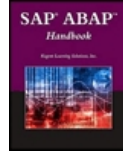


Chapters *To Go*



SAP ABAP Handbook

by Kogent Learning Solutions, Inc.
Jones and Bartlett Publishers. (c) 2010. Copying Prohibited.

Reprinted for Julio De Abreu Molina, IBM

jdeabreu@ve.ibm.com

Reprinted with permission as a subscription benefit of **Books24x7**,
<http://www.books24x7.com/>

All rights reserved. Reproduction and/or distribution in whole or in part in electronic, paper or other forms without written permission is prohibited.



Chapter 13: Reports

Overview

A report, in general, is a presentation of data in a specific format and organized structure. Many DBMS include a report writer that enables you to design and generate reports. SAP is one of the software applications that support reports creation. In SAP, you can create three types of reports: classical, interactive, and ABAP List Viewer (ALV) reports. In a classical report, the output is displayed in a single list, while in an interactive report, you can view multiple lists simultaneously. ALV reports, however, allow you to perform various functions with the displayed output, such as sorting, arranging, filtering, and retrieving data. In addition, you can view the output of ALV reports either in a grid view or a list view, by using the `REUSE_ALV_GRID_DISPLAY` and `REUSE_ALV_LIST_DISPLAY` function modules, respectively.

Classical reports are simple reports, which are created by using the output data (final data that have to be displayed in a report) in the `WRITE` statement inside a loop. These reports do not contain any sub reports. Interactive reports, on the other hand, are used when you need to interact with a report. With the help of interactive reports, first, an overview list (also called a basic list) is displayed, based on which further output lists (or called secondary lists) are displayed. These secondary lists are actually sub reports, which are displayed when you click specific values of fields included in the basic list. Moreover, SAP also provides some standard reports, such as `RSCLTCOP`, which is used to copy tables across clients, and `RSPARAM`, which is used to display instance parameters. Using these standard reports, you can create your own customized reports.


In this chapter, you learn how to create a classical report and explore the events used to create classical reports. You also learn how to create interactive reports and comprehend the events used to create an interactive report. Finally, you learn about ALV reports.

Now, let's learn about classical reports and how to create them.

Working with Classical Reports

As discussed earlier, a classical report displays data by using the `WRITE` statement inside a loop. Classical reports are normal reports and do not contain any sub report. These reports consist of only one screen/list as an output. You use various events, such as `INITIALIZATON` and `TOP-OF-PAGE`, to create a classical report. Each event has its own importance during the creation of a classical report. Each of these events is related to a specific user-action and is triggered only when the user performs that action. Table 13.1 lists the events and circumstances when these events are triggered:

Table 13.1: Description of the events

Entry	Description
INITIALIZATON	Triggered before displaying the selection screen.
AT SELECTION-SCREEN	Triggered after the processing of the user input on the selection screen. This event verifies the user input prior to the execution of a program. After processing the user input, the selection screen stills remains in the active mode.
START-OF-SELECTION	Triggered only after the processing of the selection screen is over; that is, when the user clicks the <code>Execute</code>  icon on the selection screen.
END-OF-SELECTION	Triggered after the last statement in the <code>START-OF-SELECTON</code> event is executed.
TOP-OF-PAGE	Triggered by the first <code>WRITE</code> statement to display the data on a new page.
END-OF-PAGE	Triggered to display the text at the end of a page in a report. Note that this event is the last event while creating a report, and should be combined with the <code>LINE-COUNT</code> clause of the <code>REPORT</code> statement.

Classical reports are executable-type programs; that is, these reports always start with the `REPORT` statement. The `REPORT` statement, together with the name of the executable program, appears on the screen of ABAP Editor (where the user writes the program code) by default.

Table 13.2 describes the clauses used in the `REPORT` statement:

Table 13.2: Clauses to the REPORT statement

Clauses	Description
---------	-------------

LINE-SIZE	Specifies the width of the output list, or, in other words, the number of characters to be displayed in the output list for printing or displaying. Note that a line can have 255 characters for displaying data, but for printing purposes, a maximum of 132 characters is allowed.
LINE-COUNT	Specifies the number of lines per page. After declaring the number of lines per page, the number of lines reserved for the footer is specified in a pair of brackets. This footer is accessed by using the <code>END-OF-PAGE</code> event.
MESSAGE-ID	Allows you to use the <code>MESSAGE</code> statement without explicitly specifying the message id.
NO STANDARD PAGE HEADING	Directs SAP not to display the standard page heading (as declared in the Title field of the attributes subobject) on every new page.

Apart from the events and clauses used with the `REPORT` statement, the `SELECT-OPTIONS` statement is used to create a user interface that determines how a selection screen is displayed, by entering the values of the fields of the selection table. The following syntax is used for the `SELECT-OPTIONS` statement:

```
SELECT-OPTIONS <selection_tab> FOR <fld>.
```

In this syntax, `<selection_tab>` represents an object of an internal table of the standard table type, which has a standard key and a header line. The `<fld>` expression represents a column of a database table or an internal field in the corresponding program. When the `SELECT-OPTIONS` statement is executed, an internal table (or a selection table in this context) containing four fields, `SIGN`, `OPTION`, `LOW`, and `HIGH`, are created. These fields correspond to the fields of a database table or an internal field in the corresponding program. Note that the `SELECT-OPTIONS` statement can only be used with the fields of a table that is defined in the `TABLES` statement. [Table 13.3](#) describes the four fields of the internal table created:

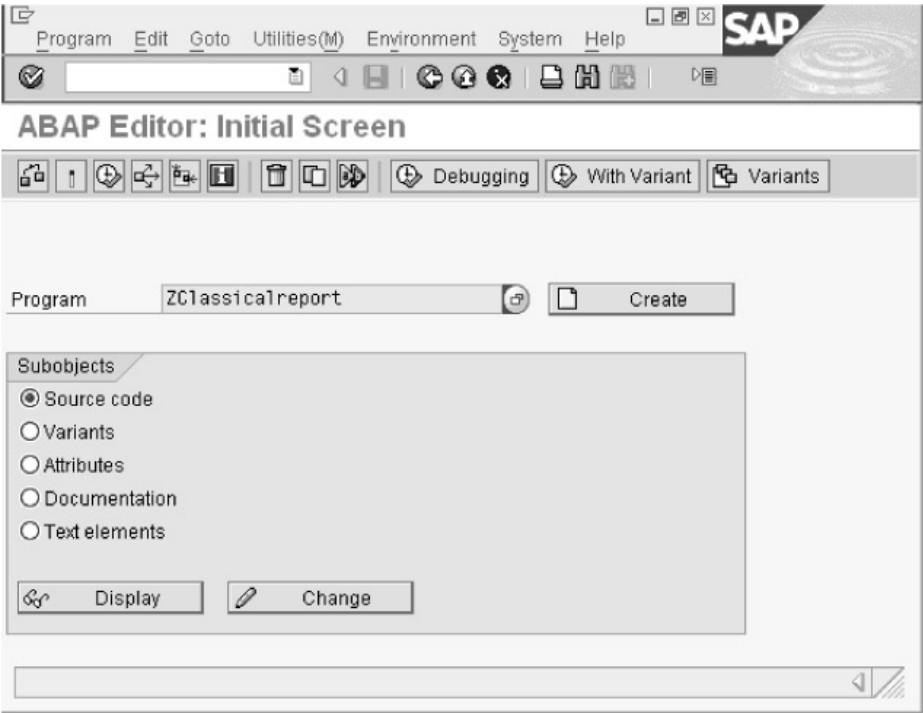
Table 13.3: Description of the fields

Fields	Description
SIGN	Holds one of the two values, I (include) or E (exclude), depending on whether or not to include the specified values.
OPTION	Specifies the option for comparison, such as <code>EQ</code> (equal to), <code>NE</code> (not equal to), and <code>GT</code> (greater than).
LOW	Specifies the lower limit for a selection range.
HIGH	Specifies the upper limit for a selection range.

Creating a Classical Report

You can create a classical report by using ABAP Editor. A classical report is used to display the information stored in any database table, such as `MARA`, `MAKT`, or `KNA1`. This task is performed by writing a sequence of statements in ABAP Editor. Perform the following steps to create a classical report:

1. Open the initial screen of ABAP Editor by either navigating to the SAP menu or executing the `SE38` transaction code.
2. In the initial screen of ABAP Editor, enter the name of the program for the classical report, select the `Source code` radio button in the `Subobjects` group box, and click the `Create` button to create a new program, as shown in [Figure 13.1](#):

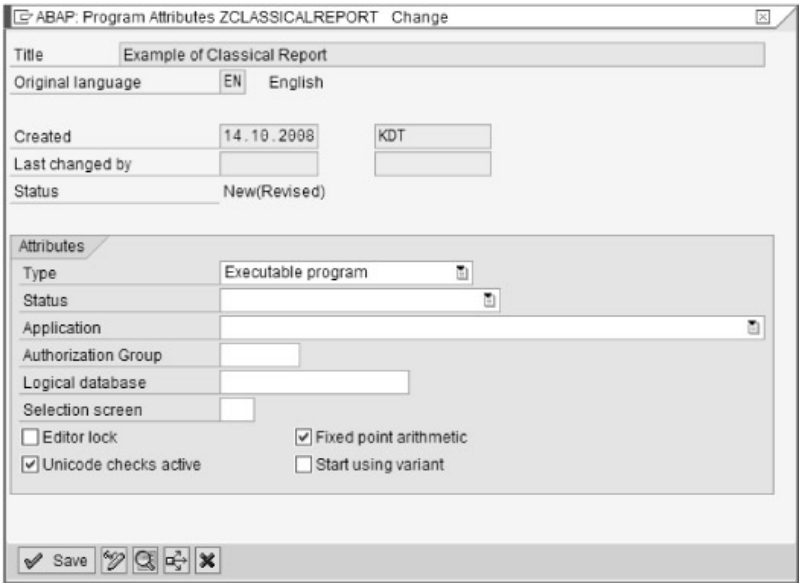


© SAP AG. All rights reserved.

Figure 13.1: Entering the name of the classical report

The ABAP: Program Attributes dialog box appears (Figure 13.2).

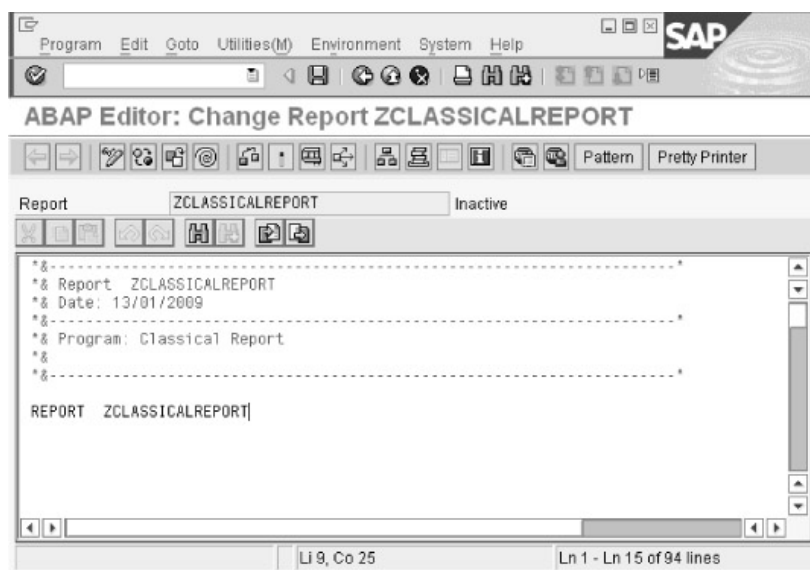
3. In the ABAP: Program Attributes dialog box, enter a short description for the program in the Title field and select the Executable program option from the Type drop-down list in the Attributes group box, as shown in Figure 13.2:
4. Now, click the Save (Save icon) button in the ABAP: Program Attributes dialog box or press the ENTER key (Figure 13.2). The Create Object Directory Entry dialog box appears.
5. In the Create Object Directory Entry dialog box, enter ZKOG_PCKG as the package name in the Package field and click the Save (Save icon) icon.



© SAP AG. All rights reserved.

Figure 13.2: Saving the attributes of ZCLASSICALREPORT

The ABAP Editor: Change Report ZCLASSICALREPORT screen appears, as shown in [Figure 13.3](#):



© SAP AG. All rights reserved.

Figure 13.3: ABAP editor change report screen

- Write the code given in [Listing 13.1](#), in the ABAP Editor: Change Report screen, to generate the respective report:

Listing 13.1: Code to create the classical report

```
REPORT ZCLASSICALREPORT
  LINE-SIZE 70
  LINE-COUNT 30(3)
  NO STANDARD PAGE HEADING.

*-----*
*/ Tables Used in Program
*/
Tables: MARA.
*-----*
*/ Data Definitions for the Program
*/
DATA: Begin of itab occurs 10,

      MATNR LIKE MARA-MATNR,
      MBRSH LIKE MARA-MBRSH,
      MEINS LIKE MARA-MEINS,
      MTART LIKE MARA-MTART,

      End of itab.
*-----*
*/ Program Selections
*/
SELECT-OPTIONS: MATERIAL FOR MARA-MATNR.
*-----*
*/ INITIALIZATION Event Used for Setting the Default Values
to the Parameters
*/
INITIALIZATION.
MATERIAL-LOW = '1'.
MATERIAL-HIGH = '500'.
APPEND MATERIAL.
```

```

*-----*
*/ AT SELECTION SCREEN Event Used for Performing Parameters
Verification
*/
AT SELECTION-SCREEN.

    IF MATERIAL-LOW = ' '.
        MESSAGE I000(ZKMESSAGE).
    ELSEIF MATERIAL-HIGH = ' '.
        MESSAGE I001(ZKMESSAGE).
    ENDIF.
*-----*
*/ START-OF-SELECTION Event
*/
START-OF-SELECTION.
SELECT MATNR MBRSH MEINS MTART FROM MARA INTO CORRESPONDING
FIELDS OF itab WHERE MATNR IN MATERIAL.



LOOP AT itab.
WRITE:/ itab-MATNR UNDER 'MATERIAL',
        itab-MBRSH UNDER 'INDUSTRY',
        itab-MEINS UNDER 'UNITS',
        itab-MTART UNDER 'MATERIAL TYPE'.
*-----*
*/ END-OF-SELECTION Event
*/
END-OF-SELECTION.
WRITE:/ 'CLASSICAL REPORT CREATED BY SHIVAM SRIVASTAVA'
COLOR 7.
ULINE.
SKIP.
*-----*
*/ TOP-OF-PAGE Event Used for Displaying Information at the
Top of Every Page
TOP-OF-PAGE.
WRITE:/ 25 'CLASSICAL REPORT CONTAINING THE GENERAL MATERIAL
DATA FROM THE TABLE MARA' COLOR 6.
ULINE.
WRITE:/ 'MATERIAL' COLOR 1,
        'INDUSTRY' COLOR 2,
        'UNITS' COLOR 3,
        'MATERIAL TYPE' COLOR 4.
ULINE.
END-OF-PAGE.



WRITE:/ 'PAGE NUMBER', SY-PAGNO,
        'DATE', SY-DATUM.
*-----*

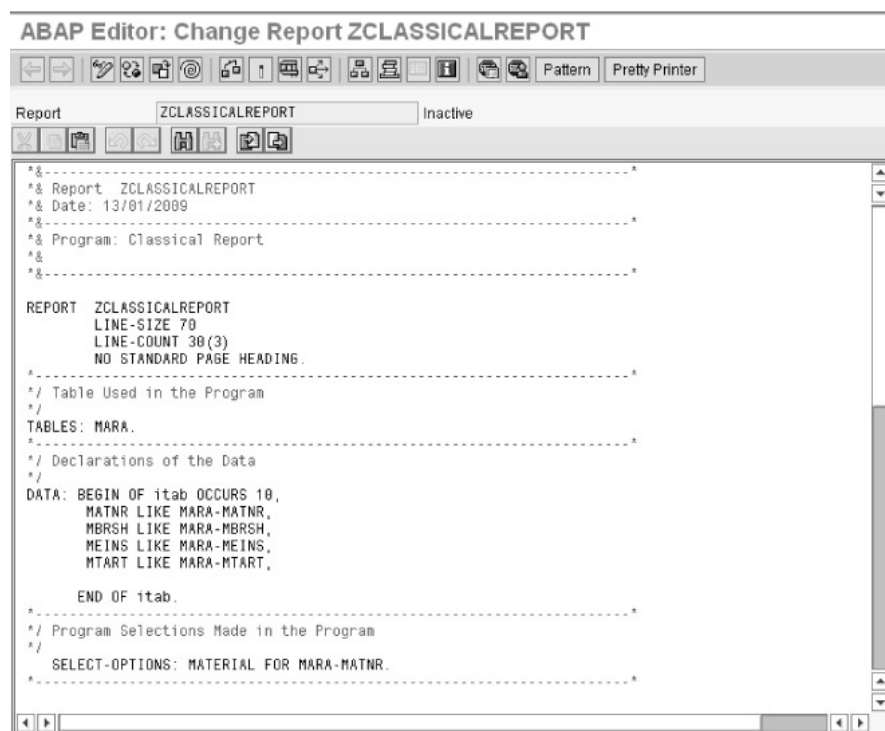
```

In [Listing 13.1](#), the width of the list to be displayed is 70 characters. The `LINE-SIZE` clause of the `REPORT` statement represents that the number of lines per page has been set to 30 and 4 lines have been left for the footer. The `TABLES` statement is used to declare all the database tables that are accessed in the report. In our case, the `MARA` table is accessed. An internal table named `MATERIAL` is declared with the help of the `SELECT-OPTIONS` statement.

[Figure 13.4](#) shows the screen of ABAP Editor after entering the code given in [Listing 13.1](#):

7. Click the Save  icon or press the CTRL + S keys combination to save the changes in the program.
8. Next, click the Check  icon or press the CTRL + F2 keys combination to check and remove syntax errors or warnings (if any) in the program.

9. Next, click the **Activate**  icon or press the CTRL + F3 keys combination to activate the program.
10. Now, click the **Direct Processing**  icon or press the F8 key to execute the program.





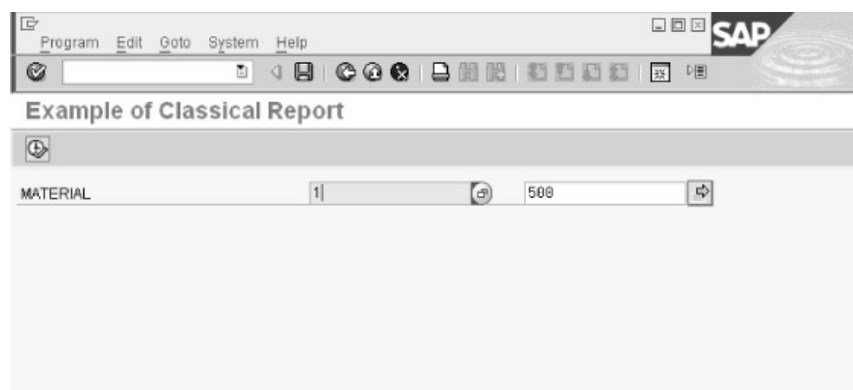
© SAP AG. All rights reserved.

Figure 13.4: Adding code in ABAP editor to create a classical report

Figure 13.5 shows the output of Listing 13.1:

In Figure 13.5, the minimum (1) and maximum (500) values of the MATERIAL variable appear in the respective fields. These values are already declared in the `INITIALIZATION` event. Note that you can modify these values as per your choice.

11. If you remove the minimum value of the MATERIAL variable [that is, 1 (by default)] and then click the **Execute**  icon, an information dialog box appears, prompting you to enter a value for the lower limit of the variable. Figure 13.6 shows the information dialog box:
12. Click the **Continue**  icon. Notice that the system takes the lower limit of the variable as 1 by default. On the output screen, the data corresponding to the MATNR, MBRSH, MEINS, and MTART fields are displayed. Figure 13.7 shows the classical report created:



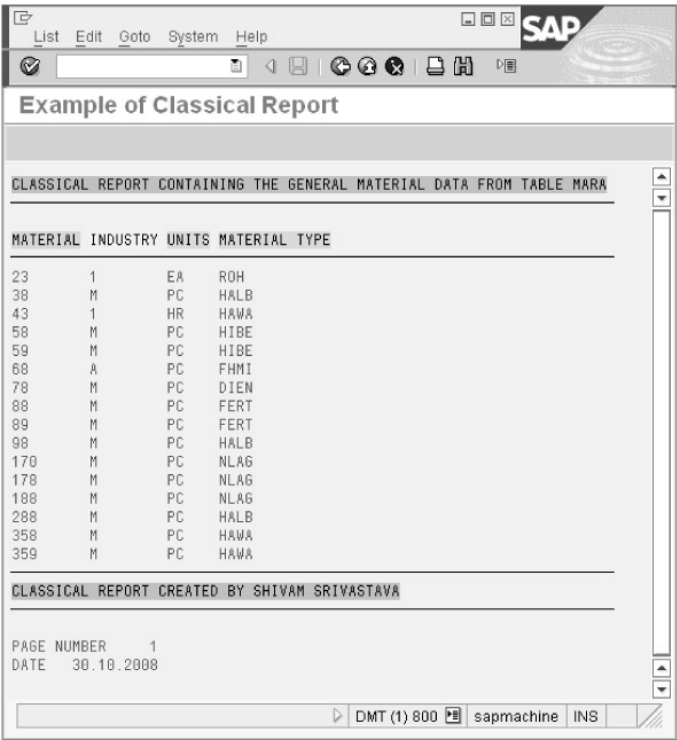
© SAP AG. All rights reserved.

Figure 13.5: The selection screen



© SAP AG. All rights reserved.

Figure 13.6: Information dialog box



© SAP AG. All rights reserved.

Figure 13.7: A classical report retrieving material data from the MARA table


Interactive Reports

Interactive reports allow you to control the retrieval of data. With the help of interactive reports, the user first displays an overview list, also known as the basic list. On the basis of the basic list, other lists are displayed according to the user requirements. These other lists or sublists (also known as secondary lists), provide a detailed description of data. For example, if the first column in an overview list of an interactive report represents the data related to material numbers, and you want to display the information of all the vendors related to a particular material number, that material number is temporarily stored at the current line and then another report or sublist containing the vendor details corresponding to the clicked material number is displayed.

Interactive reports can contain a basic list (containing the data displayed after the interactive reports are executed) and additional 19 secondary detailed lists. In other words, an interactive report can contain a maximum of 20 lists. If a user tries to generate a list numbered as 21, a runtime error occurs.

Various events are used to create an interactive report. These events are triggered only after you perform an action, such as double clicking a line in the displayed list, or pressing a function key. [Table 13.4](#) shows when these events are triggered:

Table 13.4: Description of the events

Events	Description
AT LINE-SELECTION	Triggered from a displayed list; that is, whenever the user selects the Choose  icon or double-clicks a line in the displayed list.
AT USER-COMMAND	Triggered when the user executes a function defined within a menu of the displayed list.
AT PFn	Triggered for the predefined function keys.
TOP OF PAGE DURING LINE SELECTION	Triggered for the top of the page event defined for secondary lists.

In addition to the events shown in [Table 13.4](#), interactive reports also use the events defined in [Table 13.1](#).

Creating an Interactive Report

In this section, you learn how to create an interactive report that accesses data-related various fields of multiple tables in the SAP database. The data and the table fields are as follows:

- Vendor master from the LFA1 table
- General material data from the MARA table
- Data related to the purchasing document header from the EKKO table
- Purchasing document item data from the EKPO table
- Material descriptions from the MAK1 table

This interactive report consists of one basic list and four secondary lists. The basic list contains information related to the vendor master. The first secondary list contains information related to the purchasing document header, the second secondary list contains information related to the purchasing document item, the third secondary list contains information related to the general material description, and the fourth secondary list contains the description of the material.

Note The tables accessed by interactive reports must have at least one field in common. This is because when you double-click any particular value from a list, the corresponding values from some other table for that particular value is displayed in the subsequent list.

In this case, the `LIFNR` field is common for the LFA1 and EKKO tables. The `EBELN` field is common for the EKKO and EKPO tables. The `MATNR` field is common for the EKPO and MARA tables and the last field, `MATNR`, is common for the MARA and MAK1 tables.

Perform the following steps to create an interactive report:

1. Open the initial screen of ABAP Editor by either navigating the SAP menu or entering the `SE38` transaction code. The initial screen of ABAP Editor appears (see [Figure 13.8](#)).
2. In the initial screen of ABAP Editor, enter a name for the program, select the `Source` code radio button in the `Subobjects` group box, and click the `Create` button to create a new program, as shown in [Figure 13.8](#):



Figure 13.8: Creating interactive reports

The ABAP: Program Attributes dialog box appears.

3. In the ABAP: Program Attributes dialog box, enter a short description (Example for interactive reports) for the program in the Title field and select the Executable program option from the Type drop-down in the Attributes group box. Now, click the Save (Save icon) button in the ABAP: Program Attributes dialog box or press the ENTER key. The Create Object Directory Entry dialog box appears.
4. In the Create Object Directory Entry dialog box, enter ZKOG_PCKG as the package name in the Package field and click the Save (Save icon) icon. The ABAP Editor: Change Report screen appears.
5. Write the code given in Listing 13.2, in the ABAP Editor: Change Report screen, to generate the interactive report:

Listing 13.2: Code to create an interactive report

```
REPORT      ZINTERACTIVEREPORT
            LINE-SIZE 90
            LINE-COUNT 30(4)
            NO STANDARD PAGE HEADING.

*-----*
*/ Tables Used in the Program
*/
Tables: MARA, LFA1, EKPO, EKKO, MAKT.
*-----*
*/ Program Selections Used for the Program
*/
SELECT-OPTIONS: VENDOR FOR LFA1-LIFNR.
*-----*
*/ INITIALIZATION Event Used for the Default Values to the
Parameters
*/
INITIALIZATION.
VENDOR-LOW = '1'.
VENDOR-HIGH = '200'.
```

```

VENDOR-SIGN = 'I'.
APPEND VENDOR.
*-----*
*/ TOP-OF-PAGE Event Used for the Program Heading
*/
TOP-OF-PAGE.
WRITE: 'WELCOME TO INTERACTIVE REPORT: YOU SEE THE FIRST
LIST OF THE REPORT KNOWN
AS BASIC LIST' COLOR 7.
ULINE.
SKIP.
WRITE:/10 'VENDOR' COLOR 1,
        30 'NAME' COLOR 2,
        66 'LAND1' COLOR 3,
        80 'SORTL' COLOR 4.
*-----*
*/ START-OF-SELECTION Event Used for Starting Main Program
Processing

*/
START-OF-SELECTION.

SELECT * FROM LFA1 WHERE LIFNR IN VENDOR.

WRITE:/ LFA1-LIFNR UNDER 'VENDOR',
        LFA1-NAME1 UNDER 'NAME',
        LFA1-LAND1 UNDER 'LAND',
        LFA1-SORTL UNDER 'SORTL'.

HIDE LFA1-LIFNR.
CLEAR LFA1.
ENDSELECT.
*-----*
*/ AT LINE-SELECTION Event Used for User Selection
*/
AT LINE-SELECTION.
CASE SY-LSIND.

WHEN '1'.

SELECT * FROM EKKO WHERE LIFNR = LFA1-LIFNR.
WRITE:/ EKKO-LIFNR UNDER 'VENDOR',
        EKKO-EBELN UNDER 'PURCHASE',
        EKKO-BUKRS UNDER 'COMPANY CODE',
        EKKO-BSART UNDER 'DOC TYPE'.

HIDE EKKO-EBELN.
CLEAR EKKO.
ENDSELECT.

WHEN '2'.
WRITE:/10 'PURCHASE NUMBER' COLOR 1,
        40 'MATERIAL NUMBER' COLOR 2,
        60 'DOC TYPE' COLOR 3.

SELECT * FROM EKPO WHERE EBELN = EKPO-EBELN.

WRITE:/ EKPO-EBELN UNDER 'PURCHASE',
        EKPO-BUKRS UNDER 'MATERIAL NUMBER',
        EKKO-EBELN UNDER 'DOC TYPE'.
HIDE EKPO-EBELN.
CLEAR EKPO.

```

```

ENDSELECT.

WHEN '3'.
WRITE:/10 'PURCHASE NUMBER' COLOR 1,
        30 'INDUSTRY' COLOR 2,
        40 'DOC TYPE' COLOR 3.

SELECT * FROM MARA WHERE MATNR = MARA-MATNR.
WRITE:/ MARA-MATNR UNDER 'MATERIAL NUMBER',
        MARA-MBRSH UNDER 'INDUSTRY',
        MARA-MTART UNDER 'DOC TYPE'.

HIDE MARA-MATNR.
CLEAR MARA.
ENDSELECT.

WHEN '4'.
WRITE:/10 'MATERIAL NUMBER' COLOR 1,
        30 'DESCRIPTION' COLOR 2.

SELECT * FROM MAKT WHERE MATNR = MARA-MATNR.
WRITE:/ MAKT-MATNR UNDER 'MATERIAL NUMBER',
        MAKT-MAKTX UNDER 'DESCRIPTION'.

ENDSELECT.
ENDCASE.
*-----*
*/ TOP-OF-PAGE Used for Setting the Headings for the
Secondary List
*/
TOP-OF-PAGE DURING LINE-SELECTION.
CASE SY-LSIND.

WHEN '1'.
WRITE: / 'REPORT CONTAINING THE DETAILS REAGARDING PURCHASING
DOCUMENT HEADER:: FIRST SECONDARY LIST' COLOR 5.
ULINE.
SKIP.
WRITE:/10 'VENDOR' COLOR 1,
        30 'PURCHASE' COLOR 2,
        60 'COMPANY CODE' COLOR 3,
        90 'DOC TYPE' COLOR 4.

WHEN '2'.
WRITE:/ 'REPORT CONTAINING THE DETAILS REGARDING PURCHASING
DOCUMENT ITEM:: SECOND SECONDARY LIST' COLOR 5.
ULINE.
SKIP.

WHEN '3'.
WRITE:/ 'REPORT CONTAINING THE DETAILS REAGARDING PURCHASING
DOCUMENT HEADER:: THIRD SECONDARY LIST' COLOR 5.
ULINE.
SKIP.

WHEN '4'.
WRITE:/ 'REPORT CONTAINING THE DETAILS REAGARDING MATERIAL
DESCRIPTIONS::FOURTH SECONDARY LIST' COLOR 5.
ULINE.
SKIP.
ENDCASE.
*-----*

```

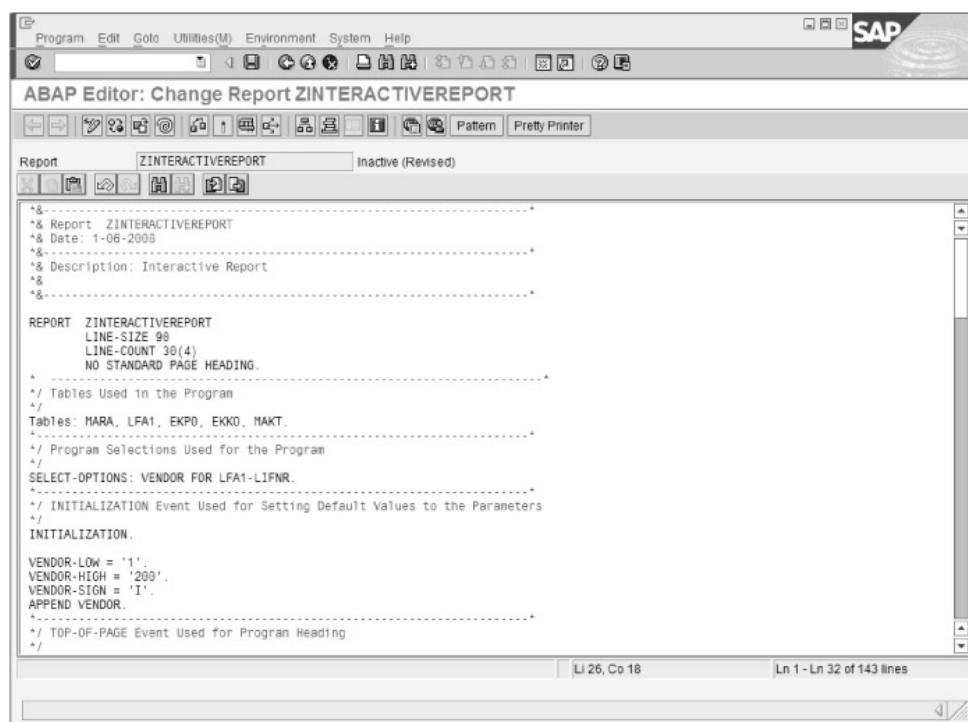
```

*/ END-OF-PAGE Event for Displaying the Page Number, Date
at the End of the Page
*/
END-OF-PAGE.
SKIP.
WRITE:/'PAGE NUMBER', SY-PAGNO,
      / 'DATE', SY-DATUM.
/ 'REPORT CREATED BY SHIVAM SRIVASTAVA'.
*-----*

```

Figure 13.9 shows the screen of ABAP Editor after entering the code given in Listing 13.2:

6. Click the Save (💾) icon and then click the Check (🔍) icon. Finally, click the Activate (🔑) icon to activate the program before executing it. Click the Direct processing (🖨️) icon to process the interactive report created. The selection screen appears, displaying the VENDOR field. The value for the lower limit and upper limit of the VENDOR field are 1 and 2000, respectively. Note that you can specify these values as per your requirement.
7. Click the Execute (🏃) icon, as shown in Figure 13.10:
8. The first output screen (basic list) appears, which displays information about the vendor master from the LFA1 table, as shown in Figure 13.11:
9. Double-click a particular value of a vendor from the basic list; for example, double-click 5.



© SAP AG. All rights reserved.

Figure 13.9: The ABAP editor screen

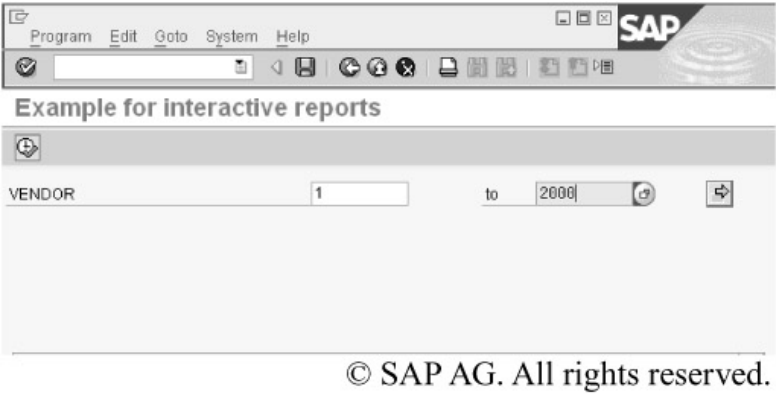


Figure 13.10: Selection screen



Figure 13.11: The basic list containing vendor master data

Note Alternatively, you can first click 5 and then the Choose  icon.

You can now see the corresponding values of the fields from the EKKO table. Use the HIDE statement to prevent any change in the selected value. The HIDE statement places the field name, that is, LFA1-LIFNR, and its content in the hidden area of the list. The CLEAR statement is used to clear the value of the fields stored on the output line.

Figure 13.12 shows the first secondary list corresponding to the value of the selected field; that is, 5:

Now, from the first secondary list, you can retrieve the information related to the purchasing document item from the EKPO table, corresponding to the value selected from the first secondary list.

List Edit Goto System Help

SAP

Example for interactive reports

REPORT CONTAINING THE DETAILS REGARDING PURCHASING DOCUMENT HEADER:: FIRST SECONDARY LIST

VENDOR	PURCHASE	COMPANY CODE	D
5	4600000062	3000	M
5	4500017119	3000	N
5	4500017120	3000	N
5	4500017122	3000	N
5	4500017132	3000	N
5	4500017133	3000	N
5	4500017134	3000	N
5	4500017137	3000	N
5	4500017138	3000	N
5	4500017154	3000	N
5	4500017156	3000	N
5	4500017157	3000	N
5	4500017158	3000	N
5	4500017159	3000	N
5	4500017160	3000	N
5	4500017161	3000	N
5	4500017162	3000	N
5	4500017163	3000	N
5	4500017164	3000	N
5	4500017165	3000	N

PAGE NUMBER 1
DATE 04.10.2008
REPORT CREATED BY SHIVAM SRIVASTAVA

REPORT CONTAINING THE DETAILS REGARDING PURCHASING DOCUMENT HEADER:: FIRST SECONDARY LIST

VENDOR	PURCHASE	COMPANY CODE	D
5	4500017166	3000	N
5	4500017167	3000	N

© SAP AG. All rights reserved.

Figure 13.12: First secondary list

Note At times, when you double-click a particular value, you may not find any respective value from the next defined table. This is because the corresponding table might not contain any entry for the selected value.

In this case, if a user double-clicks the value 5 to see the details regarding the purchasing document item from the EKPO table, the user cannot see any details because the EKPO table does not contain any entries in the EBELN, MATNR, and EBELP fields for the selected value. Figure 13.13 shows the second secondary list containing no values:

Because no values can be seen on this window, you will not be able to see any value in the third and fourth secondary lists. If you double-click any title, you see the third secondary list, as shown in Figure 13.14:

Similarly, you can also see the fourth secondary list.

List Edit Goto System Help

SAP

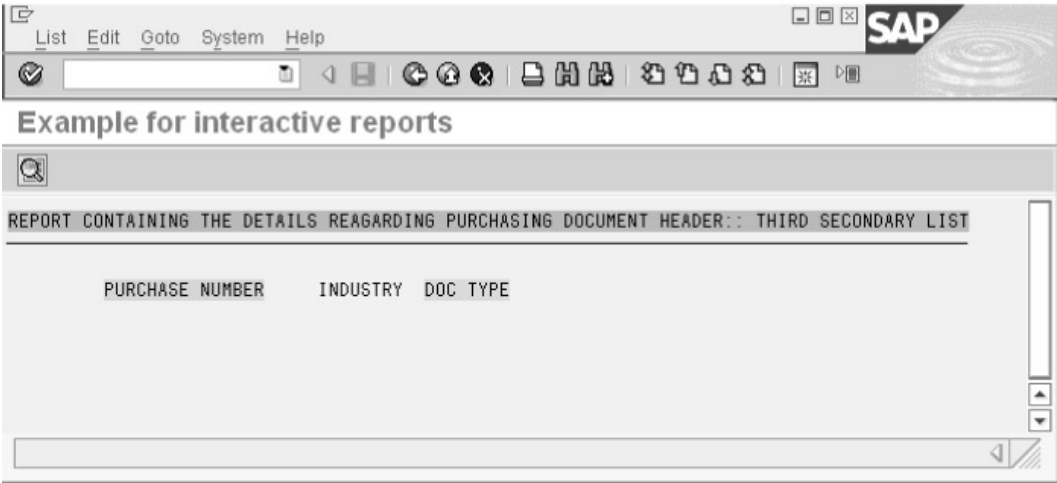
Example for interactive reports

REPORT CONTAINING THE DETAILS REGARDING PURCHASING DOCUMENT ITEM:: SECOND SECONDARY LIST

PURCHASE NUMBER	MATERIAL NUMBER	DOC TYPE
-----------------	-----------------	----------

© SAP AG. All rights reserved.

Figure 13.13: Second secondary list



© SAP AG. All rights reserved.

Figure 13.14: Third secondary list

Comparing Classical and Interactive Reports

Although both classical and interactive reports are used to retrieve data from one or more database tables, there exist some basic differences between the two. Table 13.5 compares a classical report with an interactive report:

Table 13.5: Comparison between classical and interactive reports

Classical Report	Interactive Report
Does not allow the user to interact with an SAP system	Allows the user to interact with an SAP system
Provides no control over the output after executing the report	Enables users to control the output after executing the report
Does not support drilling, which means that you cannot navigate to the next level of detail	Supports drilling, which means that the user can navigate to the next level of detail

ALV Reports

ALV reports allow you to perform various functions dynamically, such as sorting, arranging, filtering, and retrieving data. You can use various ALV function modules to perform functions on the data of the report. You need to create an ALV report when you need to include the columns that store more than 255 characters in their field values in a report. Using ALV function modules, you can select columns for displaying, can arrange the columns as per requirements, and can also save different variants for report display. An ALV report can display up to 90 columns or more with various display options. Some features of ALV reports include:

- User friendliness
- Ability to handle events
- Use of moderate code

Table 13.6 describes various ALV function modules available in the SAP system:

Table 13.6: Function modules for ALV reports

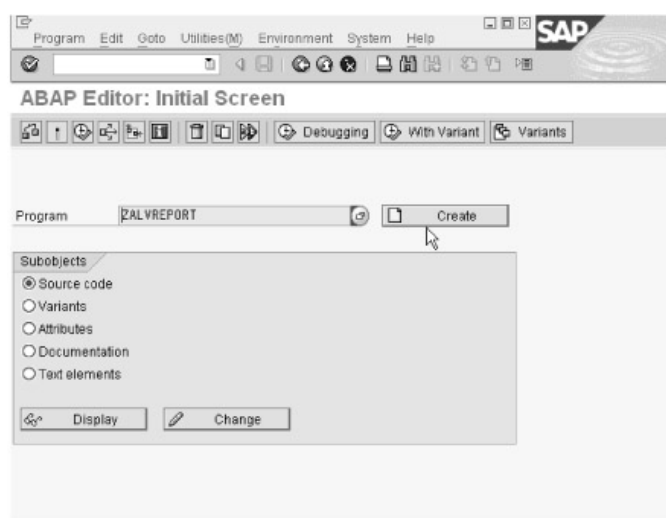
Function Module	Description
REUSE_ALV_GRID_DISPLAY	Displays an ALV grid as per the parameters defined in the function call
REUSE_ALV_LIST_DISPLAY	Displays an ALV list as per the parameters defined in the function call
REUSE_ALV_COMMENTARY_WRITE	Displays the list header information
REUSE_ALV_HIERSEQ_LIST_DISPLAY	Displays two internal tables that are formatted in the hierarchical-sequential list format
REUSE_ALV_VARIANT	Displays a variant selection dialog box
REUSE_ALV_VARIANT_EXISTENCE	Checks the existence of the variant display

Now, let's learn how to create an ALV report by using the `REUSE_ALV_GRID_DISPLAY` function module.

Creating an ALV Report

In this section, you explore how to create an ALV report, which accesses data of the QMEL table of the SAP database. First, a structure named `I_QMEL` is created, which is similar to the structure of the QMEL table. A title, "AN ALV REPORT CREATED BY SHIVAM SRIVASTAVA", is then assigned to the report by using the `WS-TITLE` variable. Perform the following steps to create the ALV report:

1. Open the initial screen of ABAP Editor by either navigating to the `SAP` menu or executing the `SE38` transaction code.
2. In the initial screen of ABAP Editor, enter `ZALVREPORT` as the name of the program, select the `Source code` radio button in the `Subobjects` group box, and click the `Create` pushbutton to create a new program, as shown in [Figure 13.15](#):



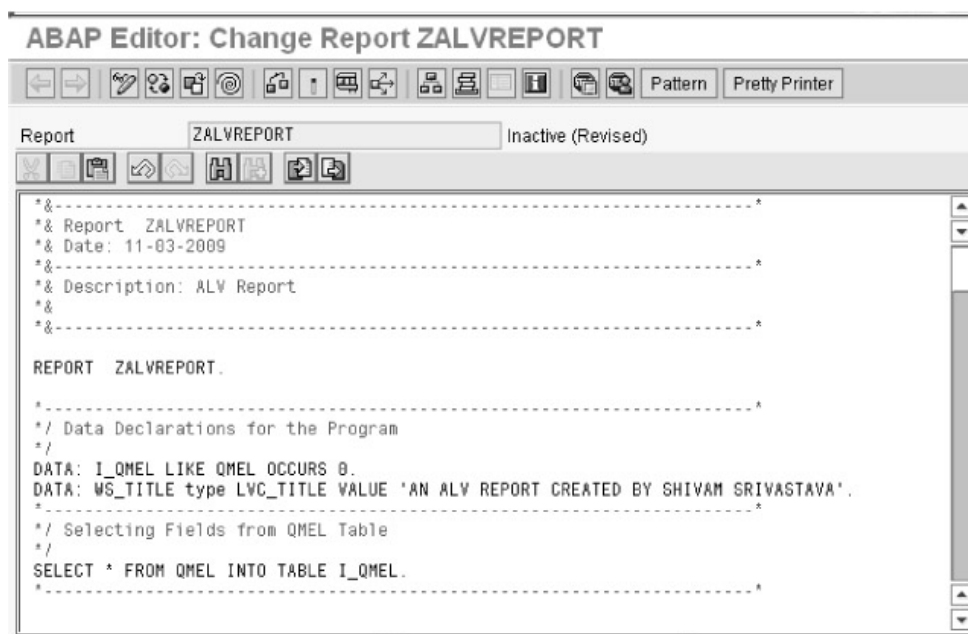
© SAP AG. All rights reserved.

Figure 13.15: The initial screen of ABAP editor

The `ABAP: Program Attributes` dialog box appears.

3. In the `ABAP: Program Attributes` dialog box, enter a short description (Example of ALV Reports) for the program in the `Title` field and select the `Executable program` option from the `Type` drop-down list in the `Attributes` group box. Now, click the `Save` (Save icon) pushbutton in the `ABAP: Program Attributes` dialog box or press the `ENTER` key. The `Create Object Directory Entry` dialog box appears.
4. In the `Create Object Directory Entry` dialog box, enter `ZKOG_PCKG` as the package name in the `Package` field and click the `Save` (Save icon) icon. The `ABAP Editor: Change Report ZALVREPORT` screen appears (see [Figure 13.16](#)).
5. Write the following code snippet for the ALV report being created in the `ABAP Editor: Change Report ZALVREPORT` screen:

```
*-----*
*/ Data Declarations for the Program
DATA: I_QMEL LIKE QMEL OCCURS 0.
DATA: WS_TITLE type LVC_TITLE VALUE 'AN ALV REPORT CREATED
BY SHIVAM SRIVASTAVA'.
*-----*
*/ Selecting Fields from QMEL Table
*/
SELECT * FROM QMEL INTO TABLE I_QMEL
*-----*
```



© SAP AG. All rights reserved.

Figure 13.16: The ABAP editor screen where the code for the ALV report is written

Figure 13.16 shows the ABAP Editor: Change Report ZALVREPORT screen after inserting the preceding code snippet:

6. Call the `REUSE_ALV_GRID_DISPLAY` function module by clicking the **Pattern** button present on the application toolbar. When you click the **Pattern** button, the **Ins Statement** dialog box appears, as shown in Figure 13.17:
7. In the **Ins Statement** dialog box, write the name of the function module (`REUSE_ALV_GRID_DISPLAY`) in the **CALL FUNCTION** field and click the **Continue** (✓) icon (see Figure 13.17). The code given in Listing 13.3, for the `REUSE_ALV_GRID_DISPLAY` function module, is displayed in the ABAP Editor:

Listing 13.3: Sequence of statements for the ALV report

```

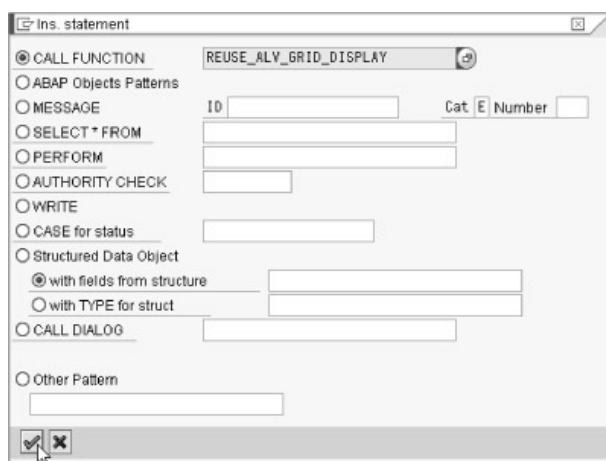
*-----*
*/ Data Declarations for the Program
DATA: I_QMEL LIKE QMEL OCCURS 0.
DATA: WS_TITLE type LVC_TITLE VALUE 'AN ALV REPORT CREATED
BY SHIVAM SRIVASTAVA'.
*-----*
*/ Selecting Fields from QMEL Table
*/
SELECT * FROM QMEL INTO TABLE I_QMEL
*-----*
*/ Calling a Function Module 'REUSE_ALV_GRID_DISPLAY'
*/
CALL FUNCTION 'REUSE_ALV_GRID_DISPLAY'
EXPORTING
*      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            = 'QMEL'
*      I_BACKGROUND_ID             = ' '

```

```

      I_GRID_TITLE                = WS_TITLE
*      I_GRID_SETTINGS            =
*      IS_LAYOUT                  =
*      IT_FIELDCAT                =
*      IT_EXCLUDING               =

*      IT_SPECIAL_GROUPS          =
*      IT_SORT                    =
*      IT_FILTER                  =
*      IS_SEL_HIDE =
      I_DEFAULT                    = 'X'
*      I_SAVE                     = ' '
*      IS_VARIANT                 =
*      IT_EVENTS                  =
*      IT_EVENT_EXIT              =
*      IS_PRINT                   =
*      IS_REPREP_ID               =
*      I_SCREEN_START_COLUMN      = 0
*      I_SCREEN_START_LINE        = 0
*      I_SCREEN_END_COLUMN        = 0
*      I_SCREEN_END_LINE          = 0
*      I_HTML_HEIGHT_TOP          = 0
*      I_HTML_HEIGHT_END          = 0
*      IT_ALV_GRAPHICS            =
*      IT_HYPERLINK               =
*      IT_ADD_FIELDCAT            =
*      IT_EXCEPT_QINFO          =
*      IR_SALV_FULLSCREEN_ADAPTER =
*      IMPORTING
*      E_EXIT_CAUSED_BY_CALLER    =
*      ES_EXIT_CAUSED_BY_USER    =
      TABLES
      T_OUTTAB                    = I_QMEL
*      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.
*-----*
```



© SAP AG. All rights reserved.

Figure 13.17: Calling the function module

Figure 13.18 shows the code given in Listing 13.3 in the ABAP Editor screen:

8. Click the Save (📁) icon and then click the Check (🔍) icon. Finally, click the Activate (🔑) icon to activate the program before executing it. Now, click the Direct processing (🖨️) icon to execute the created ALV report. Figure 13.19 shows the generated ALV report:

ABAP Editor: Change Report ZALVREPORT

Report ZALVREPORT Inactive (Revised)

```
%&
%& Report ZALVREPORT
%& Date: 11-03-2009
%&-----
%& Description: ALV Report
%&-----
%&

REPORT ZALVREPORT.

%&-----
%& / Data Declarations for the Program
%& /
DATA: I_QMEL LIKE QMEL OCCURS 0.
DATA: WS_TITLE type LVC_TITLE VALUE 'AN ALV REPORT CREATED BY SHIVAM SRIVASTAVA'.
%&-----
%& / Selecting Fields from QMEL Table
%& /
SELECT * FROM QMEL INTO TABLE I_QMEL.
%&-----
%& / Calling a Function Module 'REUSE_ALV_GRID_DISPLAY'
%& /
CALL FUNCTION 'REUSE_ALV_GRID_DISPLAY'
EXPORTING
  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            = 'QMEL'
  I_BACKGROUND_ID              = ''
  I_GRID_TITLE                 = WS_TITLE
  T_EXIT_SETTINGS              = ''
```

© SAP AG. All rights reserved.

Figure 13.18: The ABAP editor screen containing statements for the ALV report

List Edit Goto Views Settings System Help

Example of ALV Reports

AN ALV REPORT CREATED BY SHIVAM SRIVASTAVA

Notification	T	Description	P...	P	Created by	Created on	Changed by	Changed on	Notif time	Notif date	Reported by
75	Q1		QM		RICKAYZEN	11.01.1996	BARTHEL	13.01.1997	13.16.42	11.01.1996	RICKAYZEN
84	Q1	Too impure to use	QM		RICKAYZEN	12.01.1996	WF-QM-2	12.01.1996	09.37.25	12.01.1996	RICKAYZEN
85	Q1	Not pure enough	QM		RICKAYZEN	12.01.1996	BARTHEL	13.01.1997	09.40.01	12.01.1996	RICKAYZEN
94	Q1	Check the ration of the mix	QM 1		RICKAYZEN	26.03.1996	WF-QM-1	26.03.1996	15.30.56	26.03.1996	RICKAYZEN
95	Q1	The mix is not correct	QM		RICKAYZEN	26.03.1996	WF-QM-2	26.03.1996	15.58.13	26.03.1996	RICKAYZEN
124	F2	Kraber und Fingersdrücke festgestellt	QM 2		WESTERMANN	10.05.1996	WESTERMANN	15.05.1996	17.05.40	10.05.1996	WESTERMANN
144	F3		QM		KEILBACH	31.07.1996	BARTHEL	18.12.1996	19.01.55	31.07.1996	KEILBACH
154	F2	Wider keine Handschuhe benutzt	QM 2		WESTERMANN	12.12.1996	WESTERMANN	12.12.1996	18.35.15	12.12.1996	WESTERMANN
164	F3		QM		BARTHEL	20.12.1996	BARTHEL	20.12.1996	10.36.14	20.12.1996	BARTHEL
204	F2	Identification of screws not conforming	QM 3		WESTERMANN	12.04.2000	WESTERMANN	13.04.2000	20.47.34	12.04.2000	WESTERMANN
205	F2	Missing identification of screws	QM 4		WESTERMANN	12.04.2000	WESTERMANN	13.04.2000	20.48.17	12.04.2000	WESTERMANN
206	F2		QM		WESTERMANN	12.04.2000			20.57.41	12.04.2000	WESTERMANN
214	F2		QM		WESTERMANN	13.04.2000			12.09.25	13.04.2000	WESTERMANN
215	F2		QM		WESTERMANN	13.04.2000	WESTERMANN	14.04.2000	15.05.28	13.04.2000	
216	F2		QM		WESTERMANN	13.04.2000	WESTERMANN	13.04.2000	15.22.18	13.04.2000	
217	F2		QM		WESTERMANN	13.04.2000	WESTERMANN	13.04.2000	16.32.19	13.04.2000	WESTERMANN
218	F2	Problem mit Schrauben M8 x 40	QM 2		WESTERMANN	13.04.2000	WESTERMANN	14.04.2000	16.38.36	13.04.2000	WESTERMANN
219	F2	Zinküberzug porös	QM 3		WESTERMANN	13.04.2000	WESTERMANN	14.04.2000	16.55.23	13.04.2000	WESTERMANN
220	F2		QM		WESTERMANN	13.04.2000	WESTERMANN	13.04.2000	17.13.15	13.04.2000	WESTERMANN
221	F2		QM		WESTERMANN	14.04.2000	WESTERMANN	14.04.2000	13.38.57	14.04.2000	WESTERMANN
222	F2	Schraubenschaft geringfügig zu lang	QM		WESTERMANN	14.04.2000	WESTERMANN	14.04.2000	14.04.33	14.04.2000	
223	F2		QM		WESTERMANN	14.04.2000	WESTERMANN	14.04.2000	14.47.07	14.04.2000	
224	F2	Schrauben haben Grat am Gewindeende	QM 3		WESTERMANN	14.04.2000	WESTERMANN	14.04.2000	19.54.49	14.04.2000	WESTERMANN
244	F2	Kenzeichnungproblem	QM 3		WESTERMANN	10.08.2000	WESTERMANN	10.08.2000	14.28.54	10.08.2000	WESTERMANN
245	F2		QM		WESTERMANN	10.08.2000	WESTERMANN	10.08.2000	15.38.58	10.08.2000	
246	F2	Mängel an Sechskantschrauben	QM 3		WESTERMANN	10.08.2000	WESTERMANN	10.08.2000	16.32.47	10.08.2000	

Program 'REPSZALVREPORT' stored temporarily

DMT (1) 800 sapmachine INS

© SAP AG. All rights reserved.

Figure 13.19: The generated ALV report

In [Figure 13.19](#), you see the required data, such as Notification and Description, corresponding to the QMEL table in grid view. You can perform various functions, such as sorting and filtering in the generated ALV report.

Summary

In this chapter, you have learned about reports used to retrieve data from one or more tables in a database without making any changes in the tables. You have also learned about the different types of reports, such as classical reports, interactive reports, and ALV reports, and how to create each of them. In addition, you have learned about the different types of events used to create these reports, such as `INITIALIZATION`, `AT SELECTION-SCREEN`, and `START-OF-SELECTION`. You have learned about differences between classical and interactive reports. Finally, you have explored the function modules used to create ALV reports.