Objective

Extract datasets

- **Filling an extract with data**
- **Reading an extract**
- **Sorting an Extract**

Lists

- **Standard list**
- **Self-Defined list**
- **List with several pages**

Interactive lists

- **At line-selection**
- **At PF <nn>**
- **At User-command**

Refining Data

A report must sort data,calculate totals,count item in lists and so on.

you can read the data to be refined from database tables or from sequential files or u can create generic report.

The refining process is independent of the process of retrieving data.

Create data set

Refine the data set

Creating and Refining datasets:

Abap/4 offers two methods of creating datasets in the storage

Internal tables

.

Extract datasets

Internal tables :

If you want the datasets to map the underlying data structures as closely as possible and if you want to access individual data directly.

Extract datasets :

An extract is a sequential dataset you can create with a report.Use extracts if you want to process large amount of data as a whole several times

Refining data using EXTRACT DATASETS:

An Extract is a sequential dataset in the report's storage area.this means that you can access its data only within a loop.

During the report's run time,the system can create exactly one extract dataset.

As for internal tables the size of the extract data set is principally unlimited,since the system rolls it out if necessary.

In one extract dataset,you can store records of different length and structure one after the other.

Declaring Extract Records as Field Groups

An extract dataset consists of a sequence of records. These records may have different structures. All records with the same structure form a record type. You must define each record type of an extract dataset as a field group, using the FIELD-GROUPS statement.

Syntax

FIELD-GROUPS <fg>.

Filling an Extract with Data

Once you have declared the possible record types as field groups and defined their structure, you can fill the extract dataset using the following statements:

Syntax

EXTRACT <fg>.

When the first EXTRACT statement occurs in a program, the system creates the extract dataset and adds the first extract record to it. In each subsequent EXTRACT statement, the new extract record is added to the dataset

Reading an Extract

Like internal tables, you can read the data in an extract dataset using a loop.

syntax

LOOP.

…

[AT FIRST | AT <fg$_i$> [WITH <fg $_j$>] | AT LAST.

...

ENDAT.]

...

ENDLOOP.

Sorting an Extract

You can sort an extract dataset in much the same way as an internal table by using the following statement:

syntax

SORT [ASCENDING|DESCENDING] [AS TEXT] [STABLE]
BY <f_1> [ASCENDING|DESCENDING] [AS TEXT]
...
<f_n> [ASCENDING|DESCENDING] [AS TEXT].

Processing Control Levels

When you sort an extract dataset, control levels are defined in it. For general information about control levels, The control level hierarchy of an extract dataset corresponds to the sequence of the fields in the HEADER field group. After sorting, you can use the AT statement within a loop to program statement blocks that the system processes only at a control break, that is, when the control level changes.

Syntax

```
AT NEW <f> | AT END OF <f>.
...
ENDAT.
```

Lists

Lists are the output medium for data from ABAP/4 report programs. Each program can produce up to 21 lists: one basic list and 20 secondary lists.

Structure and the options of a list are

The Standard List
The Self-Defined List
Lists with Several Pages

The standard list

If an ABAP/4 report makes use only of the WRITE, SKIP, and ULINE output statements .This list is called standard list.

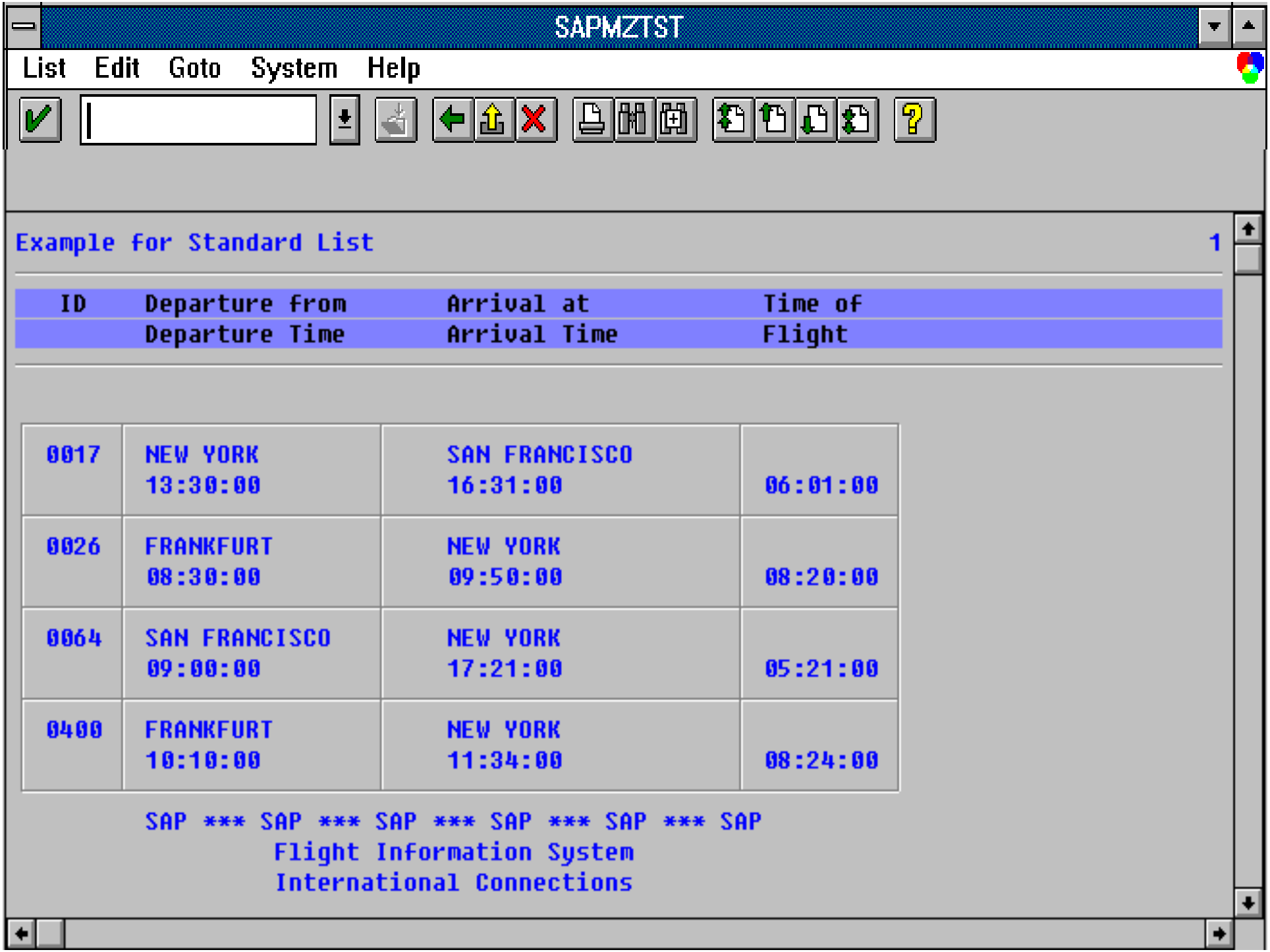
The standard list consists of:

Standard Page Header

Standard Page

User Interface of the Standard List

The next slide shows a standard list.


SAPMZTST
List Edit Goto System Help
Example for Standard List 1
ID Departure from Arrival at Time of
Departure Time Arrival Time Flight
0017 NEW YORK SAN FRANCISCO
13:30:00 16:31:00 06:01:00
0026 FRANKFURT NEW YORK
08:30:00 09:50:00 08:20:00
0064 SAN FRANCISCO NEW YORK
09:00:00 17:21:00 05:21:00
0400 FRANKFURT NEW YORK
10:10:00 11:34:00 08:24:00
SAP *** SAP *** SAP *** SAP *** SAP *** SAP
Flight Information System
International Connections

The Self-Defined List

Self-defined list is a list created by modifying the standard list by using options of the REPORT statement and using event keywords TOP-OF-PAGE and END-OF-PAGE.

The possible modifications that can be performed in a standard list are :

Page Header
List Width
Page Length
Page Footer

Page Header

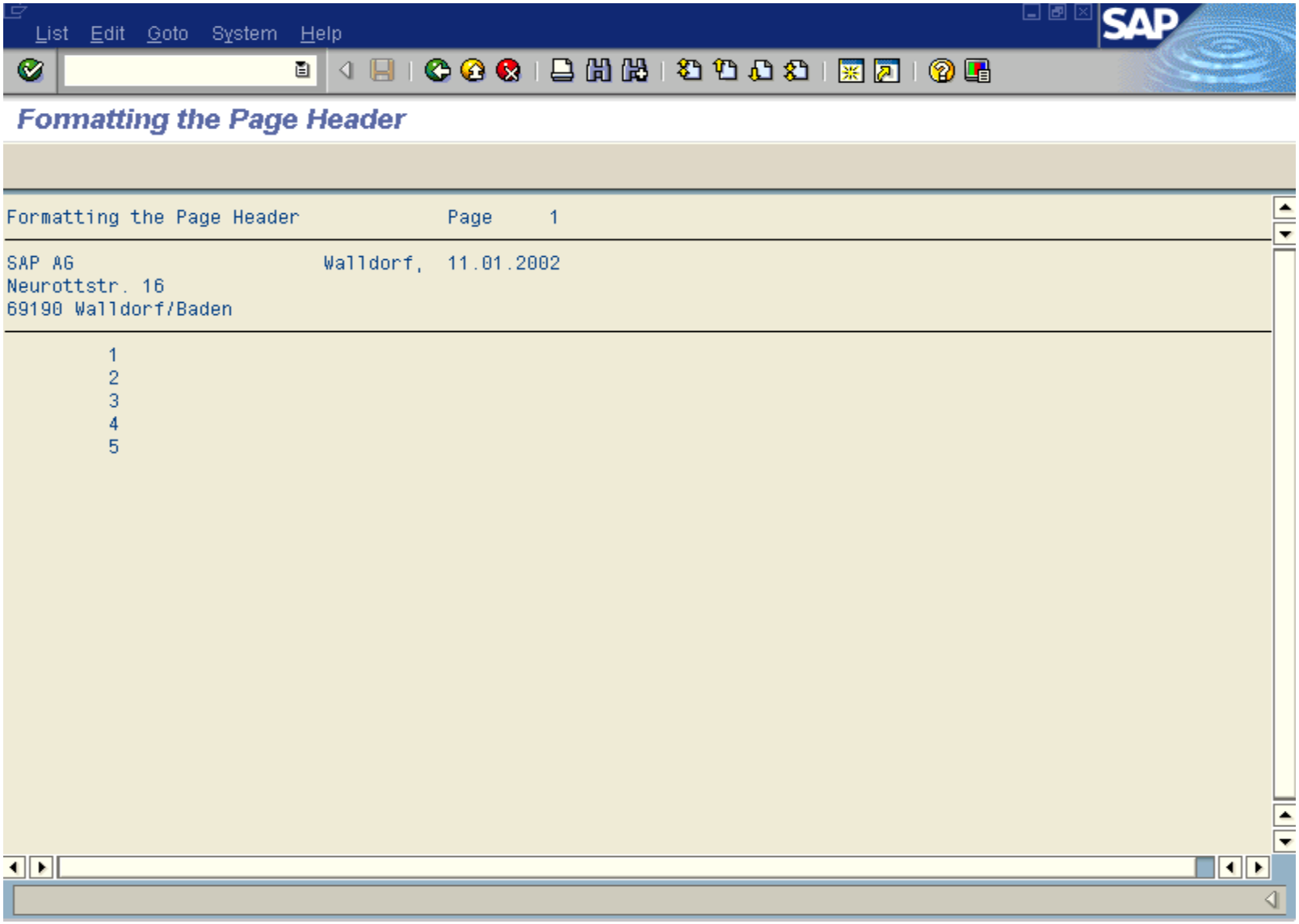
To layout a page header individually, you must define it in the processing block following the event keyword TOP-OF-PAGE:

Syntax
TOP-OF-PAGE.
WRITE: ....

The TOP-OF-PAGE event occurs as soon as the system starts processing a new page of a list and before outputting the first line on a new page.

Note :
The self-defined page header appears beneath the standard page header. If you want to suppress the standard page header, use the NO STANDARD PAGE HEADING option of the REPORT statement.

```
REPORT demo_list_page_heading NO STANDARD PAGE HEADING.

TOP-OF-PAGE.

  WRITE: sy-title, 40 'Page', sy-pagno.
  ULINE.
  WRITE: / 'SAP AG', 29 'Walldorf, ',sy-datum,
         / 'Neurottstr. 16', / '69190 Walldorf/Baden'.
  ULINE.

START-OF-SELECTION.

  DO 5 TIMES.
    WRITE / sy-index.
  ENDDO.
```


List Edit Goto System Help
SAP
Formatting the Page Header
Formatting the Page Header Page 1
SAP AG Walldorf, 11.01.2002
Neurottstr. 16
69190 Walldorf/Baden
1
2
3
4
5

List Width

To determine the width of the output list, use the LINE-SIZE option of the REPORT statement.

Syntax

REPORT <rep> LINE-SIZE <width>.

Note :

•If you set <width> to 0, the system uses the width of the standardlist .
•The system field SY-LINSZ contains the current line width

Page Length

To determine the page length of an output list, use the LINE-COUNT option of the REPORT statement.

Syntax

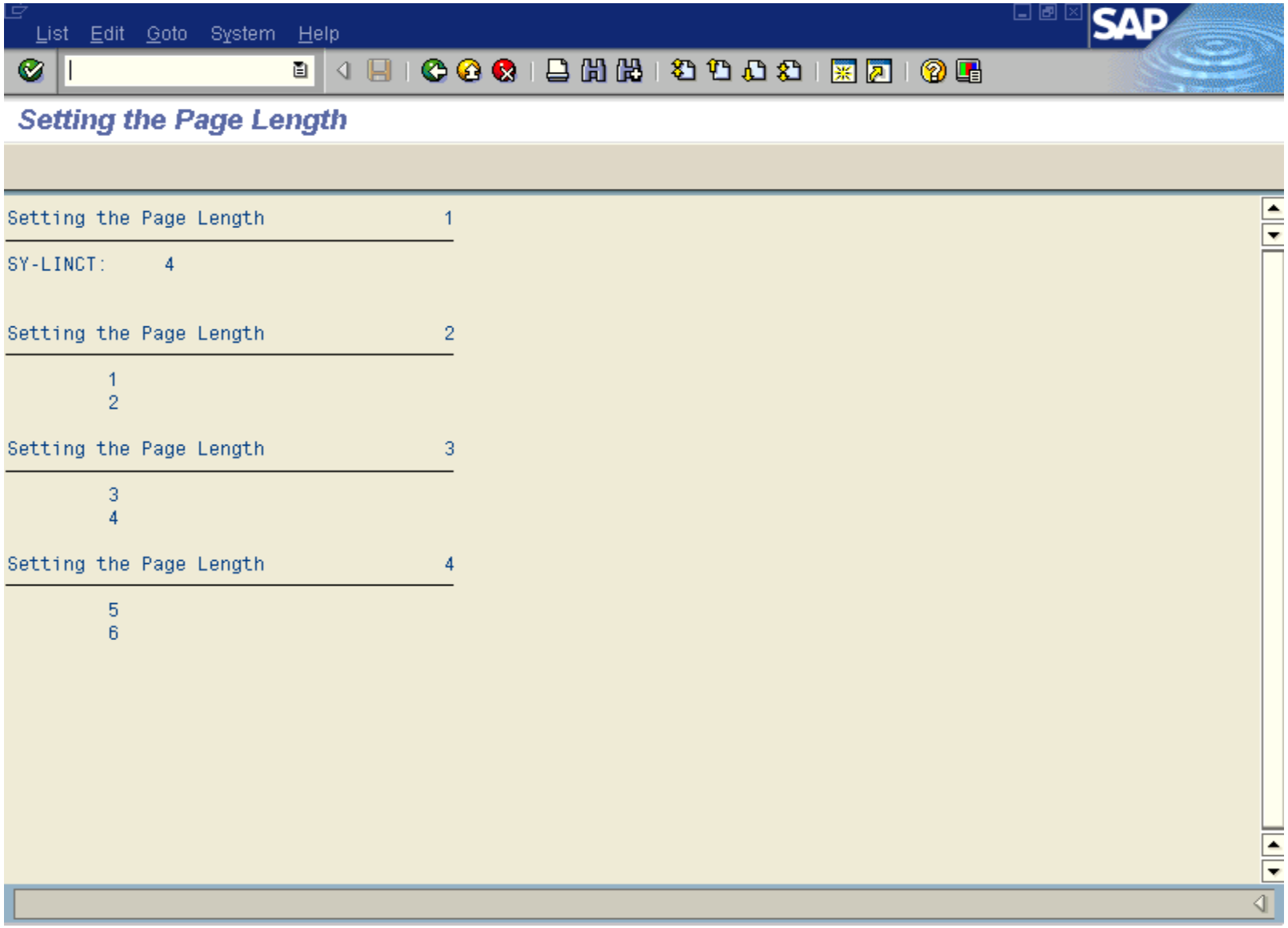
REPORT <rep> LINE-COUNT <length>[(<n>)].

<n> ---->The system reserves <n> lines of the page length for the page footer.

Note :

- **If you set <length> to zero, the system uses the standard page length .**
- **The system field SY-LINCT contains the current number of lines per page**

```
REPORT demo_list_line_count LINE-SIZE 40 LINE-
COUNT 4.


WRITE:  'SY-LINCT:', sy-linct.
SKIP.


DO 6 TIMES.
  WRITE / sy-index.
ENDDO.
```


List Edit Goto System Help
SAP
Setting the Page Length
Setting the Page Length 1
SY-LINCT: 4
Setting the Page Length 2
1
2
Setting the Page Length 3
3
4
Setting the Page Length 4
5
6

Page Footer

To define a page footer, use the END-OF-PAGE event.

Syntax

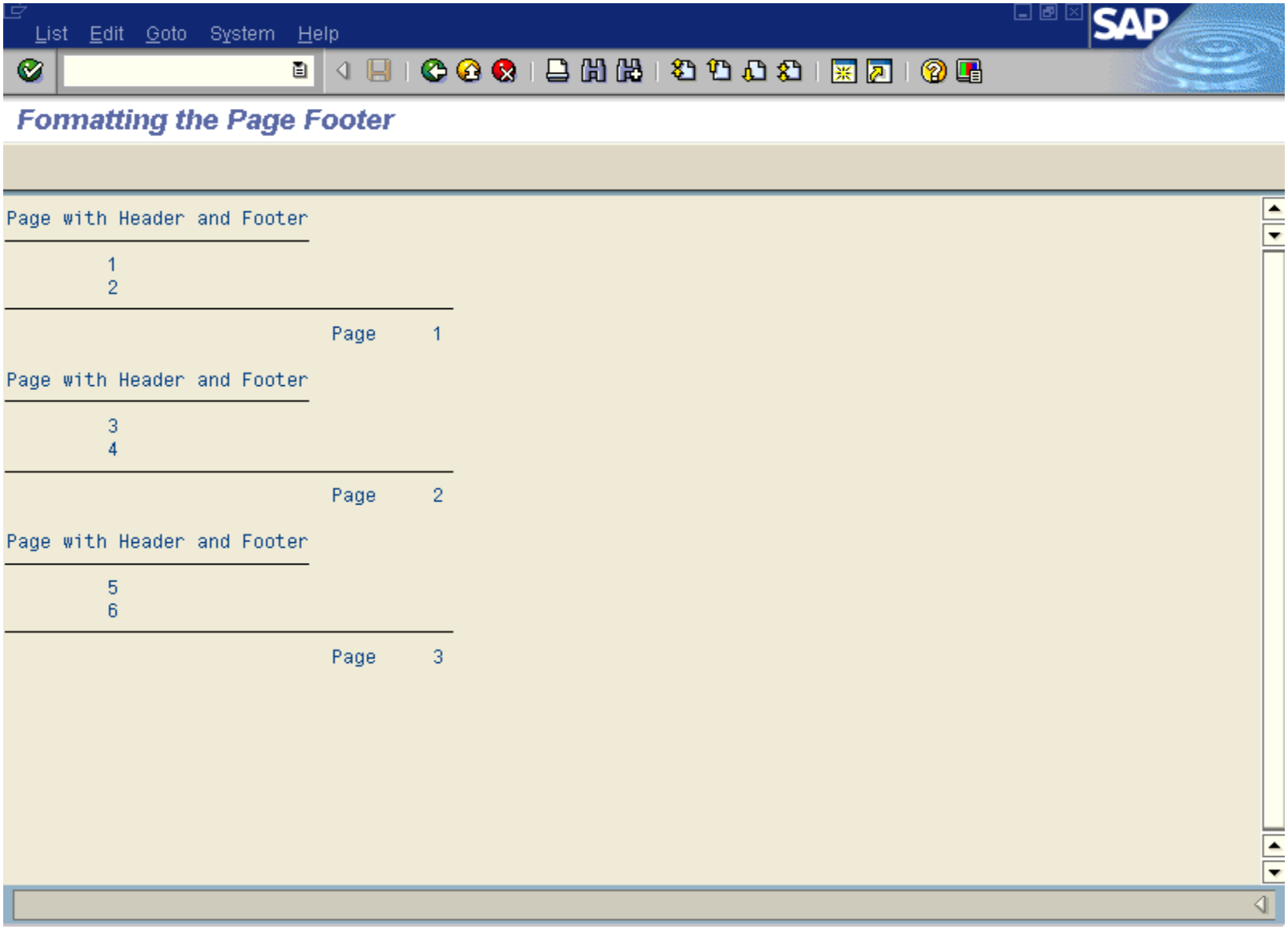
```
END-OF-PAGE.
WRITE: ....
```

This event occurs when the system reaches the lines reserved for the page footer, or if the RESERVE statement triggers a page break.

Note :

The system only processes the processing block following END-OF-PAGE if you reserve lines for the footer in the LINE-COUNT option of the REPORT statement .

```
REPORT demo_list_end_of_page LINE-SIZE 40 LINE-COUNT 6(2)
                             NO STANDARD PAGE HEADING.

TOP-OF-PAGE.

  WRITE: 'Page with Header and Footer'.
  ULINE AT /(27).

END-OF-PAGE.

  ULINE.
  WRITE: /30 'Page', sy-pagno.

START-OF-SELECTION.

  DO 6 TIMES.
    WRITE / sy-index.
  ENDDO.
```


List Edit Goto System Help
SAP
Formatting the Page Footer
Page with Header and Footer
1
2
Page 1
Page with Header and Footer
3
4
Page 2
Page with Header and Footer
5
6
Page 3

Lists with Several Pages

If in your report has more number of lines than defined in the LINE-COUNT option of the REPORT statement, the system automatically creates a new page .

Apart from automatic page breaks, you can use the NEW-PAGE and RESERVE statements to code page breaks explicitly.

Page Break- Conditional

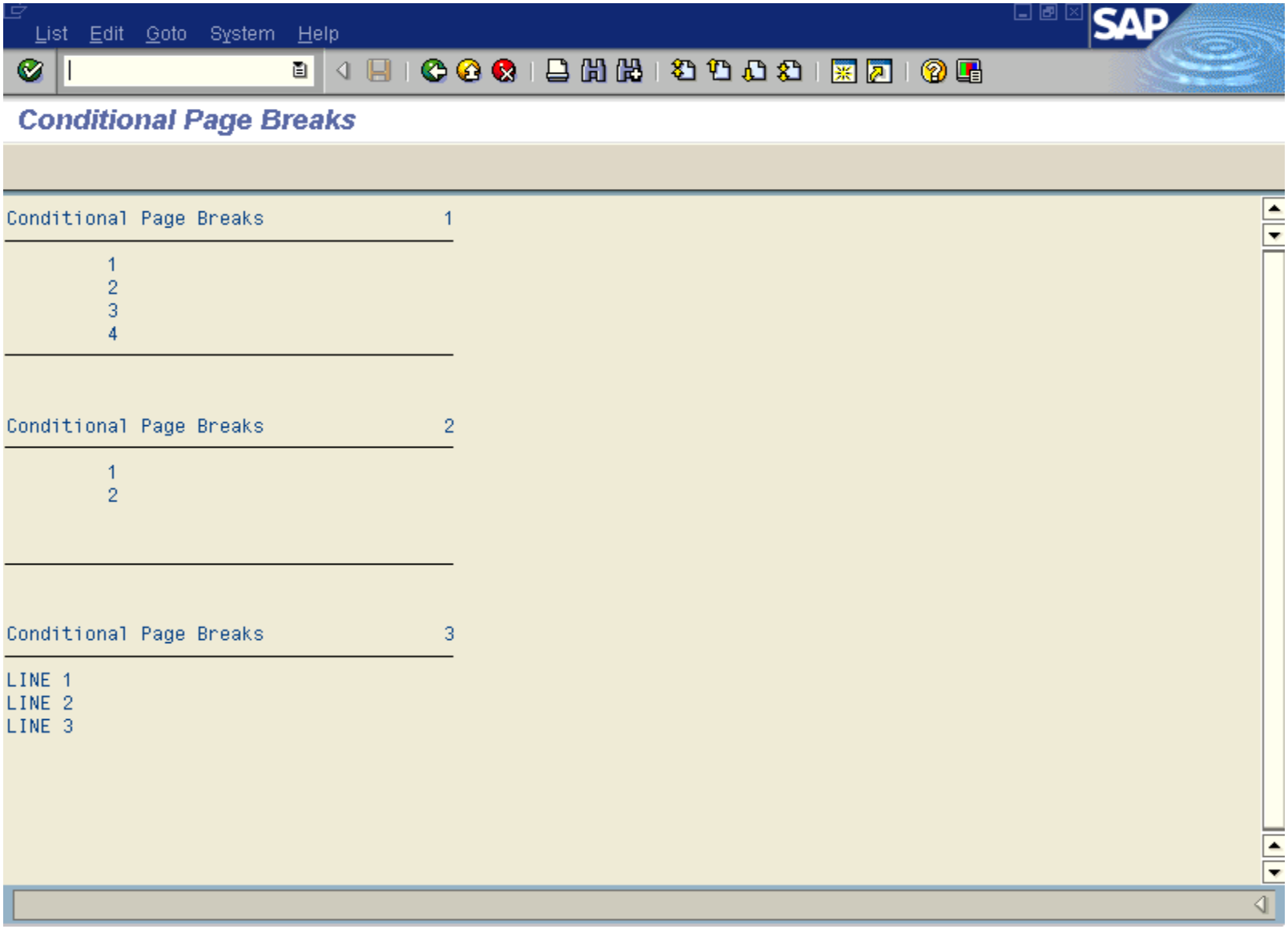
To execute a page break under the condition that less than a certain number of lines is left on a page, use the RESERVE statement:

Syntax

RESERVE <n> LINES.

This statement triggers a page break if less than <n> free lines are left on the current list page between the last output and the page footer. Before starting a new page, the system processes the END-OF-PAGE event.

```
REPORT demo_list_reserve LINE-SIZE 40 LINE-COUNT 8(2).

END-OF-PAGE.

  ULINE.

START-OF-SELECTION.

  DO 4 TIMES.
    WRITE / sy-index.
  ENDDO.

  DO 2 TIMES.
    WRITE / sy-index.
  ENDDO.

  RESERVE 3 LINES.
  WRITE: / 'LINE 1',
         / 'LINE 2',
         / 'LINE 3'.
```


List Edit Goto System Help
SAP
Conditional Page Breaks
Conditional Page Breaks 1
1
2
3
4
Conditional Page Breaks 2
1
2
Conditional Page Breaks 3
LINE 1
LINE 2
LINE 3

Page Break- Unconditional

Syntax

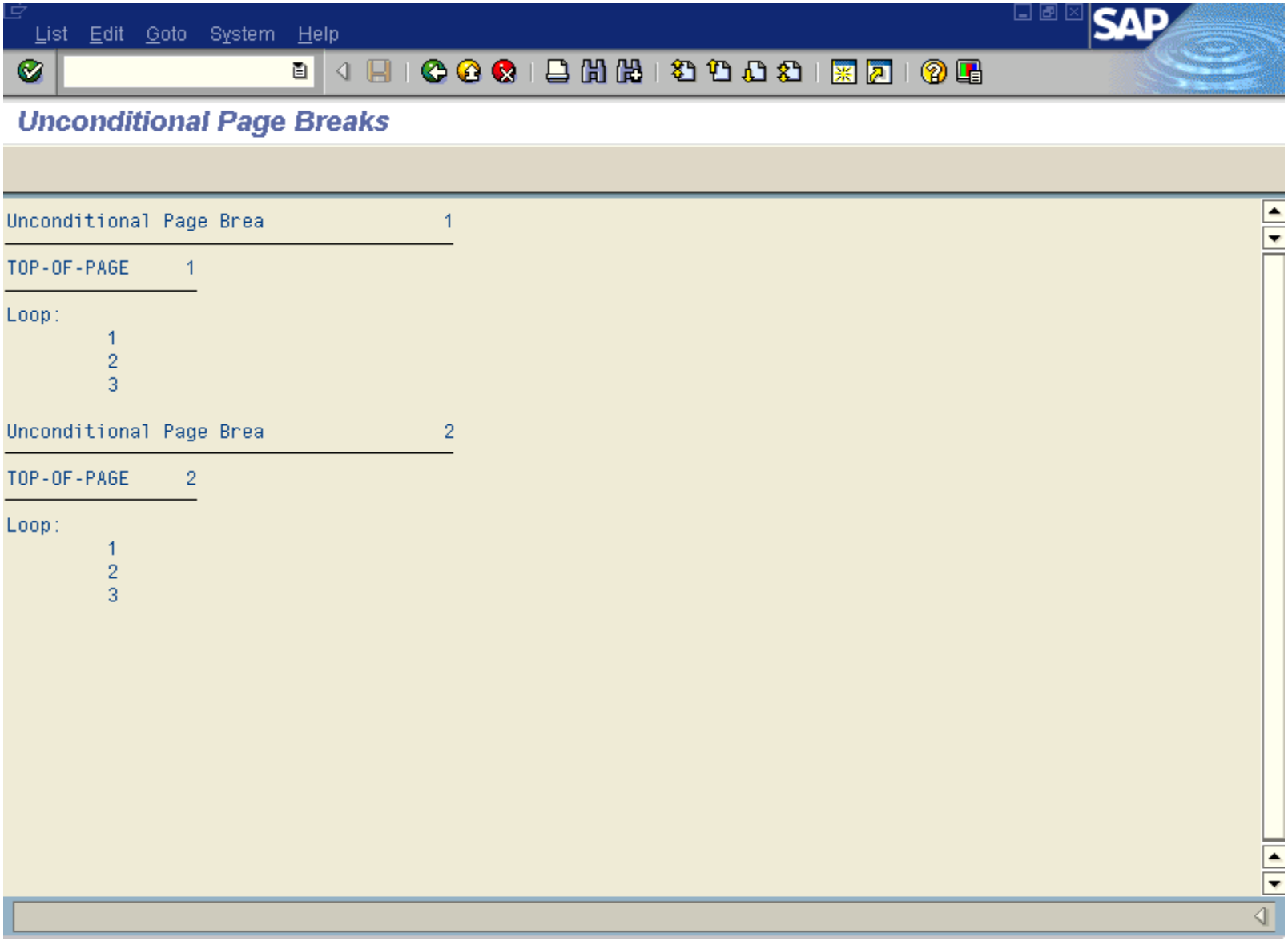
NEW-PAGE.

*** Ends the current page. All other output appears on a new page.**
*** The system then increases the SY-PAGNO system field by one.**
*** Does not trigger the END-OF-PAGE event.**

Variants in NEW_PAGE are:

NEW-PAGE [NO-TITLE|WITH-TITLE] NO-HEADING|WITH-HEADING].
NEW-PAGE LINE-COUNT <length>.
NEW-PAGE LINE-SIZE <width>.

```
REPORT demo_list_new_page LINE-SIZE 40.

TOP-OF-PAGE.

  WRITE: 'TOP-OF-PAGE', sy-pagno.
  ULINE AT /(17).

START-OF-SELECTION.

  DO 2 TIMES.
    WRITE / 'Loop:'.

   DO 3 TIMES.
      WRITE / sy-index.
    ENDDO.

 NEW-PAGE.
  ENDDO.
```


List Edit Goto System Help
SAP
Unconditional Page Breaks
Unconditional Page Brea 1
TOP-OF-PAGE 1
Loop:
1
2
3
Unconditional Page Brea 2
TOP-OF-PAGE 2
Loop:
1
2
3

Scrolling from within the Program

From within the program, you can scroll through lists vertically and horizontally.

The SCROLL statement allows you:

Vertical Scrolling

Scrolling Window by Window
Scrolling by Pages

Horizontal Scrolling

Scrolling to the List's Margins
Scrolling by Columns

Scrolling Window by Window

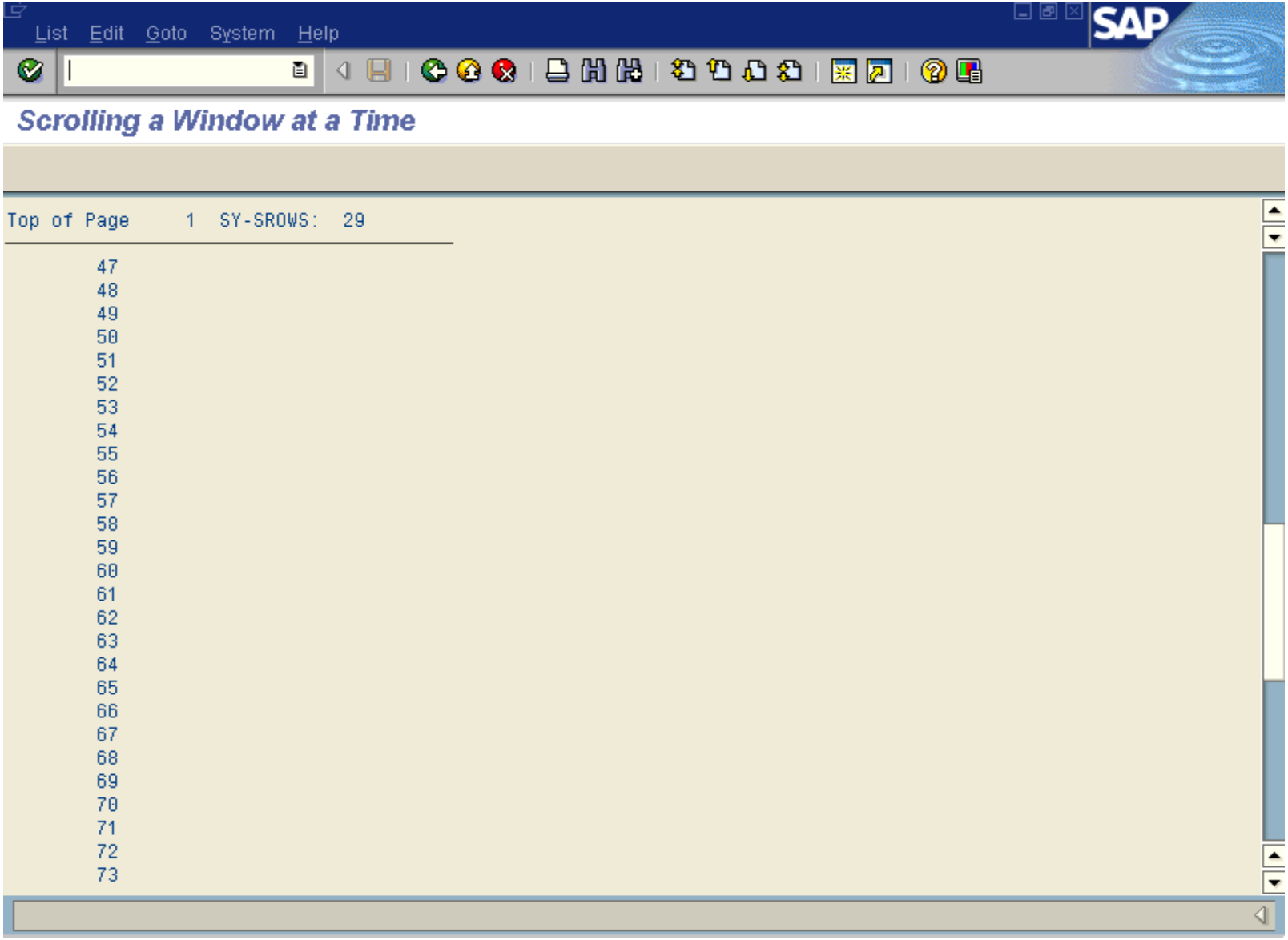
To scroll through a list vertically by the size of the current window use this statement:

Syntax

SCROLL LIST FORWARD|BACKWARD.

This statement scrolls forward or backward through the current list by the size of the current window.

```
REPORT demo_list_scroll_1 NO STANDARD PAGE HEADING LINE-SIZE 40.


TOP-OF-PAGE.

  WRITE: 'Top of Page', sy-pagno, 'SY-SROWS:', sy-srows.
  ULINE.


START-OF-SELECTION.


  DO 100 TIMES.
    WRITE / sy-index.
  ENDDO.


  DO 3 TIMES.
    SCROLL LIST FORWARD.
  ENDDO.
```


List Edit Goto System Help
SAP
Scrolling a Window at a Time
Top of Page 1 SY-SROWS: 29
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73

Scrolling by Pages

To scroll a list vertically depending on the page length, the SCROLL statement offers two options.

Scrolling to Certain Pages

Scrolling by a Certain Number of Pages

Scrolling to Certain Pages

To scroll to certain pages, use the TO option of the SCROLL statement:

Syntax

SCROLL LIST TO FIRST PAGE | LAST PAGE | PAGE <page> .

This statement scrolls the current list to the first, to the last, or to the page numbered <page> .

Scrolling by a Certain Number of Pages

To scroll a list by a certain number of pages, use the following options of the SCROLL statement:

Syntax

SCROLL LIST FORWARD | BACKWARD <n> PAGES.

This statement scrolls forward or backward <n> pages.

Scrolling to the List's Margins

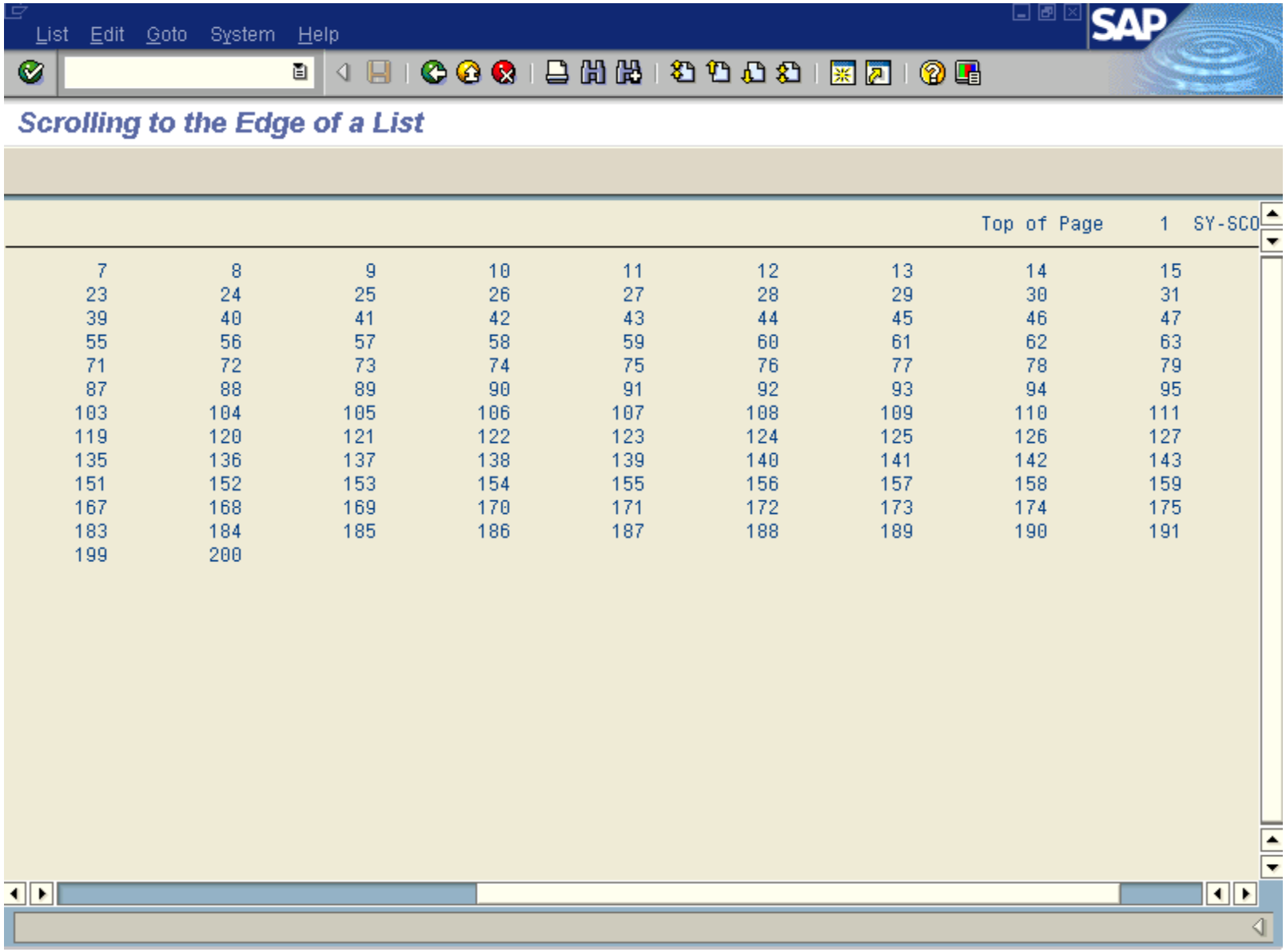
To scroll horizontally to the left or right margin of a list, use the following options of the SCROLL statement:

Syntax

SCROLL LIST LEFT | RIGHT.

This statement scrolls to the left or right margin of the currentlist.

```
REPORT demo_list_scroll_3 NO STANDARD PAGE HEADING LINE-SIZE 200.

TOP-OF-PAGE.


  WRITE: AT 161 'Top of Page', sy-pagno,

                'SY-SCOLS:', sy-scols.

  ULINE.

START-OF-SELECTION.

  DO 200 TIMES.

    WRITE sy-index.
  ENDDO.

  SCROLL LIST RIGHT.
```


List Edit Goto System Help
SAP
Scrolling to the Edge of a List
Top of Page 1 SY-SCO
7 8 9 10 11 12 13 14 15
23 24 25 26 27 28 29 30 31
39 40 41 42 43 44 45 46 47
55 56 57 58 59 60 61 62 63
71 72 73 74 75 76 77 78 79
87 88 89 90 91 92 93 94 95
103 104 105 106 107 108 109 110 111
119 120 121 122 123 124 125 126 127
135 136 137 138 139 140 141 142 143
151 152 153 154 155 156 157 158 159
167 168 169 170 171 172 173 174 175
183 184 185 186 187 188 189 190 191
199 200

Scrolling by Columns

To scroll a list horizontally by columns, the SCROLL statement offers two options. A column in this case means one character of the list line.

- **Scrolling to Certain Columns**
- **Scrolling by a Certain Number of Columns**

Scrolling to Certain Columns

To scroll to certain columns, use the TO COLUMN option of the SCROLL statement:

Syntax

SCROLL LIST TO COLUMN <colm> .

This system displays the current list starting from column <colm>.

Scrolling by a Certain Number of Columns

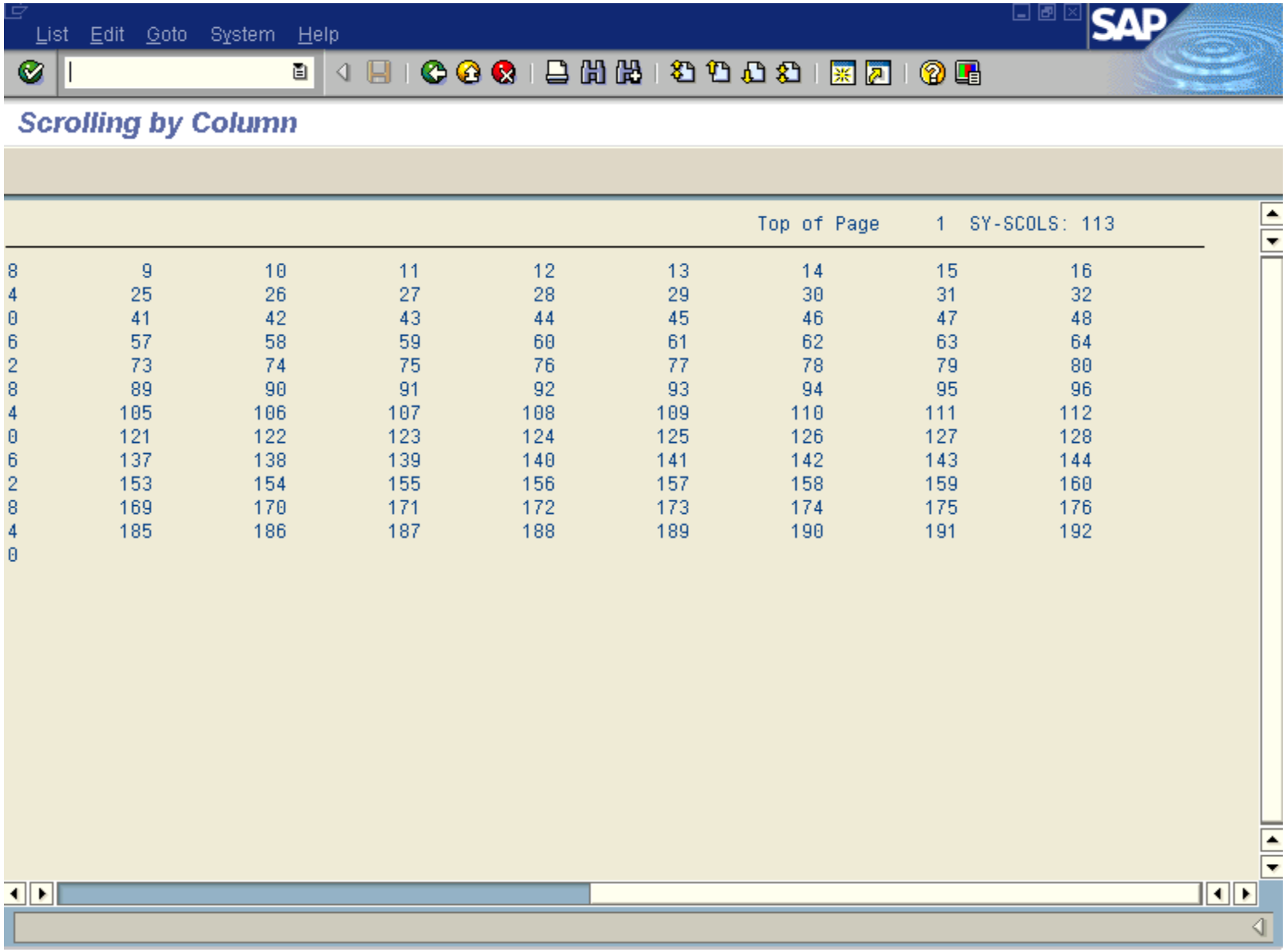
To scroll a list by a certain number of columns, use the following option of the SCROLL statement:

Syntax

SCROLL LIST LEFT | RIGHT BY <n> PLACES.

This system scrolls the current list to the left or right by <n> columns.

```
REPORT demo_list_scroll_4 NO STANDARD PAGE HEADING LINE-SIZE
200.

TOP-OF-PAGE.

  WRITE: AT 161 'Top of Page', sy-pagno,
                'SY-SCOLS:', sy-scols.
  ULINE.

START-OF-SELECTION.

  DO 200 TIMES.
    WRITE sy-index.
  ENDDO.

  SCROLL LIST TO COLUMN 178.
```


List Edit Goto System Help
SAP
Scrolling by Column
Top of Page 1 SY-SCOLS: 113
8 9 10 11 12 13 14 15 16
4 25 26 27 28 29 30 31 32
0 41 42 43 44 45 46 47 48
6 57 58 59 60 61 62 63 64
2 73 74 75 76 77 78 79 80
8 89 90 91 92 93 94 95 96
4 105 106 107 108 109 110 111 112
0 121 122 123 124 125 126 127 128
6 137 138 139 140 141 142 143 144
2 153 154 155 156 157 158 159 160
8 169 170 171 172 173 174 175 176
4 185 186 187 188 189 190 191 192
0

Left Boundary for Horizontal Scrolling

To determine the left boundary of the horizontally scrollable area, use:

Syntax

SET LEFT SCROLL-BOUNDARY [COLUMN <col>].

Excluding Lines from Horizontal Scrolling

To exclude a line (for example, a header or comment line) from horizontal scrolling, define the line feed for that line as follows:

Syntax

NEW-LINE NO-SCROLLING.

The line following the statement cannot be scrolled horizontally. However, it can be scrolled vertically.

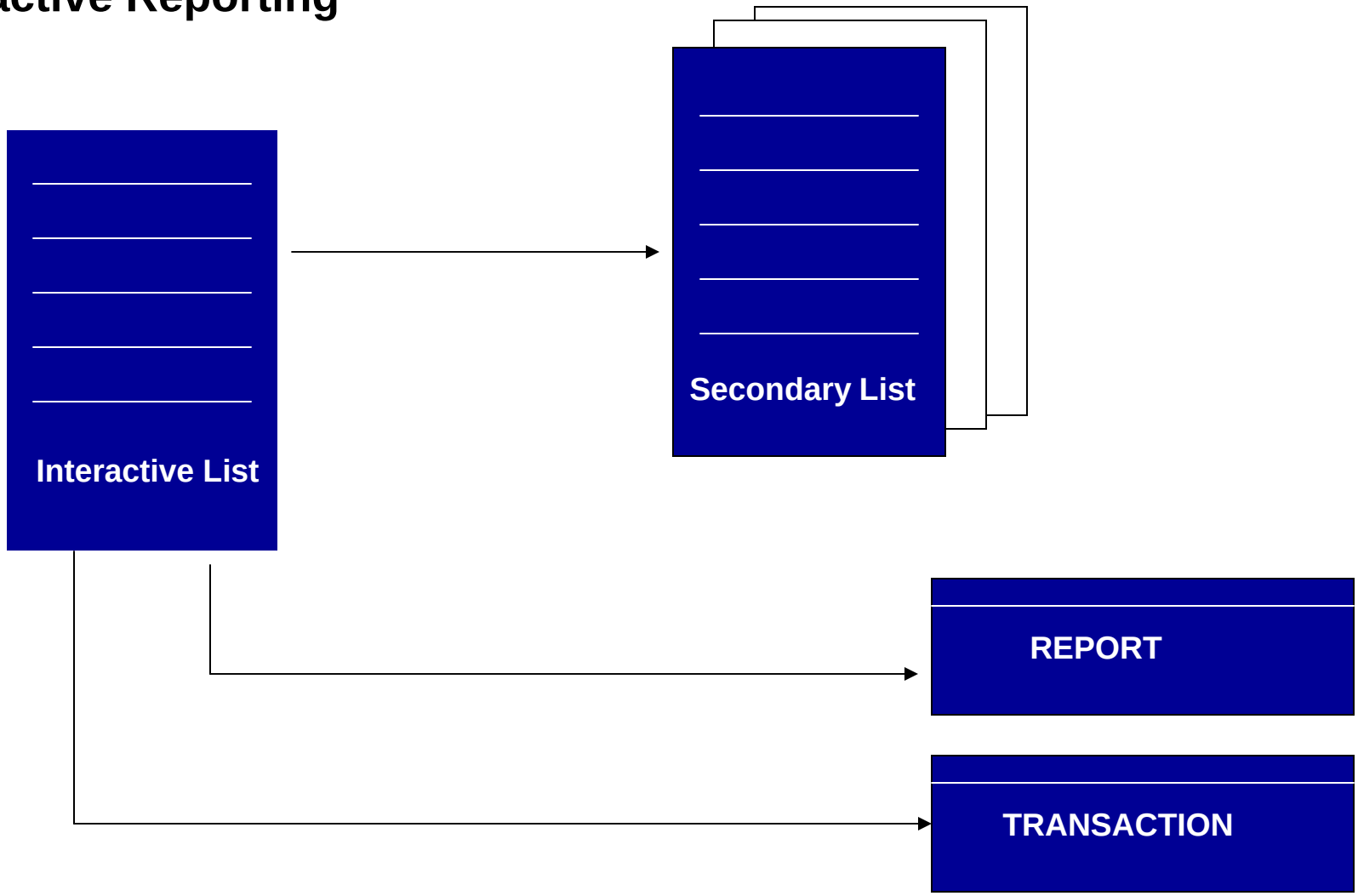
A list is an interactive list if the user interface allows actions that trigger events and if the corresponding interactive event keywords occur in the report.

All lists created during an interactive list event are secondary lists. Interactive lists enhance the classical type of output list with dialog functionality,thus coming close to dialog programming. Interactive lists provide the user with so-called “interactive reporting” facility.

Interactive reporting allows the user to participate actively in retrieving and presenting data during the the session. Instead of one extensive and detailed list,with interactive reporting you create a condensed basic list from which the user can call detailed information by positioning the cursor and entering commands.

Interactive reporting thus reduces information retrieval to the data actually required.
Detailed information is presented in secondary lists.

Interactive Reporting



Events for Interactive Lists

- AT LINE-SELECTION
- AT PF<nn>
- AT USER-COMMAND

Allowing Line Selection

To allow the user to select a line from the list, define and write a processing block for the AT LINE-SELECTION event in your program:

Syntax

```
AT LINE-SELECTION.
 <statements>.
```

In the predefined interface, Edit --> Choose and F2 are assigned to PICK.

Page Headers for Secondary Lists

On secondary lists, the system does not trigger the event TOP-OF-PAGE. To create page headers for secondary list:

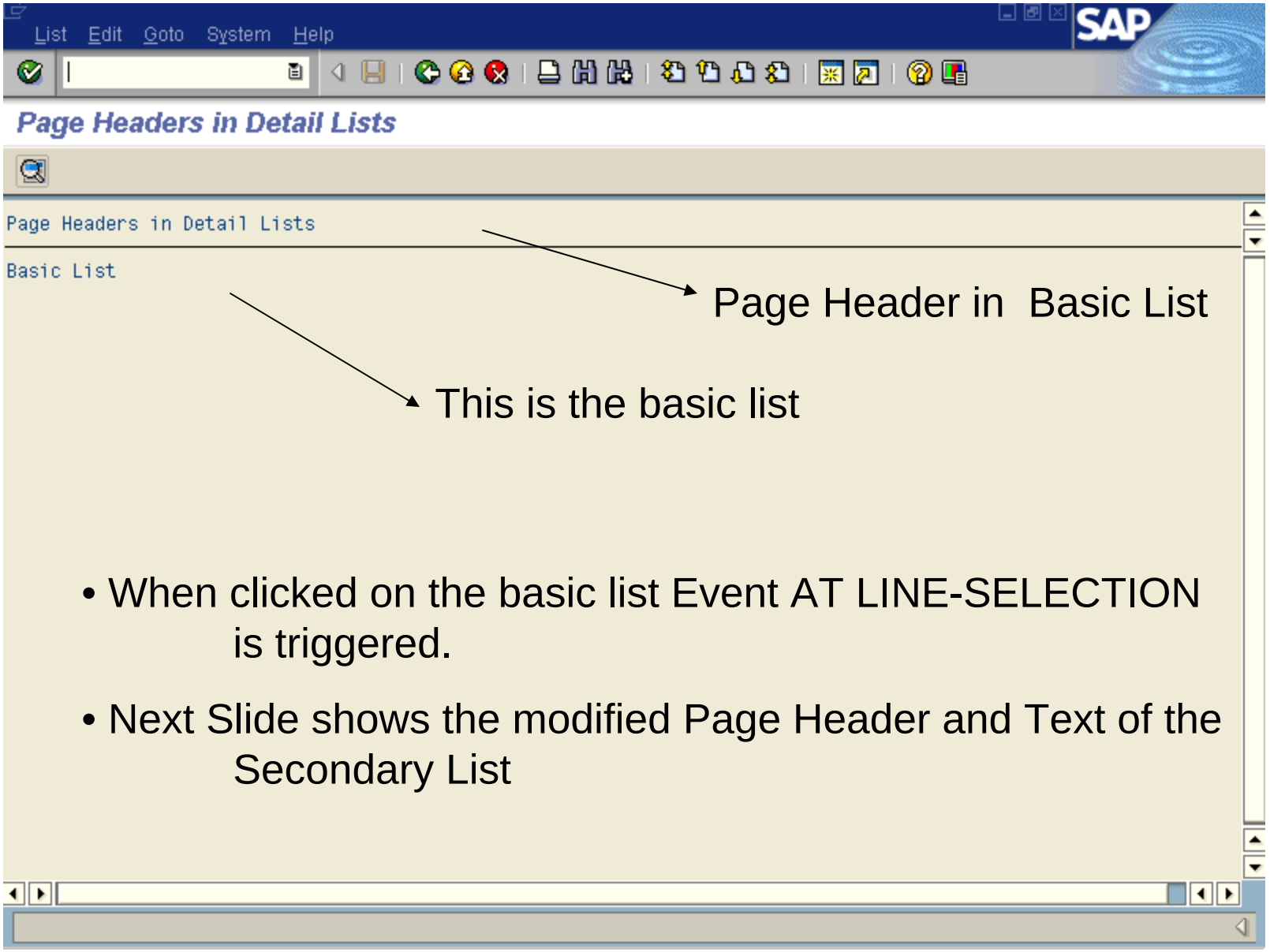
Syntax

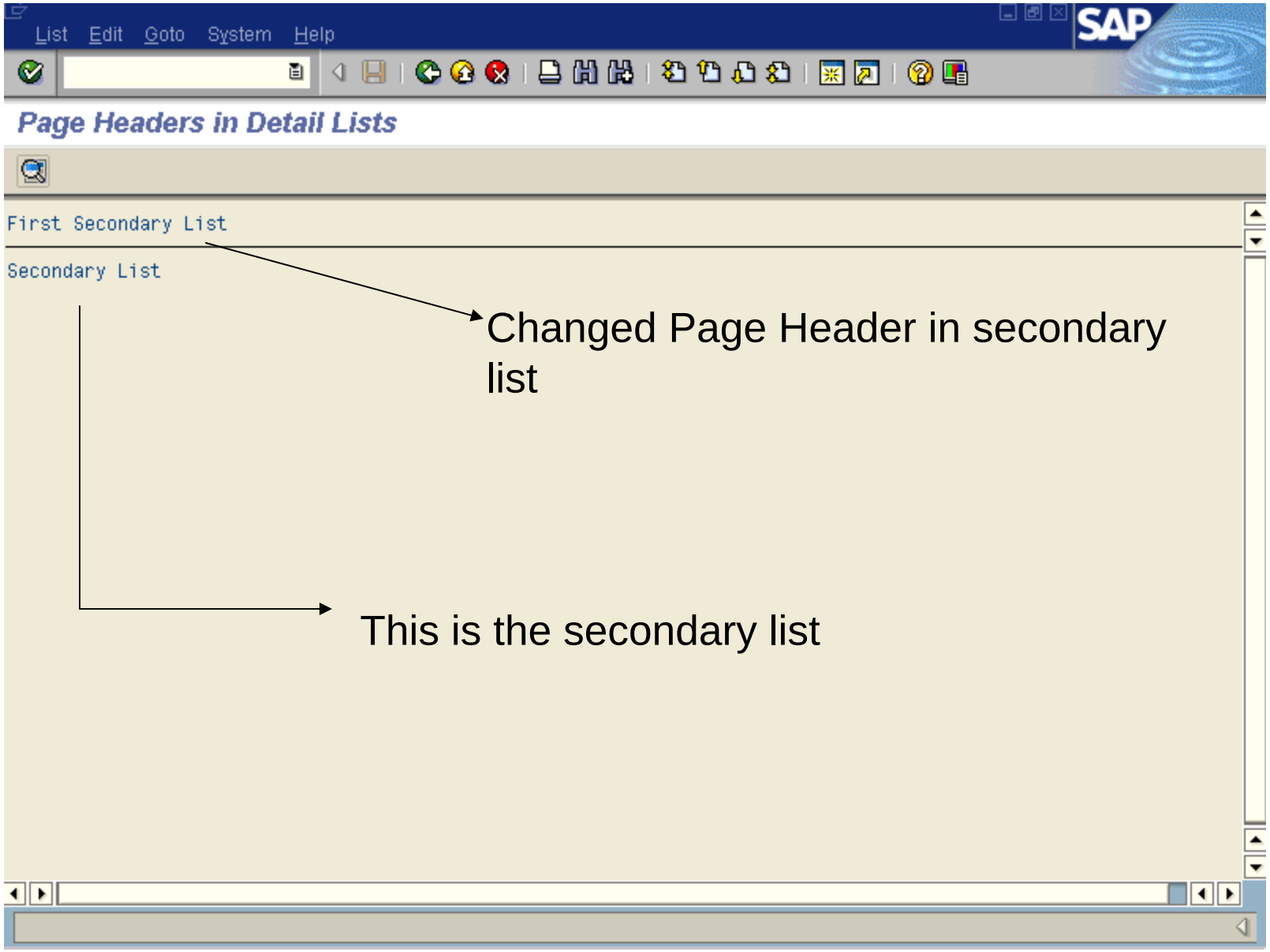
TOP-OF-PAGE DURING LINE-SELECTION.

The system triggers this event for each secondary list.

Program which shows At LINE-SELECTION & TOP-OF-PAGE DURING LINE-SELECTION.

```
REPORT demo_list_interactive_3 .

START-OF-SELECTION.

  WRITE  'Basic List'.

AT LINE-SELECTION.

  WRITE  'Secondary List'.

TOP-OF-PAGE DURING LINE-SELECTION.

  CASE sy-lsind.
    WHEN 1.
      WRITE 'First Secondary List'.
    WHEN 2.
      WRITE 'Second Secondary List'.
    WHEN OTHERS.
      WRITE: 'Secondary List, Level:', sy-lsind.
  ENDCASE.
  ULINE.
```


List Edit Goto System Help
SAP
Page Headers in Detail Lists
Page Headers in Detail Lists
Basic List
Page Header in Basic List
This is the basic list
• When clicked on the basic list Event AT LINE-SELECTION is triggered.
• Next Slide shows the modified Page Header and Text of the Secondary List


List Edit Goto System Help
SAP
Page Headers in Detail Lists
First Secondary List
Secondary List
Changed Page Header in secondary list
This is the secondary list

Allowing Function Key Selection

To allow the user to select an action by pressing a function key,

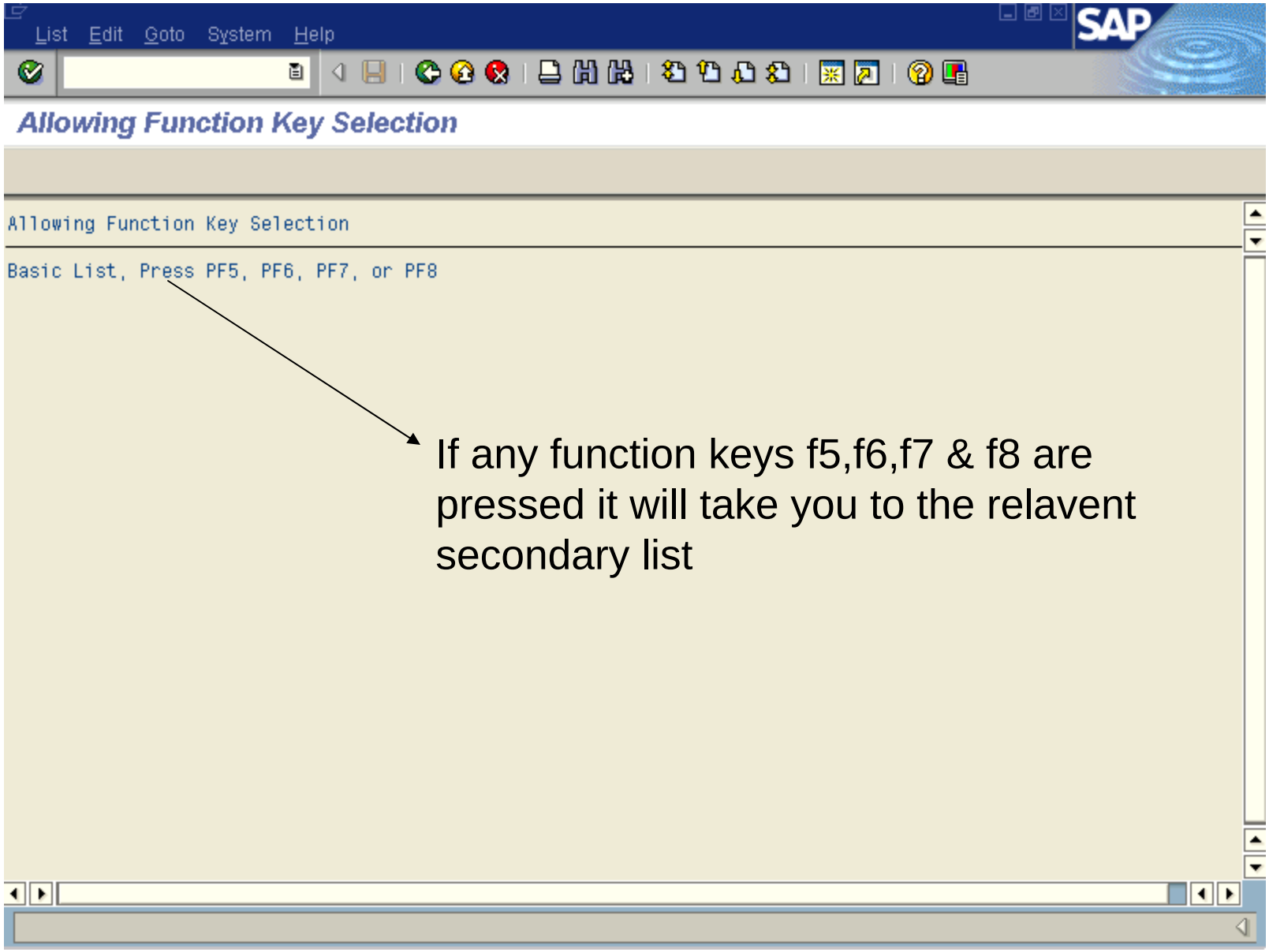
Syntax

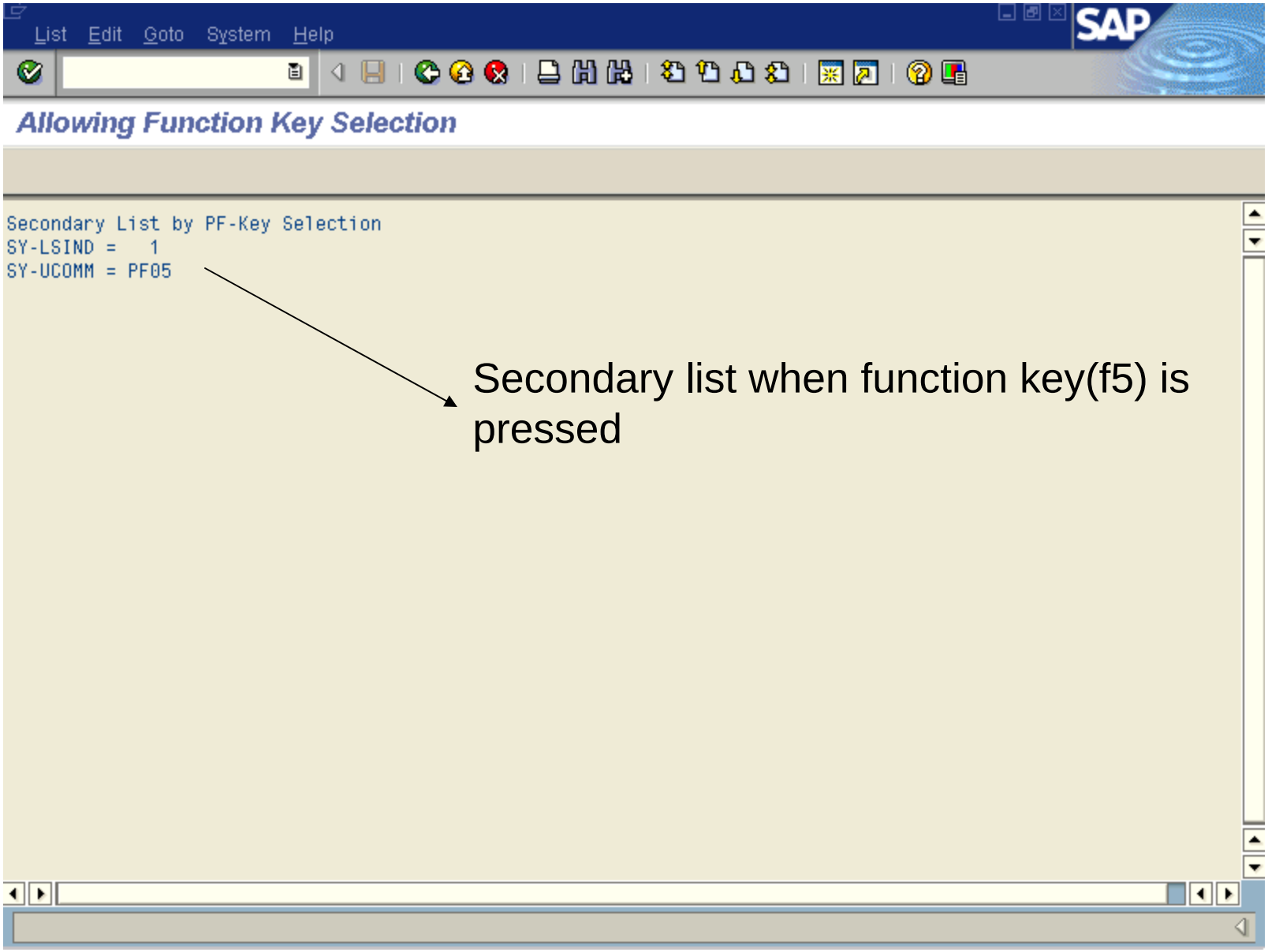
AT PF<nn>.
<statements>.

<nn> is a number between 1 and 24.

SY-UCOMM returns the function code PF<nn>.

```
START-OF-SELECTION.

  WRITE 'Basic List, Press PF5, PF6, PF7, or PF8'.

AT pf5.
  PERFORM out.

AT pf6.
  PERFORM out.

AT pf7.
  PERFORM out.

AT pf8.
  PERFORM out.

FORM out.
  WRITE: 'Secondary List by PF-Key Selection',
       / 'SY-LSIND =', sy-lsind,
       / 'SY-UCOMM =', sy-ucomm.
ENDFORM.
```


List Edit Goto System Help
SAP
Allowing Function Key Selection
Allowing Function Key Selection
Basic List, Press PF5, PF6, PF7, or PF8
If any function keys f5,f6,f7 & f8 are pressed it will take you to the relavent secondary list


List Edit Goto System Help
SAP
Allowing Function Key Selection
Secondary List by PF-Key Selection
SY-LSIND = 1
SY-UCOMM = PF05
Secondary list when function key(f5) is pressed

Setting a Status

You set the status with the SET PF-STATUS statement,

Syntax

SET PF-STATUS <stat> [EXCLUDING <f>|<itab>].

Note:

For the Status type of interactive lists, choose List or List in dialog box. The system then automatically loads function codes predefined for list processing into the Menu Painter.

The AT USER-COMMAND Event

To allow your program to react to a user action triggering a self-defined function code

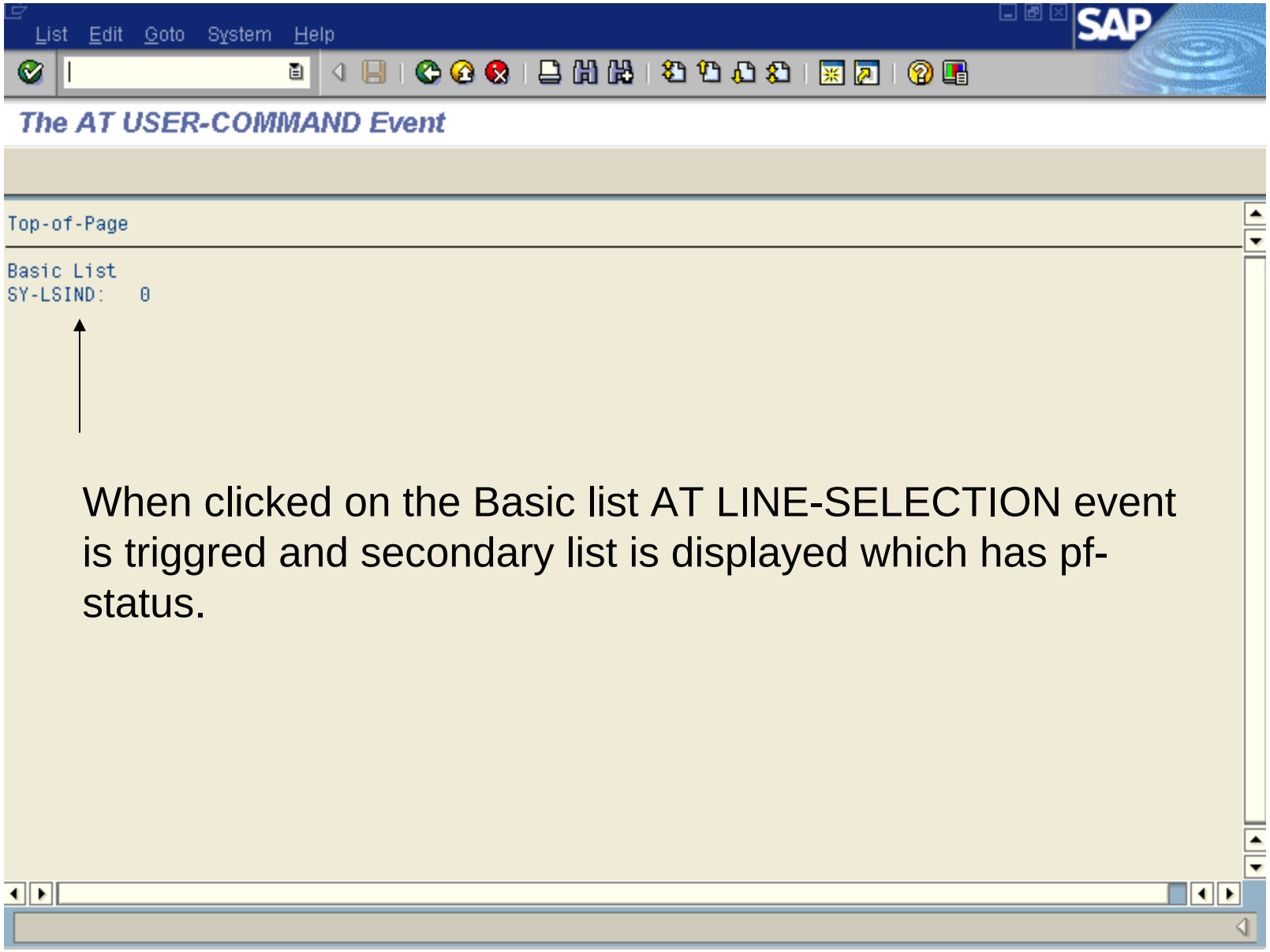
Syntax

AT USER-COMMAND.
<statements>.

The AT USER-COMMAND event occurs whenever the user selects a self-defined function code from a self-defined user interface.

The event does not occur if the user selects function codes predefined for system functions or the function code PICK, which always triggers the AT LINE-SELECTION event.

```
REPORT demo_list_at_user_command NO STANDARD PAGE HEADING.

START-OF-SELECTION.
  WRITE: 'Basic List',
       / 'SY-LSIND:', sy-lsind.

TOP-OF-PAGE.
  WRITE 'Top-of-Page'.
  ULINE.

TOP-OF-PAGE DURING LINE-SELECTION.
  CASE sy-pfkey.
    WHEN 'TEST'.
      WRITE 'Self-defined GUI for Function Codes'.
      ULINE.
  ENDCASE.

AT LINE-SELECTION.
  SET PF-STATUS 'TEST' EXCLUDING 'PICK'.
  PERFORM out.
  sy-lsind = sy-lsind - 1.
```


List Edit Goto System Help
SAP
The AT USER-COMMAND Event
Top-of-Page
Basic List
SY-LSIND: 0
When clicked on the Basic list AT LINE-SELECTION event is triggred and secondary list is displayed which has pf-status.


List Edit Goto System Help
SAP
The AT USER-COMMAND Event
FUN 1
FUN 2
FUN 3
FUN 4
FUN 5
Self-defined GUI for Function Codes
Secondary List
SY-LSIND: 1
SY-PFKEY: TEST
Button FUN 1 was pressed
At USER-COMMAND when any of the button in application tool bar are pressed it will take to relavent lists.

Important System Fields for Secondary Lists

SY-LSIND	**Index of the list created during the current event (basic list = 0)**
SY-LISTI	**Index of the list level from which the event was triggered**
SY-LILLI	**Absolute number of the line from which the event was triggered**
SY-LISEL	**Contents of the line from which the event was triggered**
SY-UCOMM	**Function code that triggered the event**
SY-PFKEY	**Status of the displayed list**

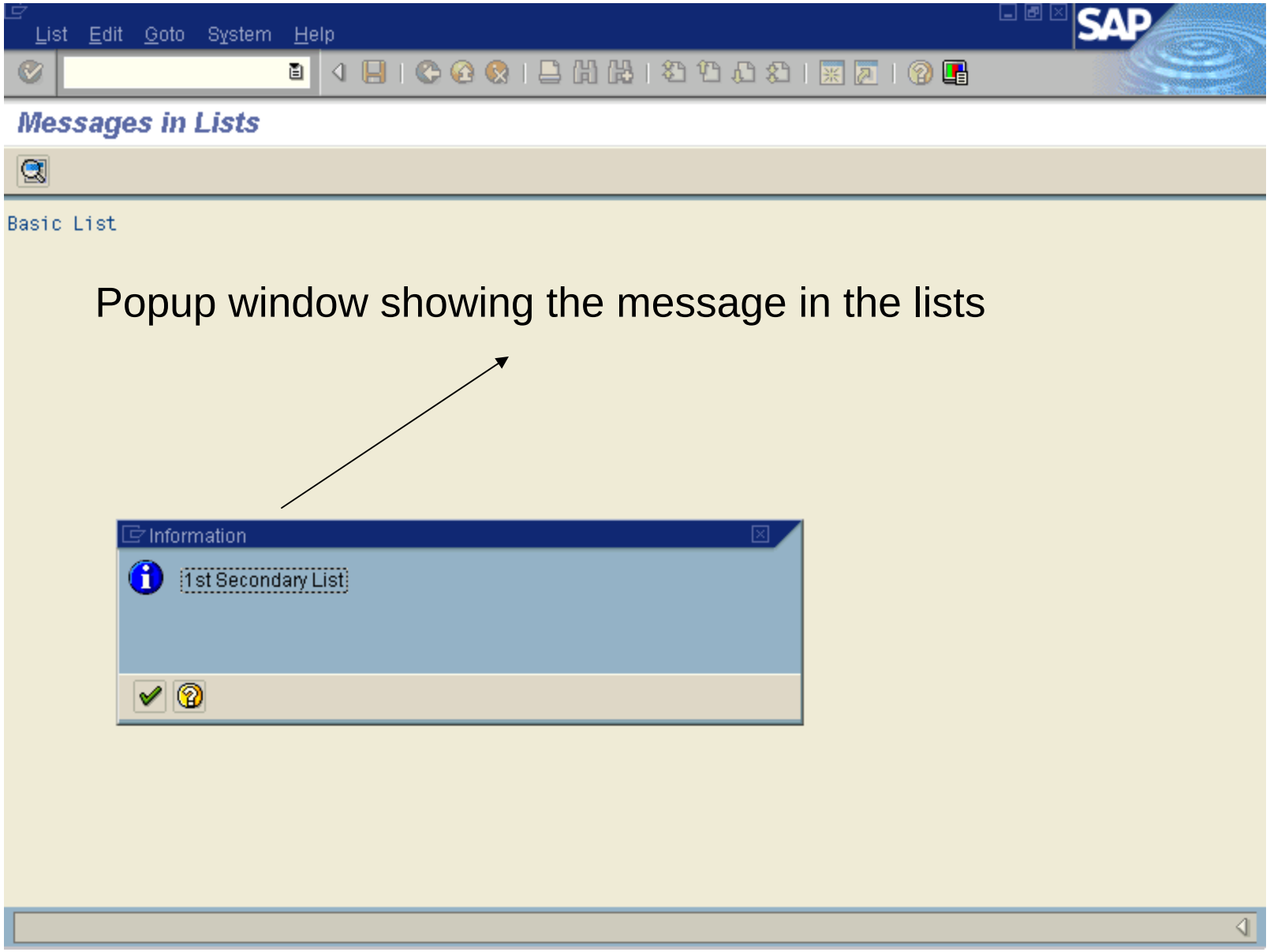
Messages in Lists:

They are grouped by language, a two-character ID, and a three-digit number. From your program, you can send a message with different qualifications:

A	**Abend; the current transaction is stopped**
E	**Error; the system waits for new input data**
I	**Information; after pressing ENTER, the system continues processing**
S	**Confirmation; the message appears on the next screen**
W	**Warning; you can change the input data or continue by pressing ENTER**

You must specify the MESSAGE-ID behind the REPORT statement of your program.

```
REPORT demo_list_interactive_4 NO STANDARD PAGE HEADING.

START-OF-SELECTION.

WRITE 'Basic List'.
MESSAGE s888(sabapdocu) WITH text-001.

AT LINE-SELECTION.
IF sy-lsind = 1.
MESSAGE i888(sabapdocu) WITH text-002.
ENDIF.
IF sy-lsind = 2.
MESSAGE e888(sabapdocu) WITH text-003 sy-lsind text-004.
ENDIF.
WRITE: 'Secondary List, SY-LSIND:', sy-lsind.
```


List Edit Goto System Help
SAP
Messages in Lists
Basic List
Popup window showing the message in the lists
Information
1st Secondary List

Defining Titles for Interactive Lists

By default, the system uses the program title as the title of the output screen of a report. To choose another title for the output screens, use this statement:

Syntax

SET TITLEBAR <tit> [WITH <g1> ... <g9>].

Displaying Lists in Dialog Windows

To display a secondary list in a dialog window use the WINDOW statement:

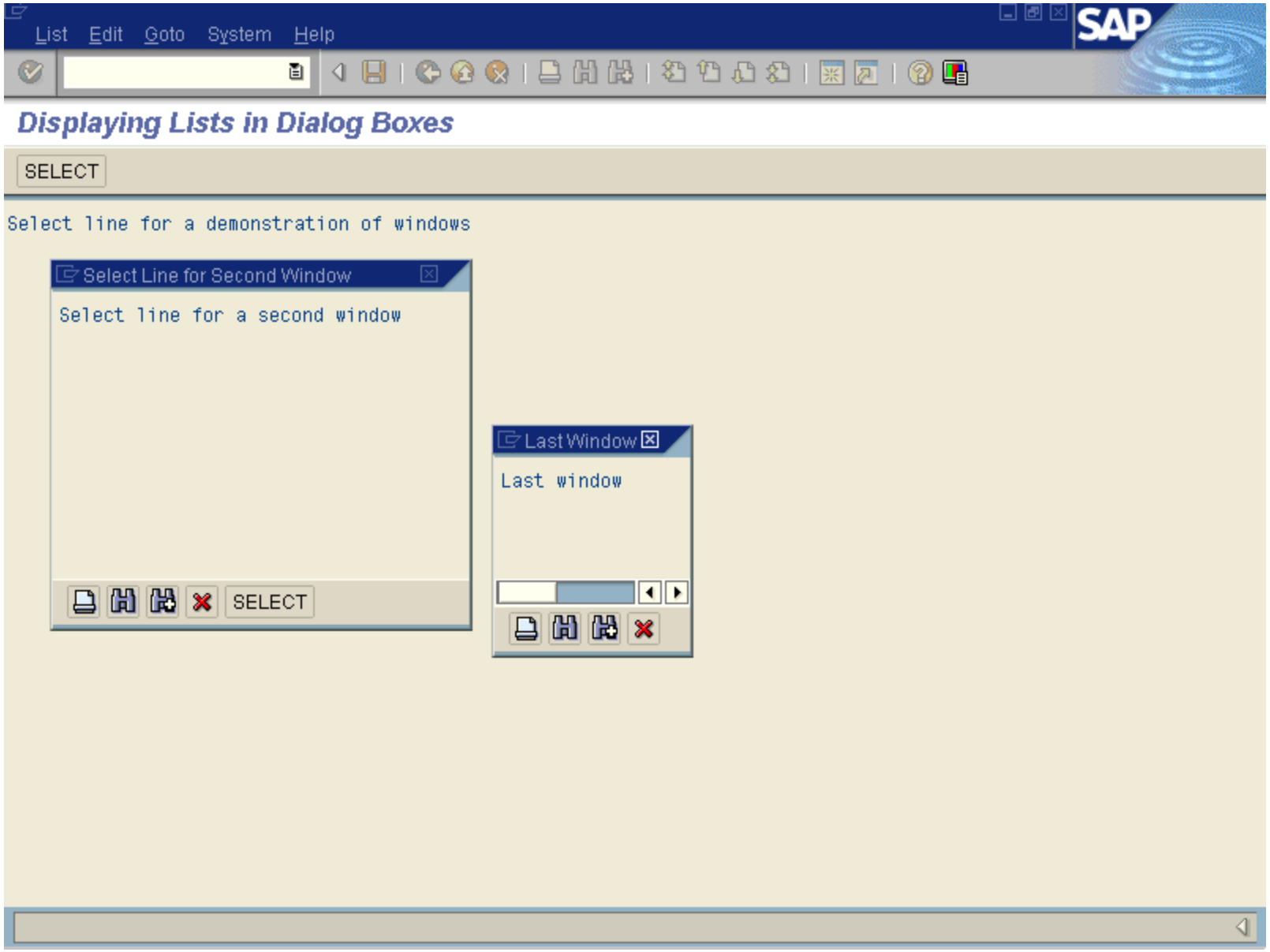
Syntax

WINDOW STARTING AT <left> <upper> [ENDING AT <right> <lower>].

The WINDOW statement takes effect only within the processing block of an interactive event, that is, only for secondary lists.

Note:
Dialog windows have no menu bar and no standard toolbar

```
REPORT demo_list_window NO STANDARD PAGE HEADING.

START-OF-SELECTION.
  SET PF-STATUS 'BASIC'.
  WRITE 'Select line for a demonstration of windows'.

AT USER-COMMAND.
  CASE sy-ucomm.
    WHEN 'SELE'.
      IF sy-lsind = 1.
        SET PF-STATUS 'DIALOG'.
        SET TITLEBAR 'WI1'.
        WINDOW STARTING AT 5 3 ENDING AT 40 10.
        WRITE 'Select line for a second window'.
      ELSEIF sy-lsind = 2.
        SET PF-STATUS 'DIALOG' EXCLUDING 'SELE'.
        SET TITLEBAR 'WI2'.
        WINDOW STARTING AT 45 10 ENDING AT 60 12.
        WRITE 'Last window'.
      ENDIF.
  ENDCASE.
```


List Edit Goto System Help
SAP
Displaying Lists in Dialog Boxes
SELECT
Select line for a demonstration of windows
Select Line for Second Window
Select line for a second window
SELECT
Last Window
Last window

Triggering Events from within the Program

Instead of letting the user trigger an interactive event by an action on the output screen, you can yourself trigger events from within the program.

Syntax

SET USER-COMMAND <fc>.

This statement takes effect after the current list is completed. Before the system displays the list, it triggers the event that corresponds to the function code stored in <fc>,independent of the applied user interface.

Note:

Function code PICK triggers an event only if the cursor is located on a list line .

Passing data from list to report

To effectively use interactive lists for interactive reporting, it is not sufficient for the program to react to events triggered by user actions on the output list. You must also be able to interpret the lines selected by the user and their contents.

Following options are available:

Passing data automatically using system fields

Using statements in the program to fetch data

Using SY-LISEL

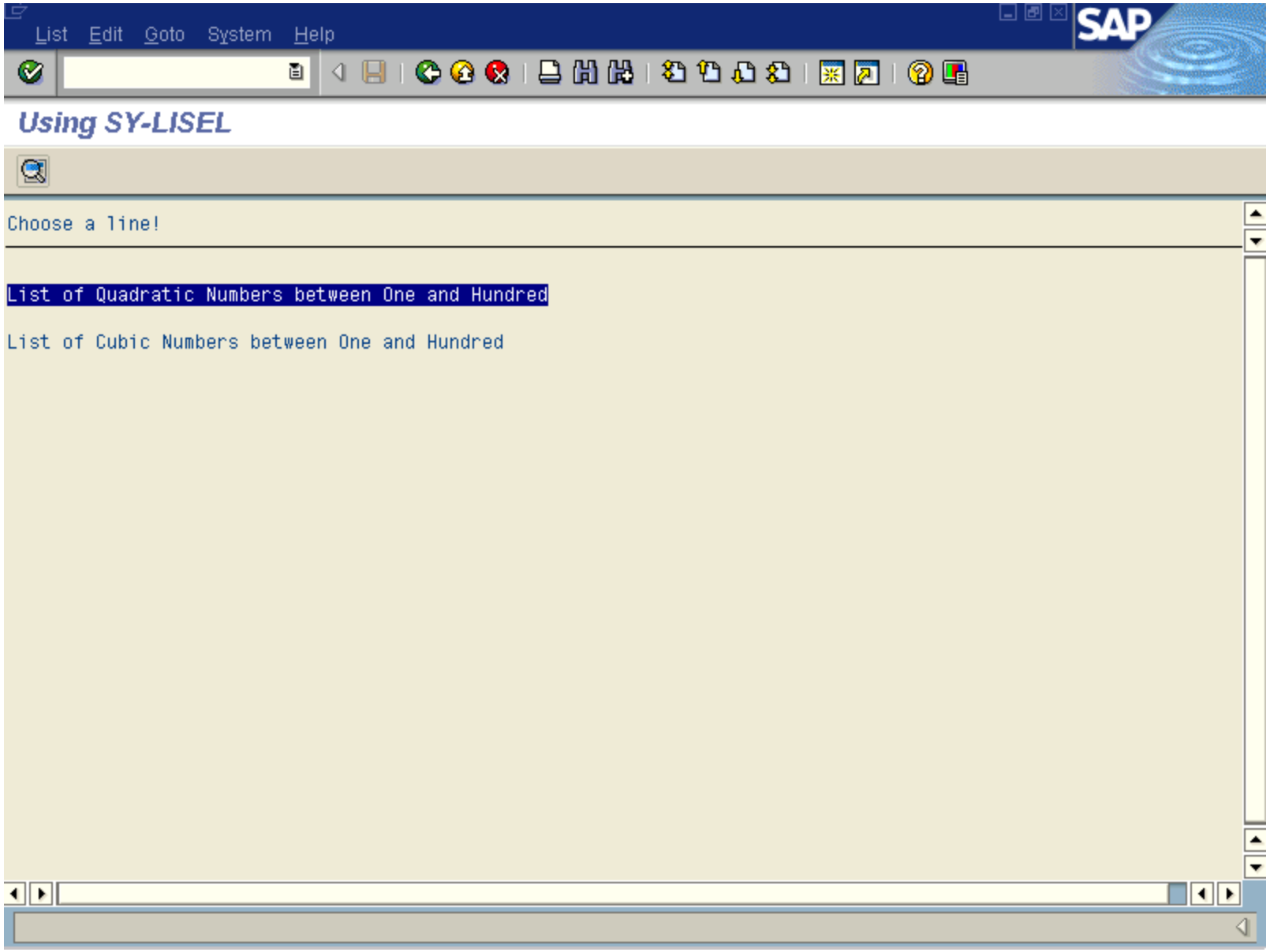
The SY-LISEL system field is a field of type C with a length of 255 characters.

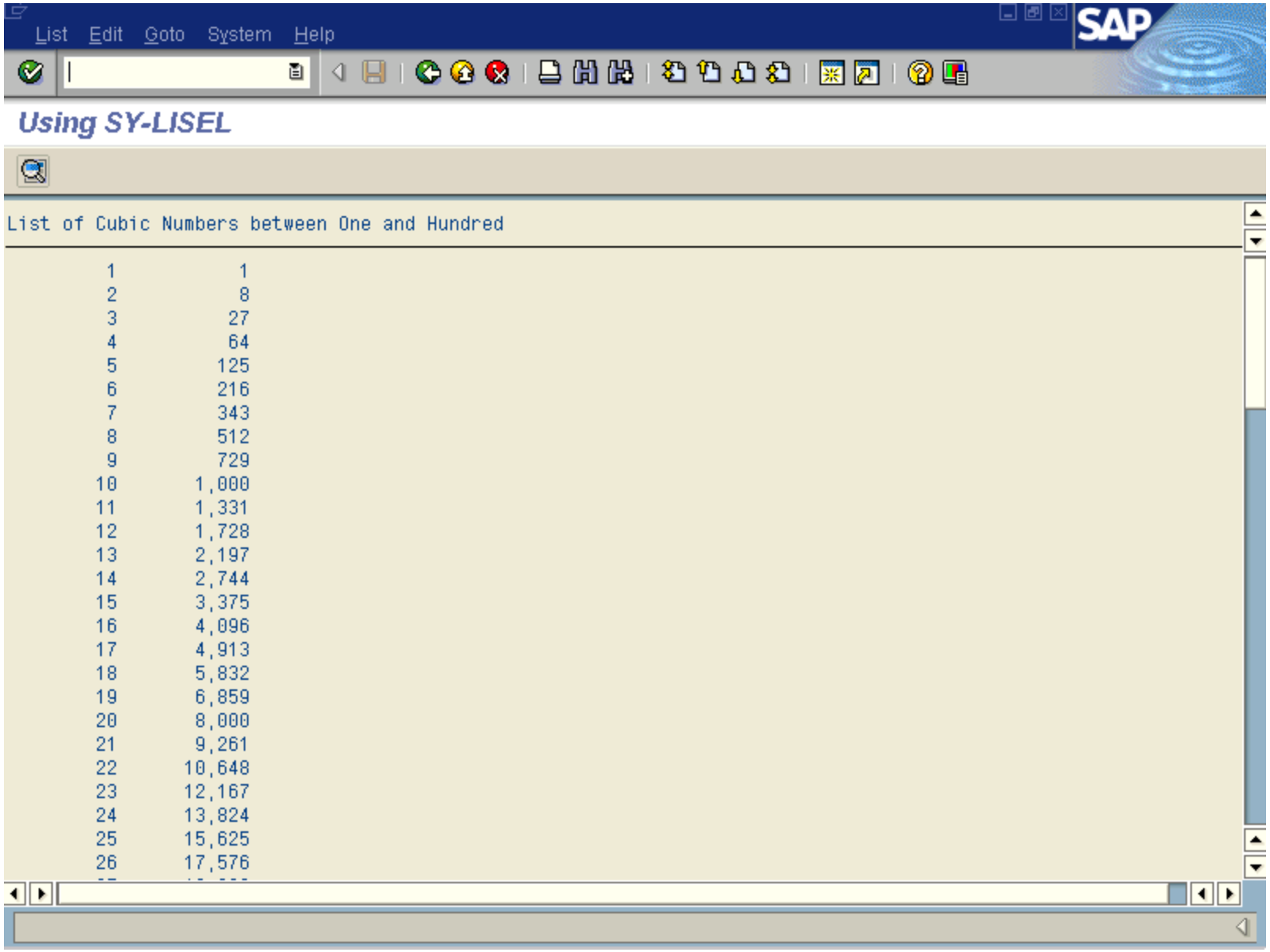
It contains the selected line as one single character string, thus making it difficult for you to retrieve the values of individual fields.

To process certain parts of SY-LISEL, you must specify the corresponding offsets

```
REPORT demo_list_sy_lisel NO STANDARD PAGE HEADING.

DATA num TYPE i.

SKIP.
WRITE 'List of Quadratic Numbers between One and Hundred'.
SKIP.
WRITE 'List of Cubic Numbers between One and Hundred'.

TOP-OF-PAGE.

 WRITE 'Choose a line!'.
 ULINE.

TOP-OF-PAGE DURING LINE-SELECTION.

  WRITE sy-lisel.
  ULINE.
```

```
AT LINE-SELECTION.

  IF sy-lisel(4) = 'List'.
    CASE sy-lilli.
      WHEN 4.
        DO 100 TIMES.
          num = sy-index ** 2.
          WRITE: / sy-index, num.
        ENDDO.
      WHEN 6.
        DO 100 TIMES.
          num = sy-index ** 3.
          WRITE: / sy-index, num.
        ENDDO.
    ENDCASE.
  ENDIF.
```


List Edit Goto System Help
SAP
Using SY-LISEL
Choose a line!
List of Quadratic Numbers between One and Hundred
List of Cubic Numbers between One and Hundred


List Edit Goto System Help
SAP
Using SY-LISEL
List of Cubic Numbers between One and Hundred
1 1
2 8
3 27
4 64
5 125
6 216
7 343
8 512
9 729
10 1,000
11 1,331
12 1,728
13 2,197
14 2,744
15 3,375
16 4,096
17 4,913
18 5,832
19 6,859
20 8,000
21 9,261
22 10,648
23 12,167
24 13,824
25 15,625
26 17,576

Passing Data by Program Statements

HIDE

The moment you create a list level you can define which information to pass to the subsequent secondary lists.

READ LINE

Use the statements READ LINE and READ CURRENT LINE to explicitly read data from the lines of existing list levels.

GET CURSOR

Use the statements GET CURSOR FIELD and GET CURSOR LINE to pass the output field or output line on which the cursor was positioned during the interactive event to the processing block.

The HIDE Technique

You use the HIDE technique while creating a list level to store line-specific information for later use.

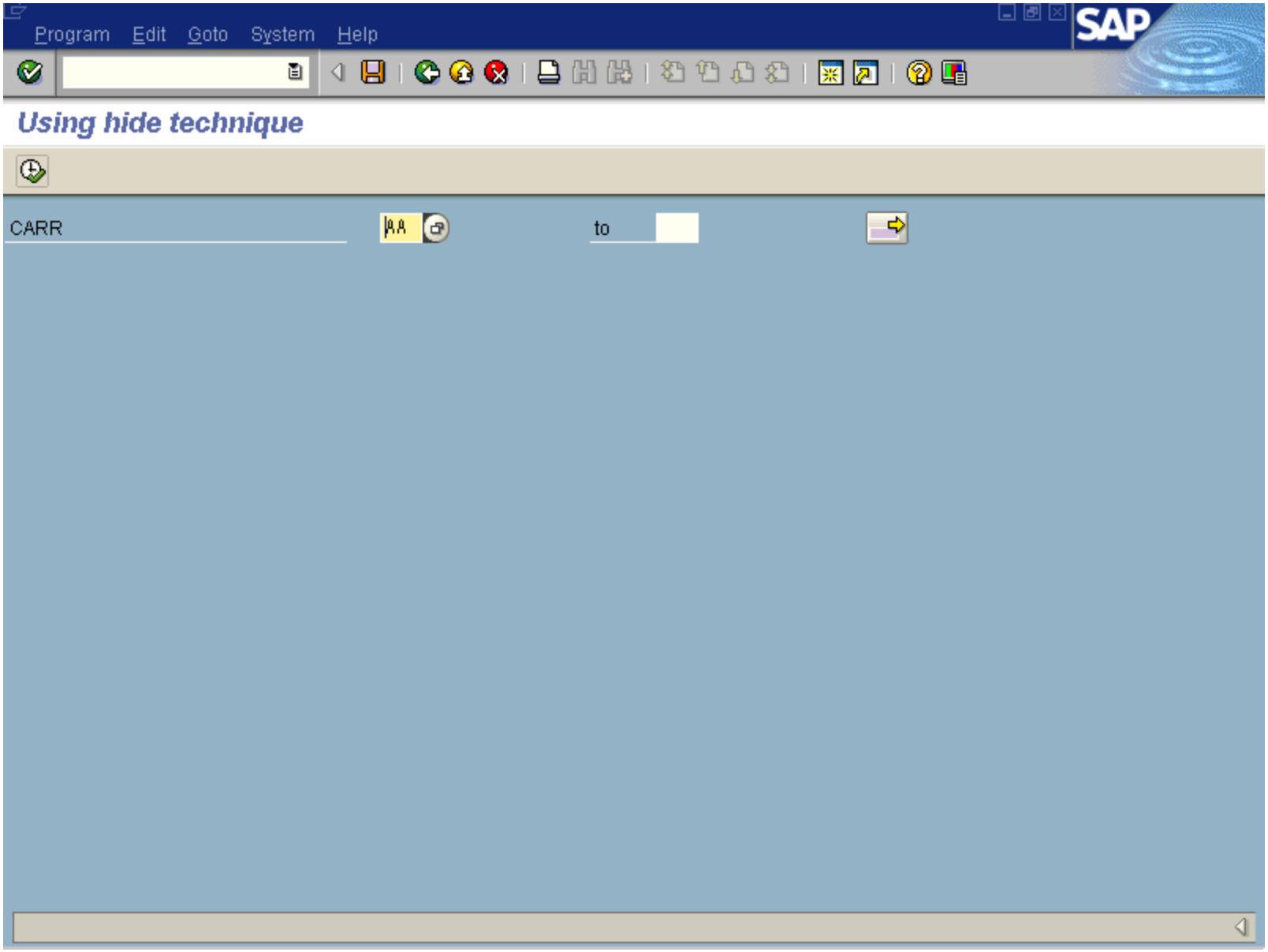
Syntax

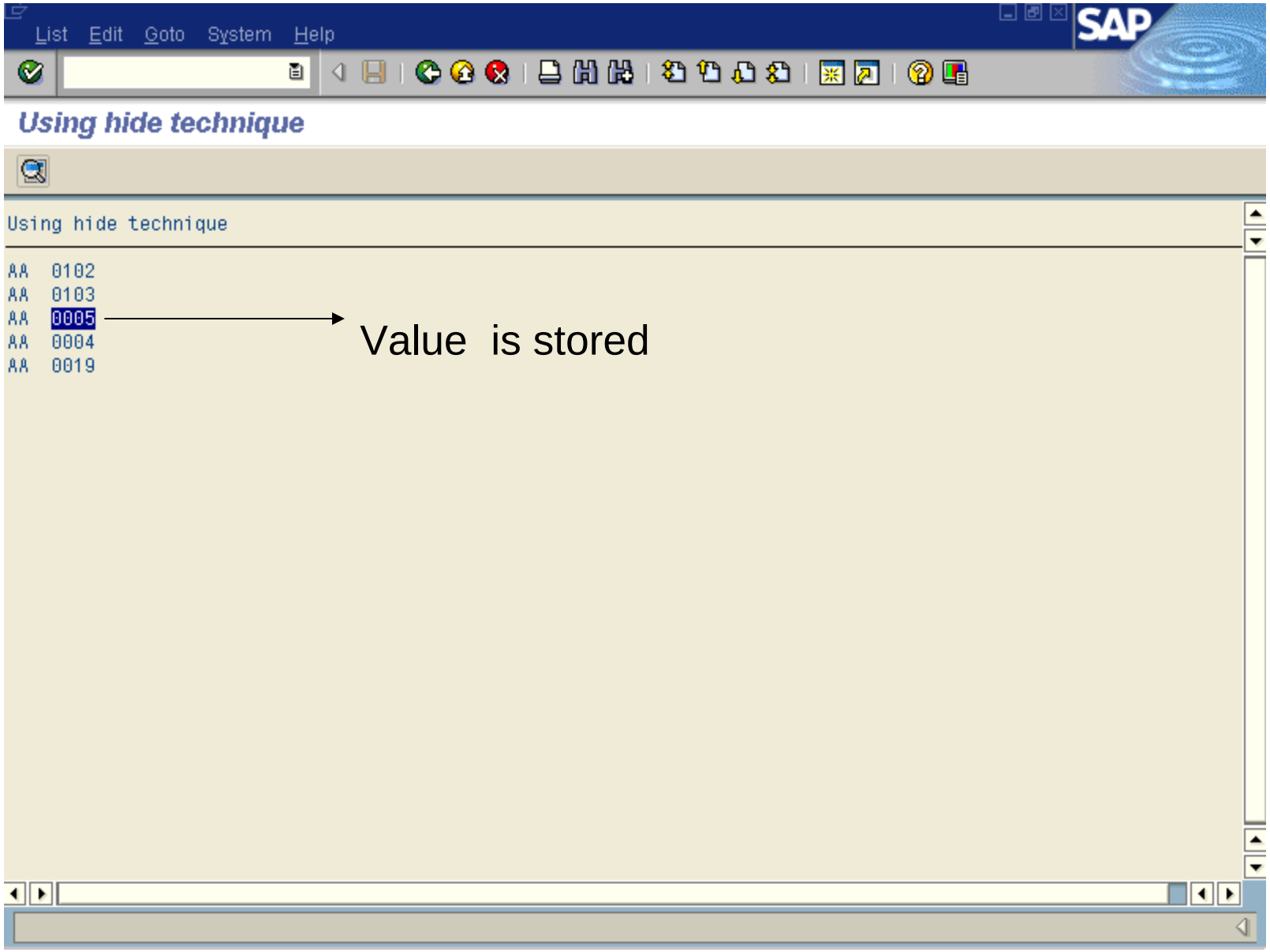
HIDE <f>.

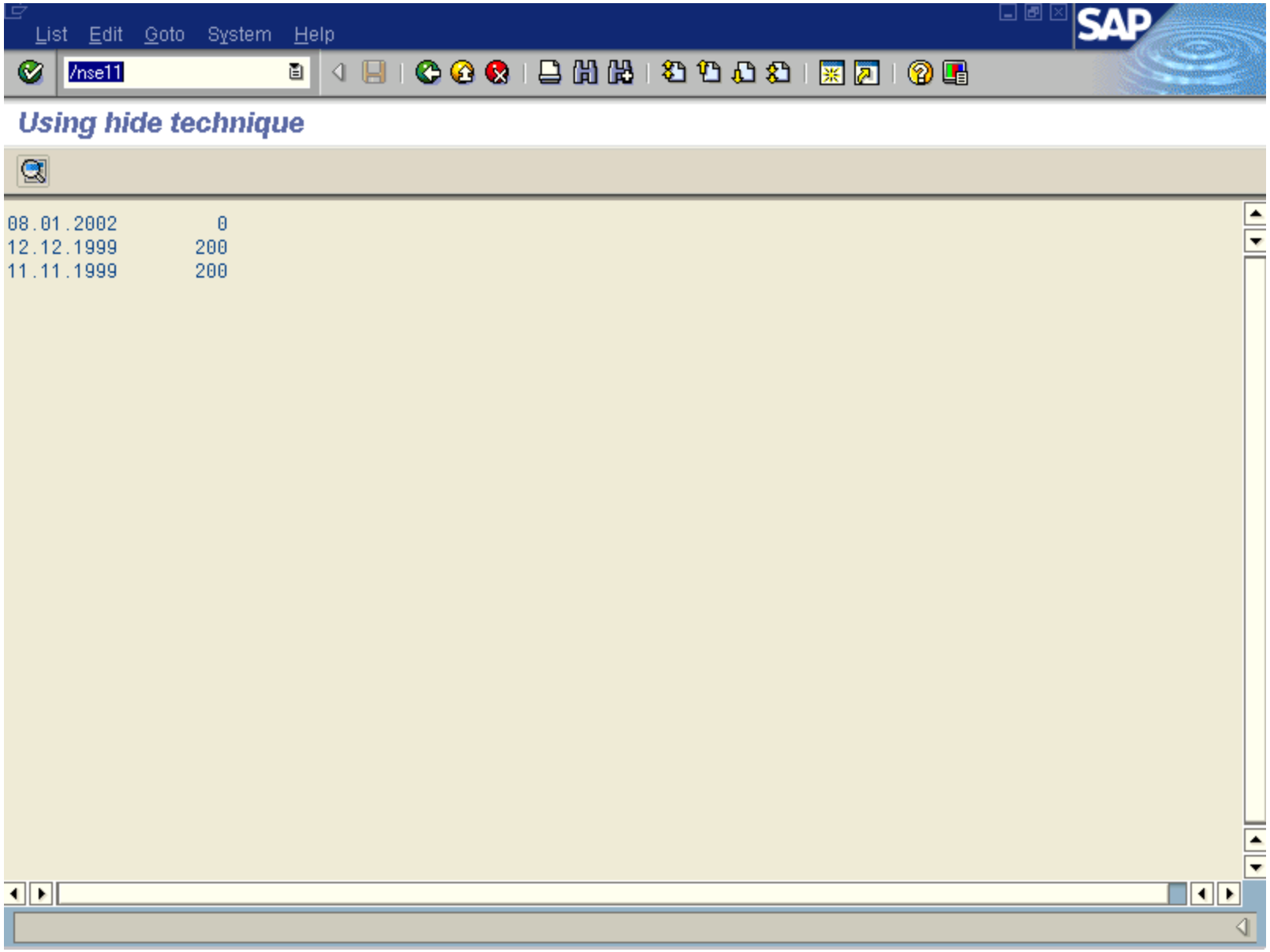
This statement stores the contents of variable <f> in relation to the current output line (system field SY-LINNO) internally in the so-called HIDE area. The variable <f> need not necessarily appear on the current line.

As soon as the user selects a line for which you stored HIDE fields, the system fills the variables in the program with the values stored.

```
REPORT Z_HIDE1 .
TABLES : SPFLI, SFLIGHT.
SELECT-OPTIONS : CARR FOR SPFLI-CARRID.
SELECT * FROM SPFLI WHERE CARRID IN CARR.
WRITE :/ SPFLI-CARRID , SPFLI-CONNID.
HIDE SPFLI-CARRID.
ENDSELECT.
AT LINE-SELECTION.
CASE SY-LSIND.
WHEN '1'.
SELECT * FROM SFLIGHT WHERE CARRID = SPFLI-CARRID.
WRITE :/ SFLIGHT-FLDATE, SFLIGHT-SEATSOCC.
ENDSELECT.
ENDCASE.
```


Program Edit Goto System Help
SAP
Using hide technique
CARR
AA
to


List Edit Goto System Help
SAP
Using hide technique
Using hide technique
AA 0102
AA 0103
AA 0005
AA 0004
AA 0019
Value is stored


List Edit Goto System Help
SAP
/nse11
Using hide technique
08.01.2002 0
12.12.1999 200
11.11.1999 200

Reading Lines from Lists

To read a line from a list after an interactive list event occurred, use the READ LINE statement:

Syntax

READ LINE <lin> [INDEX <idx>]
[FIELD VALUE <f1> [INTO <g1>] ... <fn> [INTO <gn>]].

The statement without any options stores the contents of line <lin> from the list on which the event was triggered (index SY-LILLI) in the SY-LISEL system field and fills all HIDE information stored for this line back into the corresponding fields .

Reading Lists at the Cursor Position

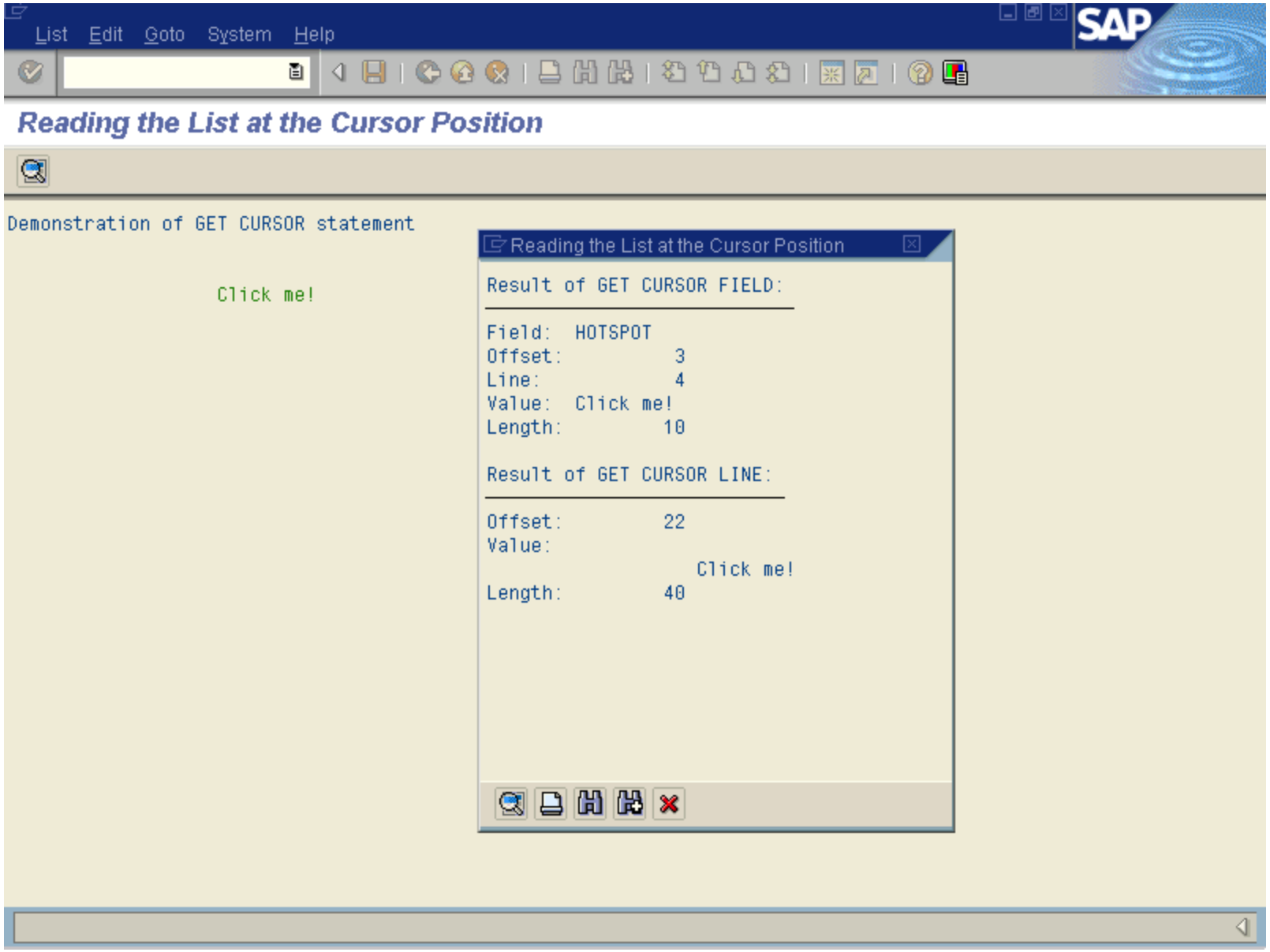
To retrieve information on the cursor position during an interactive event, use the GET CURSOR statement to refer to either the field or the line.

Syntax

GET CURSOR FIELD <f> [VALUE <val>] .

```
REPORT demo_list_get_cursor NO STANDARD PAGE HEADING LINE-SIZE 40.

DATA: hotspot(10) TYPE c VALUE 'Click me!',

      f(10) TYPE c, off TYPE i, lin TYPE i, val(40) TYPE c, len TYPE i.

FIELD-SYMBOLS <fs> TYPE ANY.

ASSIGN hotspot TO <fs>.

WRITE 'Demonstration of GET CURSOR statement'.

SKIP TO LINE 4.

POSITION 20.

WRITE <fs> HOTSPOT COLOR 5 INVERSE ON.
```

```
AT LINE-SELECTION.

  WINDOW STARTING AT 5 6 ENDING AT 45 20.
  GET CURSOR FIELD f OFFSET off
           LINE lin VALUE val LENGTH len.
  WRITE: 'Result of GET CURSOR FIELD: '.
  ULINE AT /(28).
  WRITE: / 'Field: ', f,
         / 'Offset:', off,
         / 'Line:  ', lin,
         / 'Value: ', (10) val,
         / 'Length:', len.
  SKIP.
  GET CURSOR LINE lin OFFSET off VALUE val LENGTH len.
  WRITE: 'Result of GET CURSOR LINE: '.
  ULINE AT /(27).
  WRITE: / 'Offset:', off,
         / 'Value: ', val,
         / 'Length:', len.
```


List Edit Goto System Help
SAP
Reading the List at the Cursor Position
Demonstration of GET CURSOR statement
Click me!
Reading the List at the Cursor Position
Result of GET CURSOR FIELD:
Field: HOTSPOT
Offset: 3
Line: 4
Value: Click me!
Length: 10
Result of GET CURSOR LINE:
Offset: 22
Value: Click me!
Length: 40

Calling Reports

To call a report from with in other report,use the SUBMIT statement.

To set the name of the called program statically in the programcoding.

Syntax :

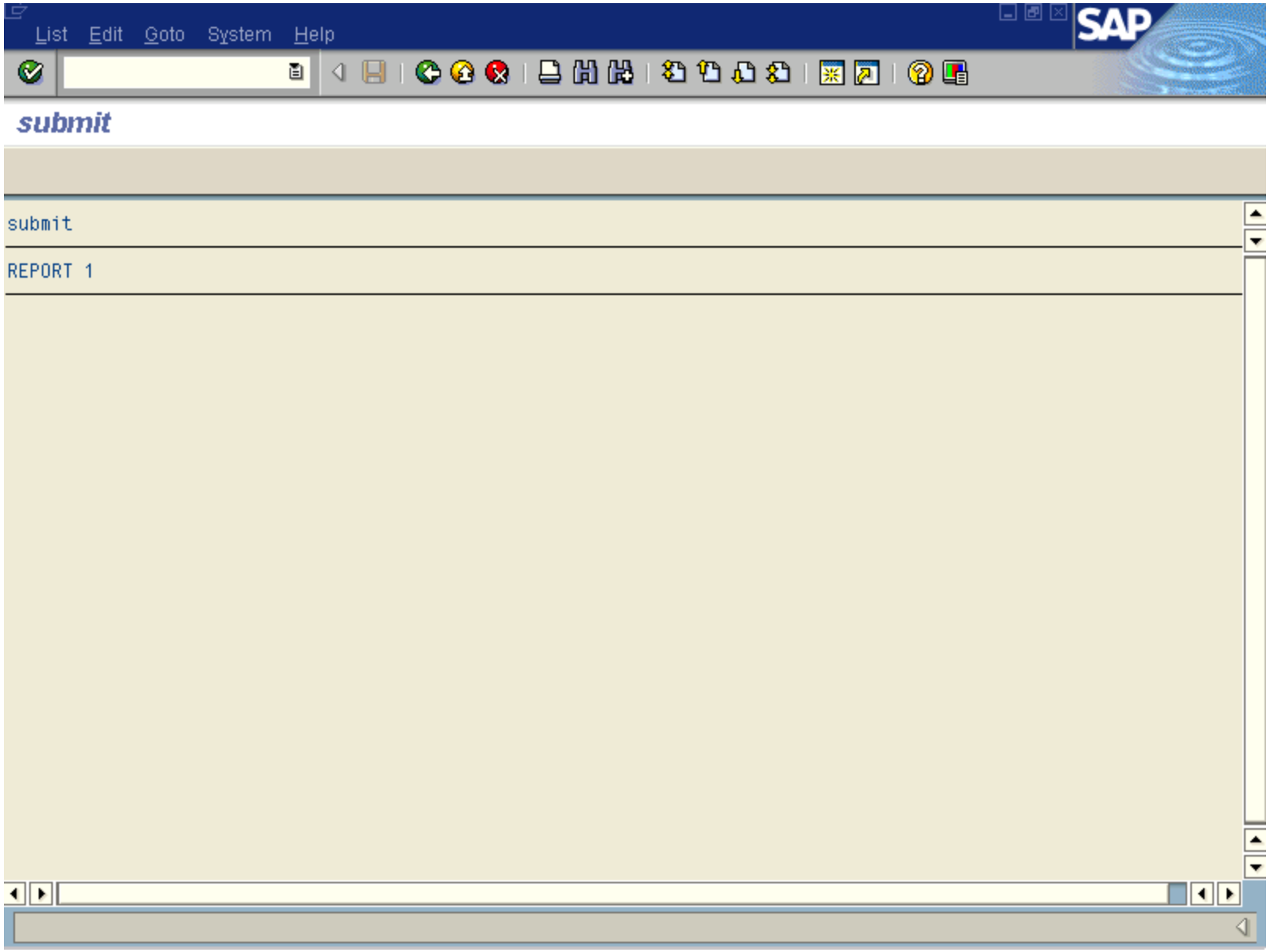
SUBMIT <rep> [AND RETURN] [<options>].

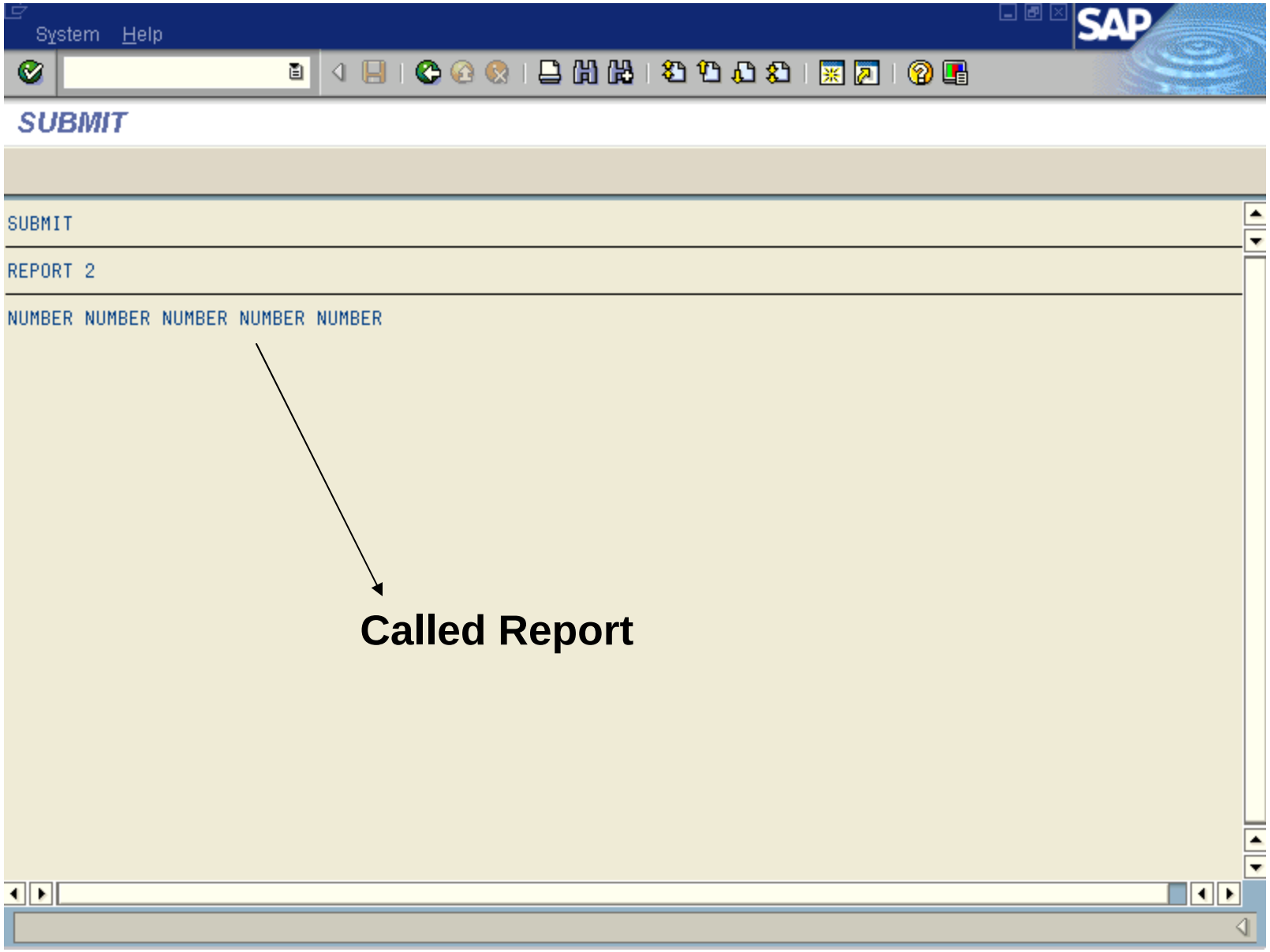
The first statement starts the report<rep>,the second statement starts the report whose name is stored in field<rep>.

In this program we r calling a program zsapmztst1 next slide shows the zsapmztst1.

```
REPORT ZSAPMZTST .

DATA : ITAB TYPE I OCCURS 10,
       NUM TYPE I.

 SUBMIT ZSAPMZTST1 AND RETURN.
 IMPORT ITAB FROM MEMORY ID 'HK'.
 LOOP AT ITAB INTO NUM.
 WRITE / NUM.
 ENDLOOP.
 WRITE 'REPORT 1'.
 ULINE.
```

```
REPORT ZSAPMZTST1 .
DATA : NUMBER TYPE I,
ITAB TYPE I OCCURS 10.
SET PF-STATUS 'MYBACK'.
DO 5 TIMES.
NUMBER = SY-INDEX.
APPEND NUMBER TO ITAB.
WRITE : 'NUMBER'.
ENDDO.
TOP-OF-PAGE.
WRITE 'REPORT 2'.
ULINE.
AT USER-COMMAND.
CASE SY-UCOMM.
WHEN 'MYBACK'.
EXPORT ITAB TO MEMORY ID 'HK'.
LEAVE.
ENDCASE.
```


List Edit Goto System Help
SAP
submit
submit
REPORT 1


System Help
SAP
SUBMIT
SUBMIT
REPORT 2
NUMBER NUMBER NUMBER NUMBER NUMBER
Called Report

Exercise 1

Create an interactive list using hide technique based on tables ekko and ekpo. Output the following field in the basic list.

EKKO-EBELN.

Based on the output of the basic list I.e., EKKO-EBELN, output the following fields in the secondary list by passing the field EKKO-MATNR by HIDE Technique.

EKPO-BUKRS

EKPO-WERKS

EKPO-LGORT

EKPO-MENGE

Solution

```
TABLES: EKKO, EKPO.

DATA: BEGIN OF ITAB1 OCCURS 0,
EBELN LIKE EKKO-EBELN,
END OF ITAB1.
DATA : BEGIN OF ITAB2 OCCURS 0,
MATNR LIKE EKPO-MATNR,
BUKRS LIKE EKPO-BUKRS,
WERKS LIKE EKPO-WERKS,
LGORT LIKE EKPO-LGORT,
MENGE LIKE EKPO-MENGE,
END OF ITAB2.
SELECT EBELN FROM EKKO INTO ITAB1-EBELN.
  WRITE :/  ITAB1-EBELN HOTSPOT.
  HIDE ITAB1-EBELN.
ENDSELECT.
```

```
TOP-OF-PAGE DURING LINE-SELECTION.
  WRITE:/ 'MATERIAL' ,12 'COMPANY',18 'PLANT',28
'STORAGE',38'QUANTITY'.

AT LINE-SELECTION.
* Interactive event, details list
* Read data corresponding to the selection criteria and
* display
  SELECT MATNR BUKRS WERKS LGORT MENGE FROM EKPO  INTO
CORRESPONDING
                    FIELDS OF ITAB2 WHERE EBELN = ITAB1-EBELN.
    WRITE: /     ITAB2-MATNR,   * Material Number
            12 ITAB2-BUKRS,       * Company code
            18 ITAB2-WERKS,       * Plant
            28 ITAB2-LGORT,       * Storage Location
            38 ITAB2-MENGE.       *Quantity
  ENDSELECT.
```

Exercise NO: 2

Task:

Create a list of all flight connections for each airline carrier.

Output the following fields:

SPFLI-CARRID, SPFLI-CONNID, SPFLI-CITYFROM,

SPFLI-CITYTO, SPFLI-DEPTIME, SPFLI-ARRTIME.

Define a selection screen on which the user can enter selections for the airline carrier (SPFLI-CARRID) and the connection ID (SPFLI-CONNID). Read the data from table SPFLI in accordance with this selection.

When the user selects a line, the system should display a secondary showing the flights (table SFLIGHT) available for the flight connection concerned (see example list). Output the selected line as the header of the secondary list. Calculate and output the number of vacant seats for each flight. Make sure that the program can handle invalid line selection.

Example list

AA 0017 NEW YORK SAN FRANCISCO 13:30:00 16:31:00

Date	Price	Seats		
		max.	occupied	free
10.29.1995	666.67 USD	660	10	650
11.11.1995	666.67 USD	660	20	640
11.16.1995	666.67 USD	660	38	622

Solution

```
REPORT ZABCD00111 .
TABLES: SPFLI, SFLIGHT.
* Report defined selection criteria for airline and for
* connection id
SELECT-OPTIONS:     SELCARR FOR SPFLI-CARRID,
                                SELCONN FOR SPFLI-CONNID.
* Auxiliary field: number of free seats
DATA:                   SFREE LIKE SFLIGHT-SEATSMAX.

START-OF-SELECTION.
* Read data corresponding to the selection criteria and
* display
SELECT * FROM SPFLI
        WHERE CARRID IN SELCARR AND CONNID IN SELCONN.
        WRITE: /        SPFLI-CARRID,
                        SPFLI-CONNID,
                        SPFLI-CITYFROM,
                        SPFLI-CITYTO,
```

```
SPFLI-DEPTIME,
                                SPFLI-ARRTIME.
          HIDE: SPFLI-CARRID, SPFLI-CONNID.
ENDSELECT.
 END-OF-SELECTION.
          CLEAR SPFLI-CARRID.
          "initialization
AT LINE-SELECTION.
* Interactive event, details list
          CHECK NOT SPFLI-CARRID IS INITIAL.
* Display detail list only if valid line-selection
  WRITE SY-LISEL.
  WRITE:          /5 TEXT-001, 29 TEXT-002, 49 TEXT-003,
          /  TEXT-004 UNDER TEXT-003, 60 TEXT-005, 70 TEXT-006.
* D: TEXT-001: Date
* D: TEXT-002: Price
* D: TEXT-003: Seats
* D: TEXT-004: Max
* D: TEXT-005: occupied         TEXT-006: free
```

```
ULINE.
        SELECT * FROM SFLIGHT WHERE CARRID EQ SPFLI-CARRID
                                    AND CONNID EQ SPFLI-CONNID.
* Read data for details list using the hided key fields,
* list output
  WRITE: /5      SFLIGHT-FLDATE,
        SFLIGHT-PRICE,
        SFLIGHT-CURRENCY.
        SFREE = SFLIGHT-SEATSMAX - SFLIGHT-SEATSOCC.
        WRITE:          SFLIGHT-SEATSMAX,
                        SFLIGHT-SEATSOCC,
                        SFREE.
        ENDSELECT.
        CLEAR SPFLI-CARRID. "initialization
```

Summary

Extract datasets :

An extract is a sequential dataset you can create with a report.Use extracts if you want to process large amount of data as a whole several times

- **Declaring Extract Records as Field Groups**
- **Filling an Extract with Data**
- **Reading an Extract**
- **Sorting an Extract**

Lists

Lists are the output medium for data from ABAP/4 report programs. Each program can produce up to 21 lists: one basic list and 20 secondary lists.

Structure and the options of a list are

- **The Standard List**
- **The Self-Defined List**
- **Lists with Several Pages**

Interactive List

A list is an interactive list if the user interface allows actions that trigger events and if the corresponding interactive event keywords occur in the report.

Events for Interactive Lists

- **AT LINE-SELECTION**
- **AT PF<nn>**
- **AT USER-COMMAND**