\* Display a list of various variants of the report when the user presses F4 key in the variant field  
 PERFORM SUB\_VARIANT\_F4.  
 START-OF-SELECTION.  
\* Prepare field catalog for the main report. State the name of the field , name of internal table , various formatting options etc  
 PERFORM SUB\_PREPARE\_FIELDCATALOG.  
\* Fetches records from database into table i\_tab to be passed as export  
\* parameter t\_outtab in function module : REUSE\_ALV\_GRID\_DISPLAY  
 PERFORM SUB\_SELECT\_RECORD.  
\* Populate stat and icon columns of internal table i\_tab with specific  
\* columns and symbols based on some logic for quantity and value fields.  
 PERFORM SUB\_MODIFY\_RECORDS.  
\* Defines the overall layout of the report  
 PERFORM SUB\_DETERMINE\_ALV\_LAYOUT.  
\* Defines the sort sequence of the report  
 PERFORM SUB\_DETERMINE\_SORT\_SEQUENCE.  
\* Defines the event table  
 PERFORM SUB\_EVENTTAB\_BUILD USING I\_EVENTS.  
\* Things to be written at the top of the page  
 PERFORM SUB\_COMMENT\_BUILD USING i\_list\_top\_of\_page.  
\* Display the ALV list  
 PERFORM SUB\_SHOW\_ALV\_LIST.  
\*  struct\_layout-hotspot\_fieldname  = 'X'.  
 AT LINE-SELECTION.  
 PERFORM SUB\_HOTSPOT.  
 \*&---------------------------------------------------------------------\*  
\*&      Form  SUB\_VARIANT\_INIT  
\*&---------------------------------------------------------------------\*  
\* Display default variant  
\*----------------------------------------------------------------------\*  
form SUB\_VARIANT\_INIT.  
I\_VARIANT1-REPORT = SY-REPID.  
 \* Search default variant for the report  
  CALL FUNCTION 'REUSE\_ALV\_VARIANT\_DEFAULT\_GET'  
       EXPORTING  
            i\_save     = 'A'  
       CHANGING  
            cs\_variant = i\_variant1  
       EXCEPTIONS  
            not\_found  = 2.  
\* If default variant is found , use it as default.  
\* Else , use the variant LAYOUT1.  
  IF sy-subrc = 0.  
    p\_var = i\_variant1-variant.  
  ELSE.  
    p\_var = 'LAYOUT1'.  
  ENDIF.  
endform.                    " SUB\_VARIANT\_INIT  
 \*&---------------------------------------------------------------------\*  
\*&      Form  SUB\_CHECK\_PVAR  
\*&---------------------------------------------------------------------\*  
\* Once the user has entered variant, check about its existence  
\*----------------------------------------------------------------------\*  
FORM SUB\_CHECK\_PVAR.  
\* If the name of the variable is not blank, check about its existence  
if not p\_var is initial.  
  clear i\_variant.  
  i\_variant-report = sy-repid.  
  i\_variant-variant = p\_var.  
  CALL FUNCTION 'REUSE\_ALV\_VARIANT\_EXISTENCE'  
         EXPORTING  
              I\_SAVE     = 'A'  
         CHANGING  
              CS\_VARIANT = I\_VARIANT.  
\* If no such variant found , flash error message  
     if sy-subrc ne 0 .  
      message e398(00) with 'No such variant exists'.  
     else.  
\* If variant exists , use the variant name to populate structure  
\* I\_VARIANT1 which will be used for export parameter : IS\_VARIANT  
\* in the function module : REUSE\_ALV\_GRID\_DISPLAY  
       clear i\_variant1.  
       move p\_var to i\_variant1-variant.  
       move sy-repid to i\_variant1-report.  
     endif.  
 else.  
   clear i\_variant.  
 endif.  
ENDFORM.                    " SUB\_CHECK\_PVAR  
 \*&---------------------------------------------------------------------\*  
\*&      Form  SUB\_PREPARE\_FIELDCATALOG  
\*&---------------------------------------------------------------------\*  
\* Prepare field catalog for the main report. State the name of  
\* the field , name of internal table , various formatting options etc  
\*----------------------------------------------------------------------\*  
form SUB\_PREPARE\_FIELDCATALOG.  
\* First field to appear in ALV list  
 X\_FIELDCAT-COL\_POS = 1.  
\* Name of the internal table field  
  X\_FIELDCAT-FIELDNAME = 'SYM'.  
\* Name of the internal table  
  X\_FIELDCAT-TABNAME = 'I\_TAB'.  
\* Heading for the field  
  X\_FIELDCAT-SELTEXT\_M = 'Stat'.  
\* The field is going to contain a symbol  
  x\_fieldcat-symbol = 'X'.  
\* Append the specifications to the internal table for field catalog.  
  append X\_fieldcat TO I\_FIELDCAT.  
  clear x\_fieldcat.  
 \* Second field to appear in ALV list  
  X\_FIELDCAT-COL\_POS = 2.  
\* Name of the field in the internal table  
  X\_FIELDCAT-FIELDNAME = 'MATNR'.  
\* Name of the internal table  
  X\_FIELDCAT-TABNAME = 'I\_TAB'.  
\* Heading for the column  
  X\_FIELDCAT-SELTEXT\_M = 'MatItem'.  
\* It is going to be the key field.The color for this field is going to  
\* be different  
  X\_fieldcat-key = 'X'.  
  X\_fieldcat-key\_sel = 'X'.  
\* Single click on the field will trigger double click event.Also, a hand  
\* will appear when the cursor navigates to the field  
  X\_fieldcat-hotspot = 'X'.  
\* The column and those left to it will not scroll  
  X\_fieldcat-fix\_column = 'X'.  
\* F1 help will come as it is referenced to DDIC table  
  x\_fieldcat-ref\_tabname = 'MSEG'.  
   append X\_fieldcat TO I\_FIELDCAT.  
  clear x\_fieldcat.  
  X\_FIELDCAT-COL\_POS = 3.  
  X\_FIELDCAT-FIELDNAME = 'MAKTG'.  
  X\_FIELDCAT-TABNAME = 'I\_TAB'.  
  X\_FIELDCAT-SELTEXT\_M = 'Description'.  
\*  X\_FIELDCAT-OUTPUTLEN = 50.  
  x\_fieldcat-hotspot = space.  
\* The field is centre(C for centre, R and L for left and  
\* right) justified  
  x\_fieldcat-just = 'C'.  
  x\_fieldcat-key = 'X'.  
  x\_fieldcat-fix\_column = 'X'.  
\*  X\_fieldcat-no\_out = 'X'.  
   X\_fieldcat-fix\_column = 'X'.  
  append X\_fieldcat TO I\_FIELDCAT.  
  clear x\_fieldcat.  
  X\_FIELDCAT-COL\_POS = 4.  
  X\_FIELDCAT-FIELDNAME = 'CHARG'.  
  X\_FIELDCAT-TABNAME = 'I\_TAB'.  
  X\_FIELDCAT-SELTEXT\_M = 'Batch'.  
\*  X\_FIELDCAT-OUTPUTLEN = 10.  
  x\_fieldcat-hotspot = space.  
  append X\_fieldcat TO I\_FIELDCAT.  
  clear x\_fieldcat.  
    X\_FIELDCAT-COL\_POS = 5.  
  X\_FIELDCAT-FIELDNAME = 'EBELN'.  
  X\_FIELDCAT-TABNAME = 'I\_TAB'.  
  X\_FIELDCAT-SELTEXT\_M = 'Purchase Order'.  
\*  X\_FIELDCAT-OUTPUTLEN = 14.  
\* The field will be colored differently(Cxyz)  
  x\_fieldcat-emphasize = 'C511'.  
\* Initially the field will be hidden  
  x\_fieldcat-no\_out = 'X'.  
  append X\_fieldcat TO I\_FIELDCAT.  
  clear x\_fieldcat.  
   X\_FIELDCAT-COL\_POS = 6.  
  X\_FIELDCAT-FIELDNAME = 'MBLNR'.  
  X\_FIELDCAT-TABNAME = 'I\_TAB'.  
  X\_FIELDCAT-SELTEXT\_M = 'Document no'.  
\*  X\_FIELDCAT-OUTPUTLEN = 14.  
  x\_fieldcat-emphasize = 'C711'.  
  x\_fieldcat-no\_out = 'X'.  
  append X\_fieldcat TO I\_FIELDCAT.  
  clear x\_fieldcat.  
   X\_FIELDCAT-COL\_POS = 7.  
  X\_FIELDCAT-FIELDNAME = 'WERKS'.  
  X\_FIELDCAT-TABNAME = 'I\_TAB'.  
  X\_FIELDCAT-SELTEXT\_M = 'Plant'.  
\*  X\_FIELDCAT-OUTPUTLEN = 5.  
  x\_fieldcat-emphasize = 'C310'.  
  append X\_fieldcat TO I\_FIELDCAT.  
 clear x\_fieldcat.  
    X\_FIELDCAT-COL\_POS = 8.  
  X\_FIELDCAT-FIELDNAME = 'LGORT'.  
  X\_FIELDCAT-TABNAME = 'I\_TAB'.  
  X\_FIELDCAT-SELTEXT\_M = 'St.Loc'.  
\*  X\_FIELDCAT-OUTPUTLEN = 7.  
\*  X\_fieldcat-no\_out = 'X'.  
  append X\_fieldcat TO I\_FIELDCAT.  
  clear x\_fieldcat.  
   X\_FIELDCAT-COL\_POS = 9.  
  X\_FIELDCAT-FIELDNAME = 'MENGE'.  
  X\_FIELDCAT-TABNAME = 'I\_TAB'.  
  X\_FIELDCAT-SELTEXT\_M = 'Quantity'.  
   X\_FIELDCAT-OUTPUTLEN = 12.  
\* Summation is allowed for this field  
  x\_fieldcat-do\_sum = 'X'.  
  X\_FIELDCAT-ref\_tabname = 'MSEG'.  
  append X\_fieldcat TO I\_FIELDCAT.  
  clear x\_fieldcat.  
   X\_FIELDCAT-COL\_POS = 10.  
  X\_FIELDCAT-FIELDNAME = 'ICN'.  
  X\_FIELDCAT-TABNAME = 'I\_TAB'.  
  X\_FIELDCAT-SELTEXT\_M = ''.  
   X\_FIELDCAT-OUTPUTLEN = 2.  
   x\_fieldcat-icon = 'X'.  
\*  X\_fieldcat-no\_out = 'X'.  
  append X\_fieldcat TO I\_FIELDCAT.  
  clear x\_fieldcat.  
  X\_FIELDCAT-COL\_POS = 11.  
  X\_FIELDCAT-FIELDNAME = 'MEINS'.  
  X\_FIELDCAT-TABNAME = 'I\_TAB'.  
  X\_FIELDCAT-SELTEXT\_M = 'Unit'.  
   X\_FIELDCAT-OUTPUTLEN = 5.  
  x\_fieldcat-qfieldname = 'MEINS'.  
  append X\_fieldcat TO I\_FIELDCAT.  
  clear x\_fieldcat.  
   X\_FIELDCAT-COL\_POS = 12.  
  X\_FIELDCAT-FIELDNAME = 'DMBTR'.  
  X\_FIELDCAT-TABNAME = 'I\_TAB'.  
  X\_FIELDCAT-SELTEXT\_M = 'Local curr'.  
   X\_FIELDCAT-OUTPUTLEN = 12.  
  x\_fieldcat-INTTYPE = 'P'.  
  x\_fieldcat-just = 'R'.  
  x\_fieldcat-do\_sum = 'X'.  
  append X\_fieldcat TO I\_FIELDCAT.  
  clear x\_fieldcat.  
 X\_FIELDCAT-COL\_POS = 13.  
  X\_FIELDCAT-FIELDNAME = 'EXCPT'.  
  X\_FIELDCAT-TABNAME = 'I\_TAB'.  
  X\_FIELDCAT-SELTEXT\_M = ''.  
  X\_FIELDCAT-OUTPUTLEN = 3.  
 append X\_fieldcat TO I\_FIELDCAT.  
  clear x\_fieldcat.  
endform.                    " SUB\_PREPARE\_FIELDCATALOG  
\*&---------------------------------------------------------------------\*  
\*&      Form  SUB\_SELECT\_RECORD  
\*&---------------------------------------------------------------------\*  
\* Fetches records from database into table i\_tab to be passed as export  
\* parameter t\_outtab in function module : REUSE\_ALV\_GRID\_DISPLAY  
\*----------------------------------------------------------------------\*  
form SUB\_SELECT\_RECORD.  
SELECT  mblnr A~matnr A~maktg charg  
        werks lgort menge meins dmbtr ebeln  
         FROM makt as a join mseg as b  
           on ( a~matnr = b~matnr )  
           INTO TABLE I\_TAB  
           where b~bwart = '101' .  
endform.                    " SUB\_SELECT\_RECORD  
\*&---------------------------------------------------------------------\*  
\*&      Form  SUB\_MODIFY\_RECORDS  
\*&---------------------------------------------------------------------\*  
\* Populate stat and icon columns of internal table i\_tab with specific  
\* columns and symbols based on some logic for quantity and value fields.  
\*----------------------------------------------------------------------\*  
form SUB\_MODIFY\_RECORDS.  
loop at i\_tab.  
if i\_tab-dmbtr gt 10000.  
\* Field icn of internal table is going to contain icon . For this column  
\*icon\_allowed is set in the field catalog table. For various icons,see  
\* type pool <ICON>  
 i\_tab-icn = ''.  
 modify i\_tab transporting icn.  
endif.  
if i\_tab-menge gt 50.  
\* Field icn of internal table is going to contain symbol . For this  
\* column symbol\_allowed is set in the field catalog table. For various  
\* icons,see type pool <SYMBOL>  
 i\_tab-sym = 'N'.  
 modify i\_tab transporting sym.  
endif.  
 IF I\_TAB-WERKS NE 'SDC1'.  
\* This field will contain lights , traffic signals : red.yellow,green  
\* That this field will be used as a light will be specified in the  
\* column of structure STRUCT\_LAYOUT.  
  I\_TAB-EXCPT = '1'.  
  MODIFY I\_TAB TRANSPORTING EXCPT.  
endif.  
endloop.  
endform.                    " SUB\_MODIFY\_RECORDS  
 \*&---------------------------------------------------------------------\*  
\*&      Form  SUB\_DETERMINE\_ALV\_LAYOUT                                 \*  
\*&---------------------------------------------------------------------\*  
\*& Defines the overall structure of the report layout                  \*  
\*----------------------------------------------------------------------\*  
form SUB\_DETERMINE\_ALV\_LAYOUT.  
\* Field EXCPT will show the light signal  
   STRUCT\_LAYOUT-LIGHTS\_FIELDNAME       = 'EXCPT'.  
\* Field BOS of the internal table will act as pushbutton and will appear  
\* at the left of the grid display. User will press that to select a  
\* record  
   struct\_layout-box\_fieldname          = 'BOX'.  
   STRUCT\_layout-totals\_text            = 'Totqty '.  
   STRUCT\_LAYOUT-ZEBRA                  = 'X'.  
   struct\_layout-confirmation\_prompt    = 'X'.  
   struct\_layout-detail\_titlebar        = 'Details of Storing'.  
   struct\_layout-no\_sumchoice           = 'X'.  
   struct\_layout-totals\_only            = 'X'.  
endform.                    " SUB\_DETERMINE\_ALV\_LAYOUT  
 \*&---------------------------------------------------------------------\*  
\*&      Form  SUB\_DETERMINE\_SORT\_SEQUENCE  
\*&---------------------------------------------------------------------\*  
\* Defines the sort sequence of the report  
\*----------------------------------------------------------------------\*  
form SUB\_DETERMINE\_SORT\_SEQUENCE.  
X\_sort-spos = 1.                           " Sort order  
X\_sort-fieldname = 'MATNR'.  
X\_sort-tabname = 'I\_TAB'.  
X\_sort-up = 'X'.  
X\_sort-subtot = 'X'.                      " Sub total allowed  
 append X\_sort TO IT\_SORT.  
clear X\_sort.  
endform.                    " SUB\_DETERMINE\_SORT\_SEQUENCE  
   
 \*&---------------------------------------------------------------------\*  
\*&      Form  SUB\_SHOW\_ALV\_LIST  
\*&---------------------------------------------------------------------\*  
\* Shows ALV list in grid form  
\*----------------------------------------------------------------------\*  
form SUB\_SHOW\_ALV\_LIST.  
 CALL FUNCTION 'REUSE\_ALV\_GRID\_DISPLAY'  
     EXPORTING  
    I\_CALLBACK\_PROGRAM           = V\_REPID            "\* Name of the program  
    I\_GRID\_TITLE                                 = 'Details of Storing'            "\* title  
    I\_callback\_pf\_status\_set               = 'PF\_STATUS\_SET'          "\* calls subroutine : PF\_STATUS\_SET  
    i\_callback\_user\_command           = 'USER\_COMMAND'          "\* Calls subroutine : user\_command  
    IS\_LAYOUT                                     = STRUCT\_LAYOUT         "\* Overall structure of the report  
    IT\_FIELDCAT                                  = I\_FIELDCAT                     "\* Passes the field catg internal table  
     IT\_SORT                                          = IT\_SORT                            "\* Passes the sort sequence internal table  
     I\_DEFAULT                                    = 'X'  
     I\_SAVE                                             = 'A'  
       IS\_VARIANT                                = i\_variant1                            
\* fetches different events into internal table i\_events  
       it\_events                                        = i\_events[]  
      TABLES  
\* Passes data table for ALV display  
        T\_OUTTAB                                  = I\_TAB  
     EXCEPTIONS  
       PROGRAM\_ERROR                     = 1  
       OTHERS                                         = 2.  
  IF SY-SUBRC <> 0.  
\* MESSAGE ID SY-MSGID TYPE SY-MSGTY NUMBER SY-MSGNO  
\*         WITH SY-MSGV1 SY-MSGV2 SY-MSGV3 SY-MSGV4.  
  ENDIF.  
endform.                    " SUB\_SHOW\_ALV\_LIST  
 \*&---------------------------------------------------------------------\*  
\*&      Form  set\_status  
\*&---------------------------------------------------------------------\*  
\*  Form used to set the Custom pf-status of the List Display  
\*----------------------------------------------------------------------\*  
\*  rt\_extab :  
\*----------------------------------------------------------------------\*  
FORM pf\_status\_set USING i\_rt\_extab TYPE slis\_t\_extab.  
 data : x\_extab type slis\_extab.  
 x\_extab-fcode = '&LFO'.  
 append x\_extab to i\_rt\_extab.  
\* Pf-status STANDARD of program SAPLSALV is copied to ZSTANDARD of the  
\* current program and the pushbutton for Information (okcode=&LFO) is  
\* excluded  
  SET PF-STATUS 'ZSTANDARD' excluding i\_rt\_extab .  
ENDFORM.  
   
 \*&---------------------------------------------------------------------\*  
\*&      Form  user\_command  
\*&---------------------------------------------------------------------\*  
\*  Form used to handle USER\_COMMAND events  
\*----------------------------------------------------------------------\*  
\*  rf\_ucomm: Function Code  
\*  rs      : Internal Table containing the selection information.  
\*----------------------------------------------------------------------\*  
 FORM user\_command USING rf\_ucomm LIKE sy-ucomm  
                          rs TYPE slis\_selfield.  
data : v\_mblnr like mseg-mblnr.  
case rf\_ucomm.  
 \* A custom pushbutton for record deletion is set in the GUI status. When a record is selected , the field BOC for that \*record becomes 'X'.  
\* The records are traced and deleted and the fields are refreshed( rs  of type slis\_selfield is refreshed)  
 when '&DEL'.                 "Print button clicked.  
    delete i\_tab where box = 'X'.  
     rs-refresh = 'X'.  
 \* When the user selects a row and presses the Select pushbutton ( user defined ) from the application toolbar, the details \* of the document will be shown in another ALV list  
 when '&SEL'.  
  PERFORM SUB\_SELECT\_DOCUMENT.  
 \*   set parameter id 'MBN'  field  i\_tab-mblnr.  
\*   call transaction 'MB03'.  
 \* Ok code for double click is &IC1 for ALV report  
 when '&IC1'.  
  perform sub\_hotspot.  
 endcase.  
ENDFORM.  
 \*&---------------------------------------------------------------------\*  
\*&      Form  SUB\_HOTSPOT  
\*&---------------------------------------------------------------------\*  
\*       text  
\*----------------------------------------------------------------------\*  
\*  -->  p1        text  
\*  <--  p2        text  
\*----------------------------------------------------------------------\*  
form SUB\_HOTSPOT.  
 message i398(00) with 'Hello'.  
endform.                    " SUB\_HOTSPOT  
   
 \*&---------------------------------------------------------------------\*  
\*&      Form  SUB\_VARIANT\_F4  
\*&---------------------------------------------------------------------\*  
\* Display a list of various variants of the report when the user presses F4 key in the variant field  
\*------------------------------------------------------------------------\*  
form SUB\_VARIANT\_F4.  
i\_variant-report = sy-repid.  
\* Utilising the name of the report , this function module will search for a list of variants and will fetch the selected one   \* into  the parameter field for variants  
CALL FUNCTION 'REUSE\_ALV\_VARIANT\_F4'  
       EXPORTING  
            IS\_VARIANT         = I\_VARIANT  
            I\_SAVE             = 'A'  
            I\_DISPLAY\_VIA\_GRID = 'X'  
       IMPORTING  
            ES\_VARIANT         = I\_VARIANT1  
       EXCEPTIONS  
            NOT\_FOUND          = 1  
            PROGRAM\_ERROR      = 2  
            OTHERS             = 3.  
  IF SY-SUBRC = 0.  
    P\_VAR = I\_VARIANT1-VARIANT.  
 ENDIF.  
ENDFORM.                    " SUB\_VARIANT\_F4  
 \*&---------------------------------------------------------------------\*  
\*&      Form  SUB\_SELECT\_DOCUMENT  
\*&---------------------------------------------------------------------\*  
form SUB\_SELECT\_DOCUMENT.  
data : v\_lines type i .  
 read table i\_tab with key box = 'X'.  
    Select \* from mseg INTO TABLE I\_DOC WHERE MBLNR = I\_TAB-MBLNR.  
   IF SY-SUBRC EQ 0 .  
  CALL FUNCTION 'REUSE\_ALV\_FIELDCATALOG\_MERGE'  
    EXPORTING  
       I\_PROGRAM\_NAME                  = V\_REPID  
       I\_INTERNAL\_TABNAME                         = 'I\_DOC'  
       I\_STRUCTURE\_NAME               = 'MSEG'  
    CHANGING  
      CT\_FIELDCAT                                              = I\_FIELDCAT1  
   EXCEPTIONS  
     INCONSISTENT\_INTERFACE                     = 1  
     PROGRAM\_ERROR                                       = 2  
     OTHERS                                                           = 3.  
   IF SY-SUBRC <> 0.  
   MESSAGE ID SY-MSGID TYPE SY-MSGTY NUMBER SY-MSGNO  
         WITH SY-MSGV1 SY-MSGV2 SY-MSGV3 SY-MSGV4.  
  ENDIF.  
 clear struct\_layout1.  
 STRUCT\_layout1-colwidth\_optimize  = 'X'.  
refresh it\_sort.  
clear it\_sort.

  CALL FUNCTION 'REUSE\_ALV\_GRID\_DISPLAY'  
     EXPORTING  
       I\_CALLBACK\_PROGRAM        = V\_REPID  
       I\_GRID\_TITLE                              = 'Details of Document'  
       IS\_LAYOUT                                  = STRUCT\_LAYOUT1  
       IT\_FIELDCAT                               = I\_FIELDCAT1  
       i\_structure\_name                          = 'MSEG'  
       I\_DEFAULT                                  = 'X'  
       I\_SAVE                                           = 'A'  
     TABLES  
        T\_OUTTAB                                  = I\_DOC  
     EXCEPTIONS  
       PROGRAM\_ERROR                     = 1  
       OTHERS                                         = 2  
              .  
  IF SY-SUBRC <> 0.  
 MESSAGE ID SY-MSGID TYPE SY-MSGTY NUMBER SY-MSGNO  
         WITH SY-MSGV1 SY-MSGV2 SY-MSGV3 SY-MSGV4.  
  ENDIF.  
ENDIF.  
endform.                    " SUB\_SELECT\_DOCUMENT  
 \*&---------------------------------------------------------------------\*  
\*&      Form  SUB\_COMMENT\_BUILD  
\*&---------------------------------------------------------------------\*  
form SUB\_COMMENT\_BUILD using   I\_top\_of\_page TYPE slis\_t\_listheader.  
DATA: ls\_line TYPE slis\_listheader.  
 \*\*\*Header  
  CLEAR ls\_line.  
  ls\_line-typ  = 'H'.  
\* LS\_LINE-KEY: not used for this type  
  ls\_line-info = 'Heading list'.  
  APPEND ls\_line TO I\_top\_of\_page.  
 \*\*\*Selection  
  CLEAR ls\_line.  
  ls\_line-typ  = 'S'.  
  ls\_line-key  = 'Key 1'.  
  ls\_line-info = 'Material '.  
  APPEND ls\_line TO i\_top\_of\_page.  
  ls\_line-key  = 'Key 2'.  
  ls\_line-info = 'Document no'.  
  APPEND ls\_line TO I\_top\_of\_page.  
 \*\*\*Action  
  CLEAR ls\_line.  
 endform.                    " SUB\_COMMENT\_BUILD

 \*&---------------------------------------------------------------------\*  
\*&      Form  SUB\_EVENTTAB\_BUILD  
\*&---------------------------------------------------------------------\*  
\* Defines the event table  
\*&---------------------------------------------------------------------\*  
FORM sub\_eventtab\_build USING l\_events TYPE slis\_t\_event.  
  DATA: ls\_event TYPE slis\_alv\_event.  
\* Get the different events of the ALV  
  CALL FUNCTION 'REUSE\_ALV\_EVENTS\_GET'  
       EXPORTING  
            i\_list\_type = 0  
       IMPORTING  
            et\_events   = l\_events.  
\* Search the top of page events  
  READ TABLE l\_events WITH KEY name = slis\_ev\_top\_of\_page INTO ls\_event.  
  IF sy-subrc = 0.  
    MOVE 'TOP\_OF\_PAGE' TO ls\_event-form.  
    APPEND ls\_event TO l\_events.  
  ENDIF.  
endform.                    " SUB\_EVENTTAB\_BUILD  
 \*&---------------------------------------------------------------------\*  
\*       FORM TOP\_OF\_PAGE                                                 \*  
\*&---------------------------------------------------------------------\*  
\* When TOP-OF-PAGE will be fired , this event will be called and it    
\* will use the contents of i\_list\_top\_of\_page for output in the header  
\*&---------------------------------------------------------------------\*  
FORM top\_of\_page.  
  CALL FUNCTION 'REUSE\_ALV\_COMMENTARY\_WRITE'  
       EXPORTING  
            i\_logo             = 'ENJOYSAP\_LOGO'  
            it\_list\_commentary = i\_list\_top\_of\_page.  
ENDFORM.

**OR**

**go through this link for reference guides** http://www.esnips.com/\_t\_/alv?q=alv\* \*\\

**Regards**

**fareedas**

* Sem r\u00f3tulos

2 Comentários

1. Ícone de usuário: 1003229hs

**Former Member**

nice document

* + [mar 18, 2014](https://wiki.scn.sap.com/wiki/display/ABAP/ALV+TUTOTIAL+FOR+BEGINERS+WITH+EXAMPLES?focusedCommentId=369066885#comment-369066885)

1. Ícone de usuário: julvepj

[**sriramula kishore**](https://wiki.scn.sap.com/wiki/display/~julvepj)

That was awsome man

Very nice effort

kudos

* + [jun 05, 2014](https://wiki.scn.sap.com/wiki/display/ABAP/ALV+TUTOTIAL+FOR+BEGINERS+WITH+EXAMPLES?focusedCommentId=377946928#comment-377946928)

* Follow SCN
* [**Contact Us**](http://go.sap.com/community/about/support.html)

* [**SAP Help Portal**](http://help.sap.com/)
* [Privacy](http://go.sap.com/about/legal/privacy.html)

* [Terms of Use](http://go.sap.com/corporate/en/legal/terms-of-use.html)

* [Legal Disclosure](http://go.sap.com/about/legal/impressum.html)

* [Copyright](http://go.sap.com/about/legal/copyright.html)