第八章 ALV 控件的使用

ALV（SAP List Viewer）控件是 SAP 业务中最常用的控件之一，本章先用一个简单的例子介绍用 ALV 控件显示数据，再以实例方式介绍 ALV 的强大功能，示例程序可以直接使用。

本章主要内容有：

（）简单的 ALV 控件实例；
（）自定义输出字段的 ALV 控件实例；
（）在屏幕上建立 ALV 控件；
（）自定义 ALV 控件的工具条按钮；
（）处理 ALV 控件双击事件；
（）通过 ALV 控件编辑内表和数据库更新；
（）ALV Tree 的使用。

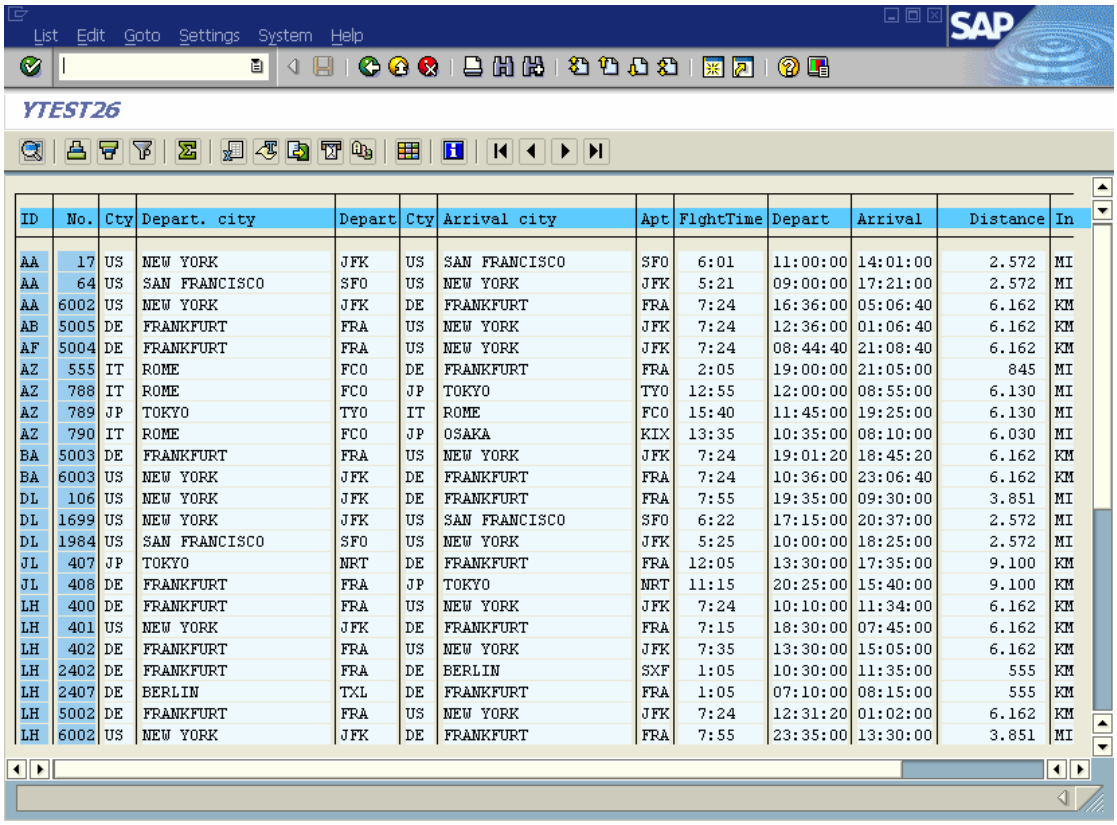
8.1 简单的 ALV 控件实例

以航班表（SPFLI）为例，使用数据字典定义结构，通过 ALV 控件显示数据。

【例 8.1】

```
REPORT  YTEST26.
*定义内表
DATA WA_SPFLI LIKE TABLE OF SPFLI WITH HEADER LINE.
*内表赋值
SELECT * INTO TABLE WA_SPFLI FROM SPFLI.
*通过数据字典结构显示ALV
CALL FUNCTION 'REUSE_ALV_LIST_DISPLAY'
  EXPORTING
    I_STRUCTURE_NAME = 'SPFLI'
  TABLES
    T_OUTTAB         = WA_SPFLI
  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.
```

输出结果如图 8-1 所示。



ID	No.	Cty	Depart. city	Depart	Cty	Arrival city	Apt	FlghtTime	Depart	Arrival	Distance	In
AA	17	US	NEW YORK	JFK	US	SAN FRANCISCO	SFO	6:01	11:00:00	14:01:00	2.572	MI
AA	64	US	SAN FRANCISCO	SFO	US	NEW YORK	JFK	5:21	09:00:00	17:21:00	2.572	MI
AA	6002	US	NEW YORK	JFK	DE	FRANKFURT	FRA	7:24	16:36:00	05:06:40	6.162	KM
AB	5005	DE	FRANKFURT	FRA	US	NEW YORK	JFK	7:24	12:36:00	01:06:40	6.162	KM
AF	5004	DE	FRANKFURT	FRA	US	NEW YORK	JFK	7:24	08:44:40	21:08:40	6.162	KM
AZ	555	IT	ROME	FCO	DE	FRANKFURT	FRA	2:05	19:00:00	21:05:00	845	MI
AZ	788	IT	ROME	FCO	JP	TOKYO	TYO	12:55	12:00:00	08:55:00	6.130	MI
AZ	789	JP	TOKYO	TYO	IT	ROME	FCO	15:40	11:45:00	19:25:00	6.130	MI
AZ	790	IT	ROME	FCO	JP	OSAKA	KIX	13:35	10:35:00	08:10:00	6.030	MI
BA	5003	DE	FRANKFURT	FRA	US	NEW YORK	JFK	7:24	19:01:20	18:45:20	6.162	KM
BA	6003	US	NEW YORK	JFK	DE	FRANKFURT	FRA	7:24	10:36:00	23:06:40	6.162	KM
DL	106	US	NEW YORK	JFK	DE	FRANKFURT	FRA	7:55	19:35:00	09:30:00	3.851	MI
DL	1699	US	NEW YORK	JFK	US	SAN FRANCISCO	SFO	6:22	17:15:00	20:37:00	2.572	MI
DL	1984	US	SAN FRANCISCO	SFO	US	NEW YORK	JFK	5:25	10:00:00	18:25:00	2.572	MI
JL	407	JP	TOKYO	NRT	DE	FRANKFURT	FRA	12:05	13:30:00	17:35:00	9.100	KM
JL	408	DE	FRANKFURT	FRA	JP	TOKYO	NRT	11:15	20:25:00	15:40:00	9.100	KM
LH	400	DE	FRANKFURT	FRA	US	NEW YORK	JFK	7:24	10:10:00	11:34:00	6.162	KM
LH	401	US	NEW YORK	JFK	DE	FRANKFURT	FRA	7:15	18:30:00	07:45:00	6.162	KM
LH	402	DE	FRANKFURT	FRA	US	NEW YORK	JFK	7:35	13:30:00	15:05:00	6.162	KM
LH	2402	DE	FRANKFURT	FRA	DE	BERLIN	SXF	1:05	10:30:00	11:35:00	555	KM
LH	2407	DE	BERLIN	TXL	DE	FRANKFURT	FRA	1:05	07:10:00	08:15:00	555	KM
LH	5002	DE	FRANKFURT	FRA	US	NEW YORK	JFK	7:24	12:31:20	01:02:00	6.162	KM
LH	6002	US	NEW YORK	JFK	DE	FRANKFURT	FRA	7:55	23:35:00	13:30:00	3.851	MI

图 8-1

8.2 自定义输出字段的 ALV 控件实例

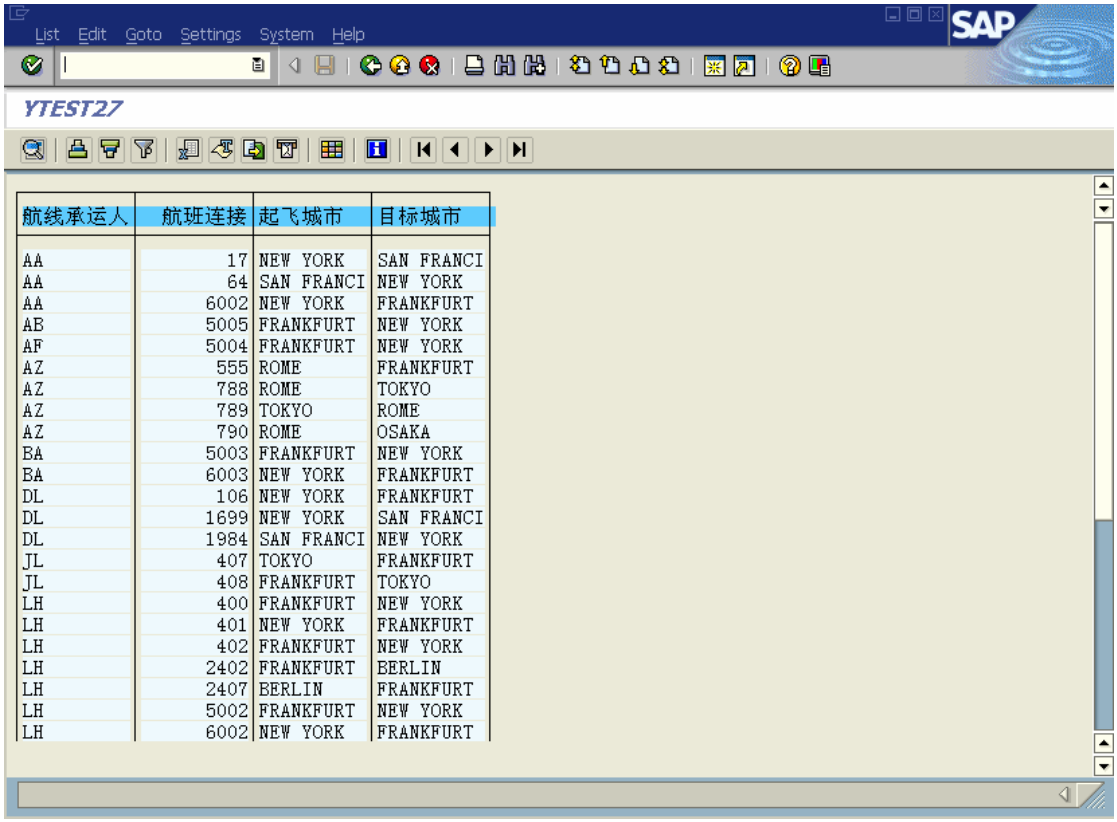
【例 8.2】

```
REPORT  YTEST27.
*ALV使用到的类库
TYPE-POOLS: SLIS.
*一列描述
DATA WA_ALV_FIELD TYPE SLIS_FIELDCAT_ALV.
*列描述内表，列清单
DATA WA_ALV_FIELDCAT TYPE SLIS_T_FIELDCAT_ALV.
*定义内表
DATA WA_SPFLI LIKE TABLE OF SPFLI WITH HEADER LINE.
*内表赋值
SELECT * INTO TABLE WA_SPFLI FROM SPFLI.

*定义第一到第四个字段
WA_ALV_FIELD-COL_POS = 1.
WA_ALV_FIELD-FIELDNAME = 'CARRID'.
WA_ALV_FIELD-SELTEXT_M = '航线承运人'.
APPEND WA_ALV_FIELD TO WA_ALV_FIELDCAT.

WA_ALV_FIELD-COL_POS = 2.
```

```
WA_ALV_FIELD-FIELDNAME = 'CONNID'.
WA_ALV_FIELD-SELTEXT_M = '航班连接'.
APPEND WA_ALV_FIELD TO WA_ALV_FIELDCAT.

WA_ALV_FIELD-COL_POS = 3.
WA_ALV_FIELD-FIELDNAME = 'CITYFROM'.
WA_ALV_FIELD-SELTEXT_M = '起飞城市'.
APPEND WA_ALV_FIELD TO WA_ALV_FIELDCAT.

WA_ALV_FIELD-COL_POS = 4.
WA_ALV_FIELD-FIELDNAME = 'CITYTO'.
WA_ALV_FIELD-SELTEXT_M = '目标城市'.
APPEND WA_ALV_FIELD TO WA_ALV_FIELDCAT.

*调用ALV显示表单数据
CALL FUNCTION 'REUSE_ALV_LIST_DISPLAY'
 EXPORTING
*   I_INTERFACE_CHECK              = ' '
*   I_BYPASSING_BUFFER             =
*   I_BUFFER_ACTIVE                = ' '
*   I_CALLBACK_PROGRAM             = ' '
*   I_CALLBACK_PF_STATUS_SET       = ' '
*   I_CALLBACK_USER_COMMAND        = ' '
*   I_STRUCTURE_NAME               =
*   IS_LAYOUT                      =
   IT_FIELDCAT                     = WA_ALV_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
* IMPORTING
*   E_EXIT_CAUSED_BY_CALLER        =
```

```
*    ES_EXIT_CAUSED_BY_USER          =
  TABLES
     T_OUTTAB                        = WA_SPFLI
* 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.
```

输出结果如图 8-2 所示。



航线承运人	航班连接	起飞城市	目标城市
AA	17	NEW YORK	SAN FRANCI
AA	64	SAN FRANCI	NEW YORK
AA	6002	NEW YORK	FRANKFURT
AB	5005	FRANKFURT	NEW YORK
AF	5004	FRANKFURT	NEW YORK
AZ	555	ROME	FRANKFURT
AZ	788	ROME	TOKYO
AZ	789	TOKYO	ROME
AZ	790	ROME	OSAKA
BA	5003	FRANKFURT	NEW YORK
BA	6003	NEW YORK	FRANKFURT
DL	106	NEW YORK	FRANKFURT
DL	1699	NEW YORK	SAN FRANCI
DL	1984	SAN FRANCI	NEW YORK
JL	407	TOKYO	FRANKFURT
JL	408	FRANKFURT	TOKYO
LH	400	FRANKFURT	NEW YORK
LH	401	NEW YORK	FRANKFURT
LH	402	FRANKFURT	NEW YORK
LH	2402	FRANKFURT	BERLIN
LH	2407	BERLIN	FRANKFURT
LH	5002	FRANKFURT	NEW YORK
LH	6002	NEW YORK	FRANKFURT

图 8-2

8.3 在屏幕上建立 ALV 控件

8.3.1 定义 SCREEN 窗口

如图 8-3 所示，在屏幕上创建两个文本元素控件、一个退出按钮控件、一个定制控制控件。

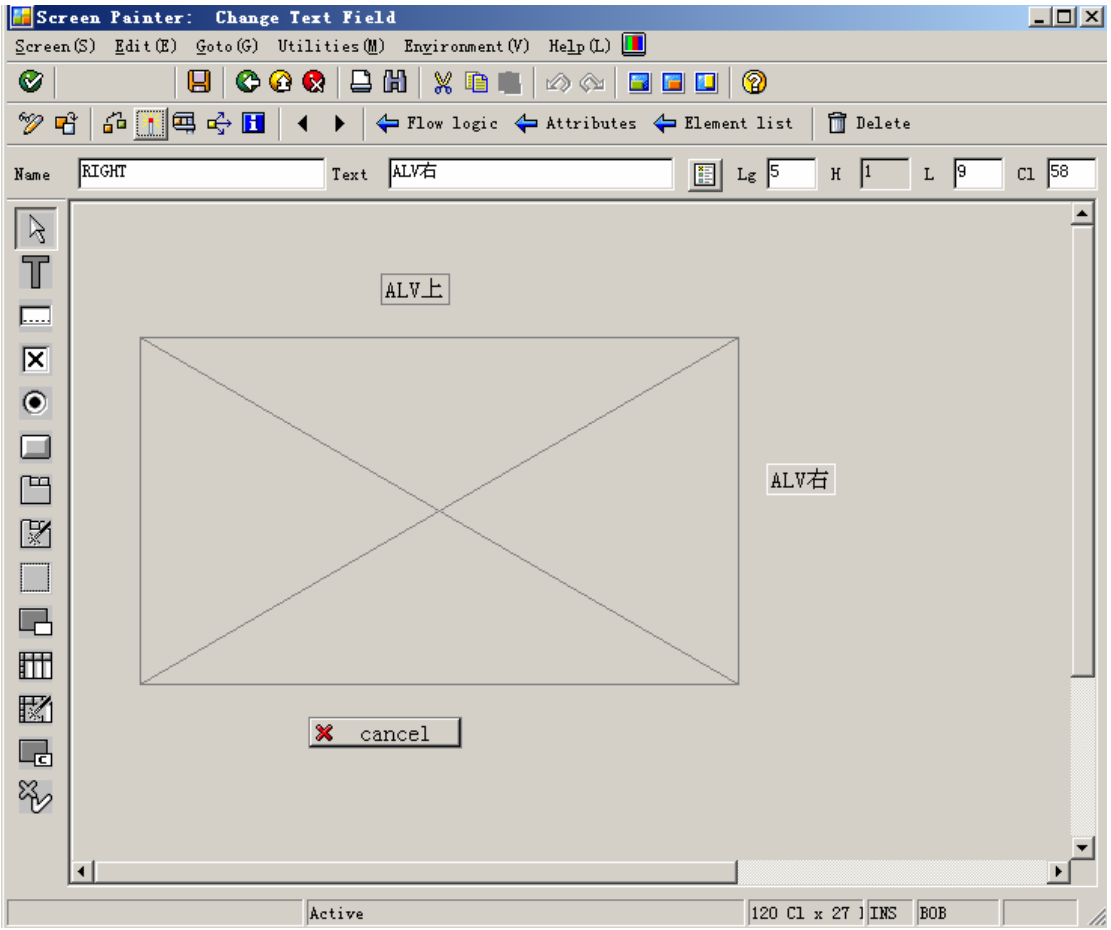


图 8-3

8.3.2 定义逻辑流

逻辑流程序：

```
*逻辑流
*PBO显示屏幕前的处理
PROCESS BEFORE OUTPUT.
  MODULE STATUS_0100.
*PAI用户输入后的处理
PROCESS AFTER INPUT.
  MODULE USER_COMMAND_0100.
```

主程序：

```
REPORT  YTEST28.
*功能码返回值
DATA: OK_CODE TYPE SY-UCOMM,
      SAVE_OK TYPE SY-UCOMM.
*定义内表，变量需要传递，不加HEADER LINE
DATA WA_SPFLI TYPE TABLE OF SPFLI .
*内表赋值
```

```
SELECT * INTO TABLE WA_SPFLI FROM SPFLI.
* ALVDATA 是屏幕100中定义控制控件的名称
DATA: WA_CONTAINER TYPE SCRFNAME VALUE 'ALVDATA',
      ALV_GRID TYPE REF TO CL_GUI_ALV_GRID,
      WA_CUSTOM_CONTAINER TYPE REF TO CL_GUI_CUSTOM_CONTAINER.
*直接调用窗口
CALL SCREEN 100.

*&---------------------------------------------------------------------*
*&      Module  STATUS_0100  OUTPUT
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
MODULE STATUS_0100 OUTPUT.
  SET PF-STATUS 'STATUS1'.
*如果窗口还没有创建ALV对象则创建它
  IF WA_CUSTOM_CONTAINER IS INITIAL.
    CREATE OBJECT WA_CUSTOM_CONTAINER
      EXPORTING
        CONTAINER_NAME = WA_CONTAINER.

    CREATE OBJECT ALV_GRID
      EXPORTING
        I_PARENT = WA_CUSTOM_CONTAINER.

    CALL METHOD ALV_GRID->SET_TABLE_FOR_FIRST_DISPLAY
      EXPORTING
        I_STRUCTURE_NAME = 'SPFLI'
      CHANGING
        IT_OUTTAB        = WA_SPFLI.
  ENDIF.

ENDMODULE.                    " STATUS_0100  OUTPUT
*&---------------------------------------------------------------------*
*&      Module  USER_COMMAND_0100  INPUT
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
MODULE USER_COMMAND_0100 INPUT.
  SAVE_OK = OK_CODE.
  CLEAR OK_CODE.
  CASE SAVE_OK.
    WHEN 'EXIT'.
      LEAVE PROGRAM.
```

```
  ENDCASE.
ENDMODULE.                    " USER_COMMAND_0100  INPUT
```

输出结果如图 8-4 所示。

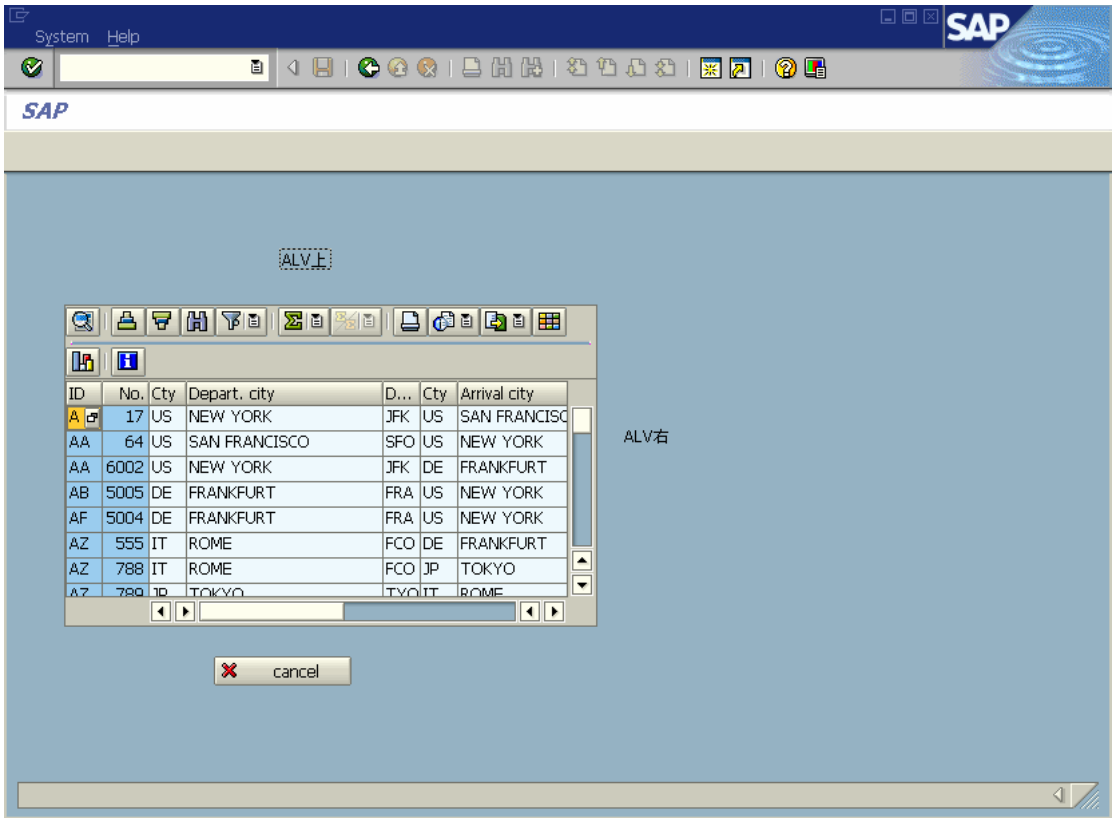
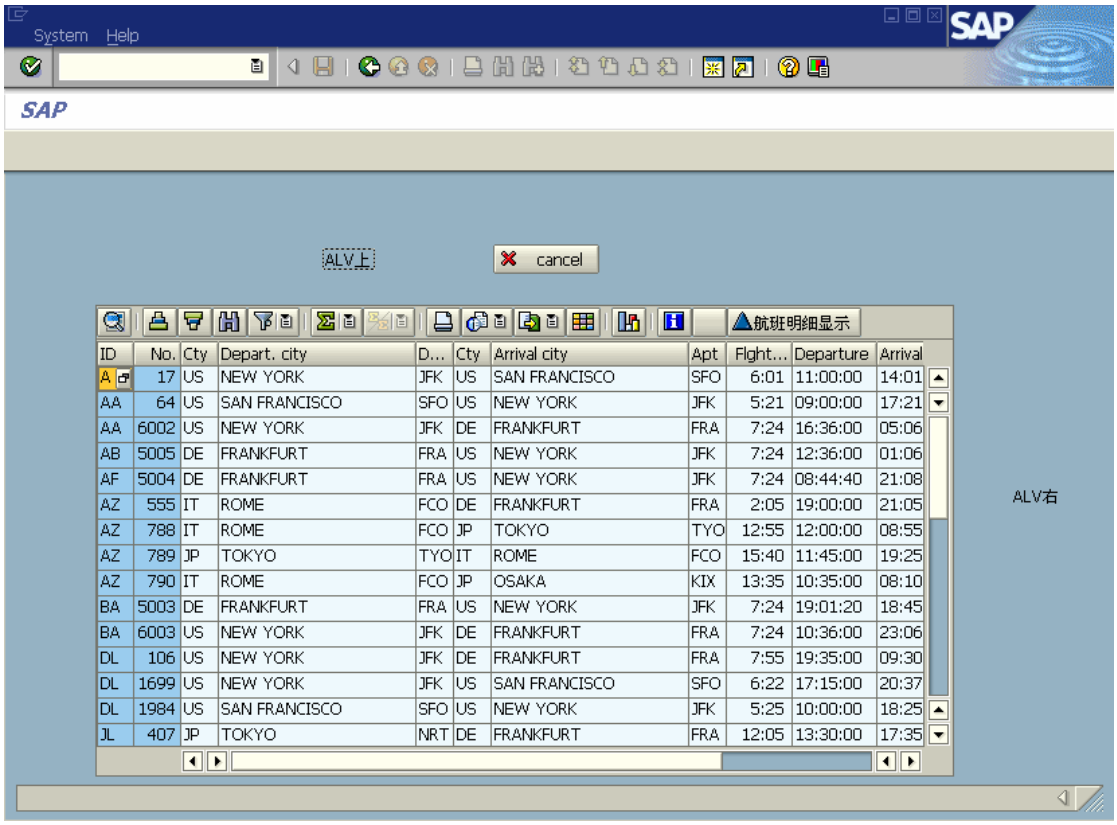


图 8-4

8.4 自定义 ALV 控件的工具条按钮

在 ALV 的工具条上增加一个自定义的按钮，单击它弹出窗口提示选择行数据内容。执行结果如图 8-5 所示。



ID	No.	Cty	Depart. city	D...	Cty	Arrival city	Apt	Flght...	Departure	Arrival
A	17	US	NEW YORK	JFK	US	SAN FRANCISCO	SFO	6:01	11:00:00	14:01
AA	64	US	SAN FRANCISCO	SFO	US	NEW YORK	JFK	5:21	09:00:00	17:21
AA	6002	US	NEW YORK	JFK	DE	FRANKFURT	FRA	7:24	16:36:00	05:06
AB	5005	DE	FRANKFURT	FRA	US	NEW YORK	JFK	7:24	12:36:00	01:06
AF	5004	DE	FRANKFURT	FRA	US	NEW YORK	JFK	7:24	08:44:40	21:08
AZ	555	IT	ROME	FCO	DE	FRANKFURT	FRA	2:05	19:00:00	21:05
AZ	788	IT	ROME	FCO	JP	TOKYO	TYO	12:55	12:00:00	08:55
AZ	789	JP	TOKYO	TYO	IT	ROME	FCO	15:40	11:45:00	19:25
AZ	790	IT	ROME	FCO	JP	OSAKA	KIX	13:35	10:35:00	08:10
BA	5003	DE	FRANKFURT	FRA	US	NEW YORK	JFK	7:24	19:01:20	18:45
BA	6003	US	NEW YORK	JFK	DE	FRANKFURT	FRA	7:24	10:36:00	23:06
DL	106	US	NEW YORK	JFK	DE	FRANKFURT	FRA	7:55	19:35:00	09:30
DL	1699	US	NEW YORK	JFK	US	SAN FRANCISCO	SFO	6:22	17:15:00	20:37
DL	1984	US	SAN FRANCISCO	SFO	US	NEW YORK	JFK	5:25	10:00:00	18:25
JL	407	JP	TOKYO	NRT	DE	FRANKFURT	FRA	12:05	13:30:00	17:35

图 8-5

单击自定义按钮后输出如图 8-6 所示。

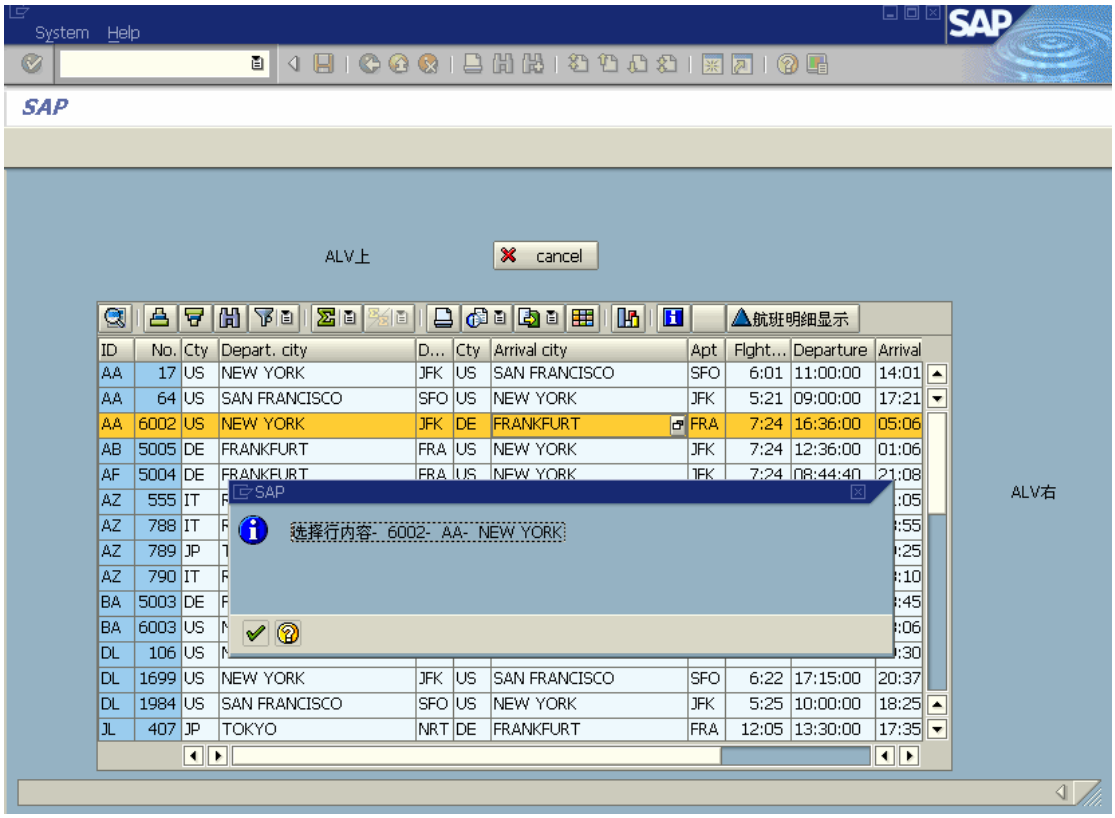


图 8-6

程序处理流程的说明：

定义 ALV 控件相关事件、接口和实现方法等，主要定义以下事件。
（1）ALV 控件的工具条处理事件，定义了新按钮和功能码；
（2）ALV 控件的功能码处理事件，定义用户单击按钮产生的功能码事件。

主程序：

```
REPORT  YTEST29.

INCLUDE <ICON>.

CLASS LCL_EVENT_RECEIVER DEFINITION DEFERRED.

DATA: OK_CODE TYPE SY-UCOMM,
      SAVE_OK TYPE SY-UCOMM.

DATA EVENT_RECEIVER TYPE REF TO LCL_EVENT_RECEIVER.

DATA: WA_SPFLI TYPE TABLE OF SPFLI,
      A_SPFLI LIKE SPFLI.

SELECT * INTO TABLE WA_SPFLI FROM SPFLI.

DATA: WA_CONTAINER TYPE SCRFNAME VALUE 'ALVDATA',
      ALV_GRID TYPE REF TO CL_GUI_ALV_GRID,
      WA_CUSTOM_CONTAINER TYPE REF TO CL_GUI_CUSTOM_CONTAINER.

CALL SCREEN 100.

*---------------------------------------------------------------------*
*       CLASS lcl_event_receiver DEFINITION
*---------------------------------------------------------------------*
*
*---------------------------------------------------------------------*
CLASS LCL_EVENT_RECEIVER DEFINITION.

  PUBLIC SECTION.
    METHODS:
      HANDLE_TOOLBAR
        FOR EVENT TOOLBAR OF CL_GUI_ALV_GRID
         IMPORTING E_OBJECT E_INTERACTIVE,

      HANDLE_USER_COMMAND
        FOR EVENT USER_COMMAND OF CL_GUI_ALV_GRID
          IMPORTING E_UCOMM.
```

```
ENDCLASS.                    "lcl_event_receiver DEFINITION

*---------------------------------------------------------------------*
*       CLASS lcl_event_receiver IMPLEMENTATION
*---------------------------------------------------------------------*
*
*---------------------------------------------------------------------*
CLASS LCL_EVENT_RECEIVER IMPLEMENTATION.

  METHOD HANDLE_TOOLBAR.
    DATA: LS_TOOLBAR TYPE STB_BUTTON.
    CLEAR LS_TOOLBAR-BUTN_TYPE.
    APPEND LS_TOOLBAR TO E_OBJECT->MT_TOOLBAR.
    CLEAR LS_TOOLBAR.

    MOVE 'SHOW_DETA' TO LS_TOOLBAR-FUNCTION.

    MOVE ICON_PPE_VNODE TO LS_TOOLBAR-ICON.
    MOVE '航班明细显示' TO LS_TOOLBAR-QUICKINFO.
    MOVE '航班明细显示'(112) TO LS_TOOLBAR-TEXT.
    MOVE '' TO LS_TOOLBAR-DISABLED.
    APPEND LS_TOOLBAR TO E_OBJECT->MT_TOOLBAR.
  ENDMETHOD.                    "handle_toolbar

  METHOD HANDLE_USER_COMMAND.
    DATA: LT_ROWS TYPE LVC_T_ROW.
    CASE E_UCOMM.
      WHEN 'SHOW_DETA'.
        CALL METHOD ALV_GRID->GET_SELECTED_ROWS
          IMPORTING
            ET_INDEX_ROWS = LT_ROWS.
        CALL METHOD CL_GUI_CFW=>FLUSH.
        IF SY-SUBRC = 0.
          MESSAGE S005(YMESS) WITH '已选择行!'.
          PERFORM MESSDETA TABLES LT_ROWS.
        ENDIF.
    ENDCASE.
  ENDMETHOD.                    "HANDLE_USER_COMMAND
ENDCLASS.                    "LCL_EVENT_RECEIVER IMPLEMENTATION
*&---------------------------------------------------------------------*
*&      Module  STATUS_0100  OUTPUT
*&---------------------------------------------------------------------*
*       text
*---------------------------------------------------------------------*
```

```
MODULE STATUS_0100 OUTPUT.
  SET PF-STATUS 'STATUS1'.
  IF WA_CUSTOM_CONTAINER IS INITIAL.
    CREATE OBJECT WA_CUSTOM_CONTAINER
      EXPORTING CONTAINER_NAME = WA_CONTAINER.
    CREATE OBJECT ALV_GRID
        EXPORTING I_PARENT = WA_CUSTOM_CONTAINER.

    CALL METHOD ALV_GRID->SET_TABLE_FOR_FIRST_DISPLAY
      EXPORTING
        I_STRUCTURE_NAME = 'SPFLI'
      CHANGING
        IT_OUTTAB        = WA_SPFLI.

    CREATE OBJECT EVENT_RECEIVER.
    SET HANDLER EVENT_RECEIVER->HANDLE_USER_COMMAND FOR ALV_GRID.

    SET HANDLER EVENT_RECEIVER->HANDLE_TOOLBAR FOR ALV_GRID.
    CALL METHOD ALV_GRID->SET_TOOLBAR_INTERACTIVE.
  ENDIF.
ENDMODULE.                    " STATUS_0100  OUTPUT
*&---------------------------------------------------------------------*
*&      Module  USER_COMMAND_0100  INPUT
*&---------------------------------------------------------------------*
*       text
*---------------------------------------------------------------------*
MODULE USER_COMMAND_0100 INPUT.
  SAVE_OK = OK_CODE.
  CLEAR OK_CODE.
  CASE SAVE_OK.
    WHEN 'EXIT'.
      LEAVE PROGRAM.
  ENDCASE.
ENDMODULE.                    " USER_COMMAND_0100  INPUT
*&---------------------------------------------------------------------*
*&      Form  MESSDETA
*&---------------------------------------------------------------------*
*       text
*---------------------------------------------------------------------*
*      -->P_LT_ROWS  text
*---------------------------------------------------------------------*
FORM MESSDETA  TABLES   P_ET_INDEX_ROWS STRUCTURE LVC_S_ROW.
  "Insert correct name for <...>.
  DATA: LS_SELECTED_LINE LIKE LVC_S_ROW,
```

```
        LF_ROW_INDEX TYPE LVC_INDEX.

  DATA: S1(200) TYPE C,
        S2(3) TYPE C.

  S2 = '-'.

  LOOP AT P_ET_INDEX_ROWS INTO LS_SELECTED_LINE.
    LF_ROW_INDEX = LS_SELECTED_LINE-INDEX.

    READ TABLE WA_SPFLI INDEX LF_ROW_INDEX INTO A_SPFLI.

    S1 = '选择行内容'.
    CONCATENATE S1 A_SPFLI-CONNID A_SPFLI-CARRID A_SPFLI-CITYFROM
                INTO S1 SEPARATED BY S2.
    MESSAGE I005(YMESS) WITH S1.
  ENDLOOP.
ENDFORM.                    " MESSDETA
```

8.5 处理 ALV 双击事件

【例 8.3】

主程序：

```
REPORT  YTEST30.
*功能码返回值
DATA: OK_CODE TYPE SY-UCOMM,
      SAVE_OK TYPE SY-UCOMM.
*定义内表，变量需要传递，不加HEADER LINE
DATA WA_SPFLI TYPE TABLE OF SPFLI.
*内表赋值
SELECT * INTO TABLE WA_SPFLI FROM SPFLI.
*定义窗口定制控制,定义ALV对象
DATA: WA_CONTAINER TYPE SCRFNAME VALUE 'ALVDATA',
      ALV_GRID     TYPE REF TO CL_GUI_ALV_GRID,
      WA_CUSTOM_CONTAINER TYPE REF TO CL_GUI_CUSTOM_CONTAINER.
*定义事件类型
CLASS LCL_EVENT_RECEIVER DEFINITION DEFERRED.
*定义事件
DATA EVENT_RECEIVER TYPE REF TO LCL_EVENT_RECEIVER.
*直接调用窗口
CALL SCREEN 100.

*---------------------------------------------------------------------*
*       CLASS lcl_event_receiver DEFINITION
```

```
*---------------------------------------------------------------------*
* 定义事件的属性和方法
*---------------------------------------------------------------------*
CLASS LCL_EVENT_RECEIVER DEFINITION.
  PUBLIC SECTION.
    METHODS:
    HANDLE_DOUBLE_CLICK
      FOR EVENT DOUBLE_CLICK OF CL_GUI_ALV_GRID
        IMPORTING E_ROW E_COLUMN.
ENDCLASS.                    "lcl_event_receiver DEFINITION

*---------------------------------------------------------------------*
*       CLASS lcl_event_receiver  IMPLENTATION
*---------------------------------------------------------------------*
* 双击方法实现
*---------------------------------------------------------------------*
CLASS LCL_EVENT_RECEIVER IMPLEMENTATION.
  METHOD HANDLE_DOUBLE_CLICK.
    DATA: LI_SPFLI LIKE LINE OF WA_SPFLI.
    READ TABLE WA_SPFLI INDEX E_ROW-INDEX INTO LI_SPFLI.
*将行列等信息合并到字符串
    DATA: S1(100) TYPE C.
    CONCATENATE '行:' E_ROW-INDEX '列名:' E_COLUMN-FIELDNAME INTO S1.
    CONCATENATE S1 'connid:' LI_SPFLI-CONNID INTO S1.
    CONCATENATE S1 'carrid:' LI_SPFLI-CARRID INTO S1.
*在状态条显示单击的行与列信息
    MESSAGE S208(00) WITH S1.
  ENDMETHOD.                    "handle_double_click
ENDCLASS.                    "lcl_event_receiver  IMPLENTATION
*&---------------------------------------------------------------------*
*&      Module  STATUS_0100  OUTPUT
*&---------------------------------------------------------------------*
* 定义状态条、包括菜单、工具条按钮、系统按钮等
*---------------------------------------------------------------------*
MODULE STATUS_0100 OUTPUT.
  SET PF-STATUS 'STATUS1'.
*如果窗口还没有创建ALV对象则创建它
  IF WA_CUSTOM_CONTAINER IS INITIAL.
    CREATE OBJECT WA_CUSTOM_CONTAINER
      EXPORTING CONTAINER_NAME = WA_CONTAINER.
    CREATE OBJECT ALV_GRID
      EXPORTING I_PARENT = WA_CUSTOM_CONTAINER.
    CALL METHOD ALV_GRID->SET_TABLE_FOR_FIRST_DISPLAY
      EXPORTING
```

```
        I_STRUCTURE_NAME = 'SPFLI'
      CHANGING
        IT_OUTTAB        = WA_SPFLI.
*ALV对象分配双击事件
    CREATE OBJECT EVENT_RECEIVER.
    SET HANDLER EVENT_RECEIVER->HANDLE_DOUBLE_CLICK
    FOR ALV_GRID.
  ENDIF.
ENDMODULE.                    " STATUS_0100  OUTPUT
*&---------------------------------------------------------------------*
*&      Module  USER_COMMAND_0100  INPUT
*&---------------------------------------------------------------------*
* 用户交互
*---------------------------------------------------------------------*
MODULE USER_COMMAND_0100 INPUT.
  SAVE_OK = OK_CODE.
  CLEAR OK_CODE.
  CASE SAVE_OK.
    WHEN 'EXIT'.
      LEAVE PROGRAM.
  ENDCASE.
ENDMODULE.                    " USER_COMMAND_0100  INPUT
```

输出界面如图 8-7 所示。双击时，注意状态条显示的信息：行号、选中列名、航班表中的 CONNID 和 CARRID 的值。

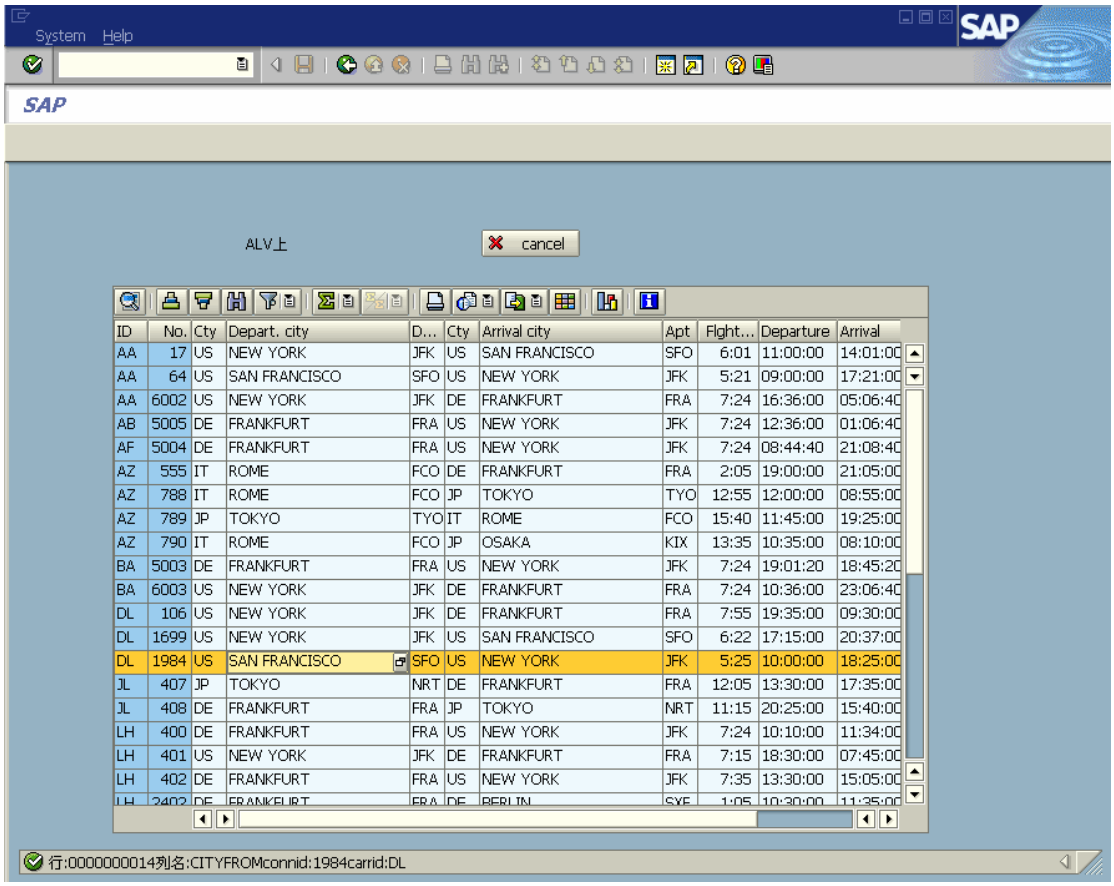


图 8-7

8.6 通过 ALV 控件编辑内表和数据库更新

处理过程如下：

（1）设定 ALV 控件可以编辑；

（2）退出屏幕时将数据更新到内表；

（3）捕捉 ALV 控件的数据更改信息，将 ALV 控件的删除行信息保存到内表中；

（4）在输出时，比较删除行和最后的内表，删除重复的行；

（5）将数据更新到数据表。

【例 8.4】

主程序：

```
REPORT  YTEST31.
DATA: OK_CODE TYPE SY-UCOMM,
      SAVE_OK TYPE SY-UCOMM.
TABLES SPFLI.

DATA LS_SPFLI TYPE SPFLI.

DATA WA_SPFLI TYPE TABLE OF SPFLI.

DATA WADEL_SPFLI TYPE TABLE OF SPFLI.
```

```
SELECT * INTO TABLE WA_SPFLI FROM SPFLI.

DATA: WA_CONTAINER TYPE SCRFNAME VALUE 'ALVDATA',
      ALV_GRID TYPE REF TO CL_GUI_ALV_GRID,
      WA_CUSTOM_CONTAINER TYPE REF TO CL_GUI_CUSTOM_CONTAINER.

DATA WA_LAYOUT TYPE LVC_S_LAYO.
WA_LAYOUT-EDIT = 'X'.
CLASS LCL_EVENT_RECEIVER DEFINITION DEFERRED.

DATA EVENT_RECEIVER TYPE REF TO LCL_EVENT_RECEIVER.

*---------------------------------------------------------------------*
*       CLASS lcl_event_receiver DEFINITION
*---------------------------------------------------------------------*
*
*---------------------------------------------------------------------*
CLASS LCL_EVENT_RECEIVER DEFINITION.

  PUBLIC SECTION.
    TYPES: DEL_ROWS TYPE STANDARD TABLE OF SPFLI.
    DATA: DDEL_ROWS TYPE STANDARD TABLE OF SPFLI.

    METHODS:
    HANDLE_DATA_CHANGED
      FOR EVENT DATA_CHANGED OF CL_GUI_ALV_GRID
        IMPORTING ER_DATA_CHANGED.

    METHODS:
    UPDATE_DELTA_TABLES
      IMPORTING
        PR_DATA_CHANGED TYPE REF TO CL_ALV_CHANGED_DATA_PROTOCOL.

    METHODS:
    GET_DELETED_ROWS
    EXPORTING
      DELETED_ROWS TYPE DEL_ROWS.
ENDCLASS.                    "LCL_EVENT_RECEIVER DEFINITION
*---------------------------------------------------------------------*
*       CLASS LCL_EVENT_RECEIVER IMPLEMENTATION
*---------------------------------------------------------------------*
*
*---------------------------------------------------------------------*
```

```
CLASS LCL_EVENT_RECEIVER IMPLEMENTATION.

  METHOD HANDLE_DATA_CHANGED.
    CALL METHOD UPDATE_DELTA_TABLES( ER_DATA_CHANGED ).
  ENDMETHOD.                    "HANDLE_DATA_CHANGED

  METHOD UPDATE_DELTA_TABLES.

    DATA: L_DEL_ROW TYPE LVC_S_MOCE.

    LOOP AT PR_DATA_CHANGED->MT_DELETED_ROWS INTO L_DEL_ROW.
      READ TABLE WA_SPFLI INTO LS_SPFLI INDEX L_DEL_ROW-ROW_ID.
      IF SY-SUBRC NE 0.
        MESSAGE E208(00) WITH '处理错误'.
      ELSE.
        APPEND LS_SPFLI TO DDEL_ROWS.
      ENDIF.
    ENDLOOP.
  ENDMETHOD.                    "update_delta_tables
  METHOD GET_DELETED_ROWS.
    DELETED_ROWS = ME->DDEL_ROWS.
  ENDMETHOD.                    "get_deleted_rows
ENDCLASS.                    "LCL_EVENT_RECEIVER DEFINITION

START-OF-SELECTION.
  CALL SCREEN 100.
  WRITE / '删除的内表记录'.
  WRITE / '______________'.
  CALL METHOD EVENT_RECEIVER->GET_DELETED_ROWS
    IMPORTING
      DELETED_ROWS = WADEL_SPFLI.

  LOOP AT WADEL_SPFLI INTO SPFLI.
    WRITE: / SPFLI-CARRID,SPFLI-CONNID,SPFLI-CITYFROM.
  ENDLOOP.

  WRITE: / '更新后的内表记录：'.
  WRITE: / '_________________'.
  LOOP AT WA_SPFLI INTO SPFLI.
    WRITE : / SPFLI-CARRID,SPFLI-CONNID,SPFLI-CITYFROM.
  ENDLOOP.

*WHEN 'SAVE'.
*  MODIFY SPFLI FORM TABLE WA_SPFLI.
```

```
*
*  IF SY-SUBRC NE 0.
*    MESSAGE I005(YMESS) WITH '更新数据错误！'.
*    EXIT.
*  ELSE.
*    MESSAGE I005(YMESS) WITH '更新数据OK！'.
*  ENDIF.
*  DELETE SPFLI FROM TABLE DELA_SPFLI.
*  IF SY-SUBRC NE 0.
*    MESSAGE I005(YMESS) WITH '更新数据错误！'.
*  ELSE.
*    MESSAGE I005(YMESS) WITH '更新数据OK！'.
*  ENDIF.

END-OF-SELECTION.
*&---------------------------------------------------------------------*
*&      Module  STATUS_0100  OUTPUT
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
MODULE STATUS_0100 OUTPUT.
  SET PF-STATUS 'STATUS1'.
  IF WA_CUSTOM_CONTAINER IS INITIAL.
    CREATE OBJECT WA_CUSTOM_CONTAINER
    EXPORTING CONTAINER_NAME = WA_CONTAINER.
    CREATE OBJECT ALV_GRID
      EXPORTING I_PARENT = WA_CUSTOM_CONTAINER.

    CREATE OBJECT EVENT_RECEIVER.
    SET HANDLER EVENT_RECEIVER->HANDLE_DATA_CHANGED FOR ALV_GRID.

    CALL METHOD ALV_GRID->SET_TABLE_FOR_FIRST_DISPLAY
      EXPORTING
        I_STRUCTURE_NAME = 'SPFLI'
        IS_LAYOUT        = WA_LAYOUT
      CHANGING
        IT_OUTTAB        = WA_SPFLI.
  ENDIF.

ENDMODULE.                    " STATUS_0100  OUTPUT
*&---------------------------------------------------------------------*
*&      Module  USER_COMMAND_0100  INPUT
*&---------------------------------------------------------------------*
*       text
```

```
*---------------------------------------------------------------------*
MODULE USER_COMMAND_0100 INPUT.
  SAVE_OK = OK_CODE.
  CLEAR OK_CODE.
  CASE SAVE_OK.
    WHEN 'EXIT'.
      DATA L_RET VALUE 'X'.
      CALL METHOD ALV_GRID->CHECK_CHANGED_DATA
        IMPORTING
          E_VALID = L_RET.
      LEAVE TO SCREEN 0.
  ENDCASE.
ENDMODULE.                                          "  USER_COMMAND_0100
INPUT
```

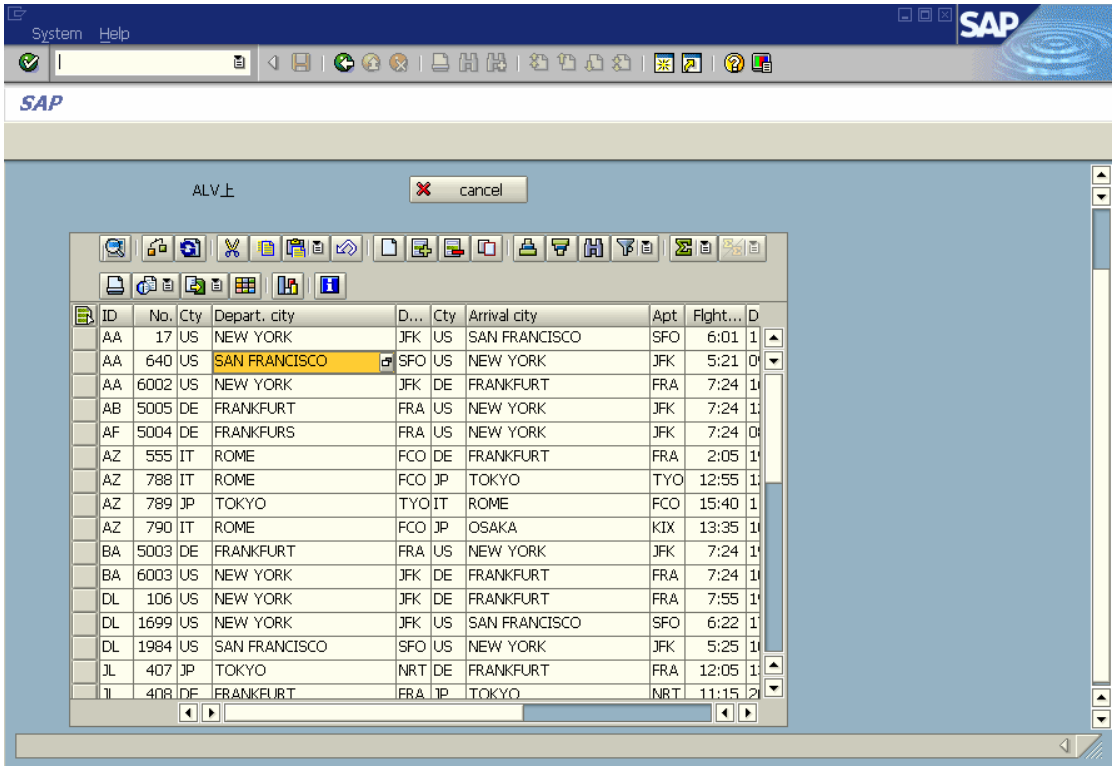


图 8-8

再删除两行后退出，输出结果如图 8-9 所示。

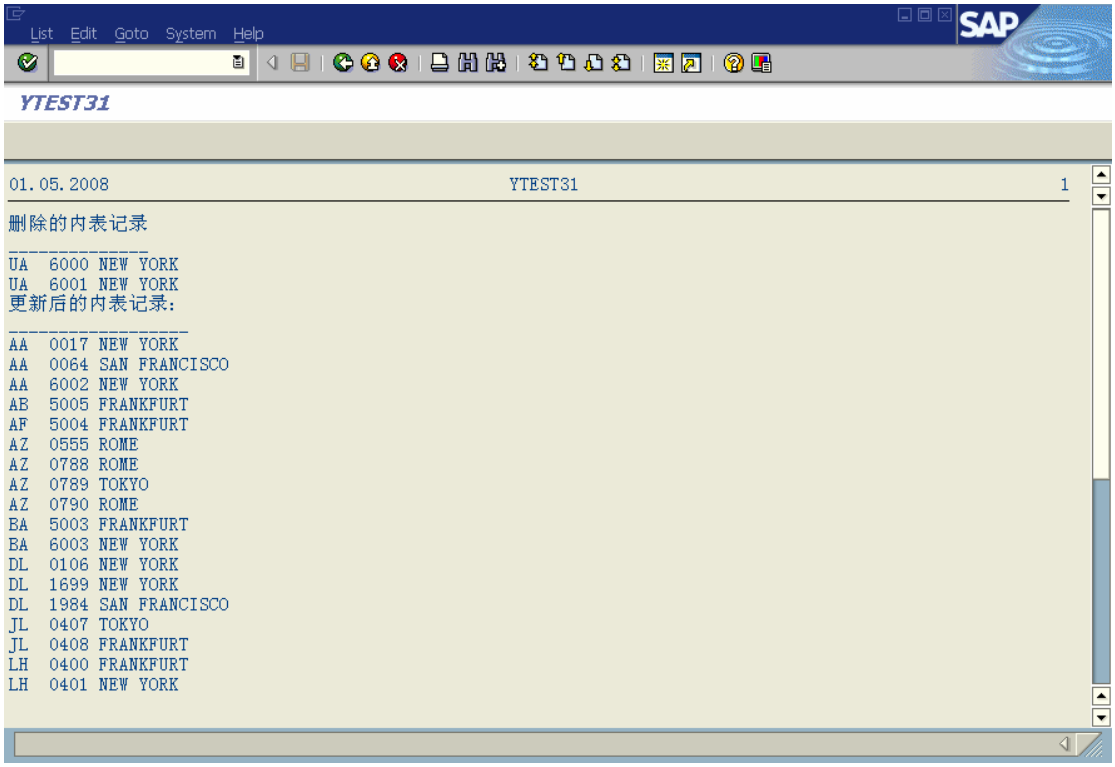


图 8-9

注：更新到数据库程序是被注释的。

8.7 ALV Tree 的使用

在 SAP 业务系统中，大量地使用到了 ALV Tree 对象，该对象在表格基础上对同类数据进行归类，并对各分类能进行数据汇总，如图 8-10 所示。

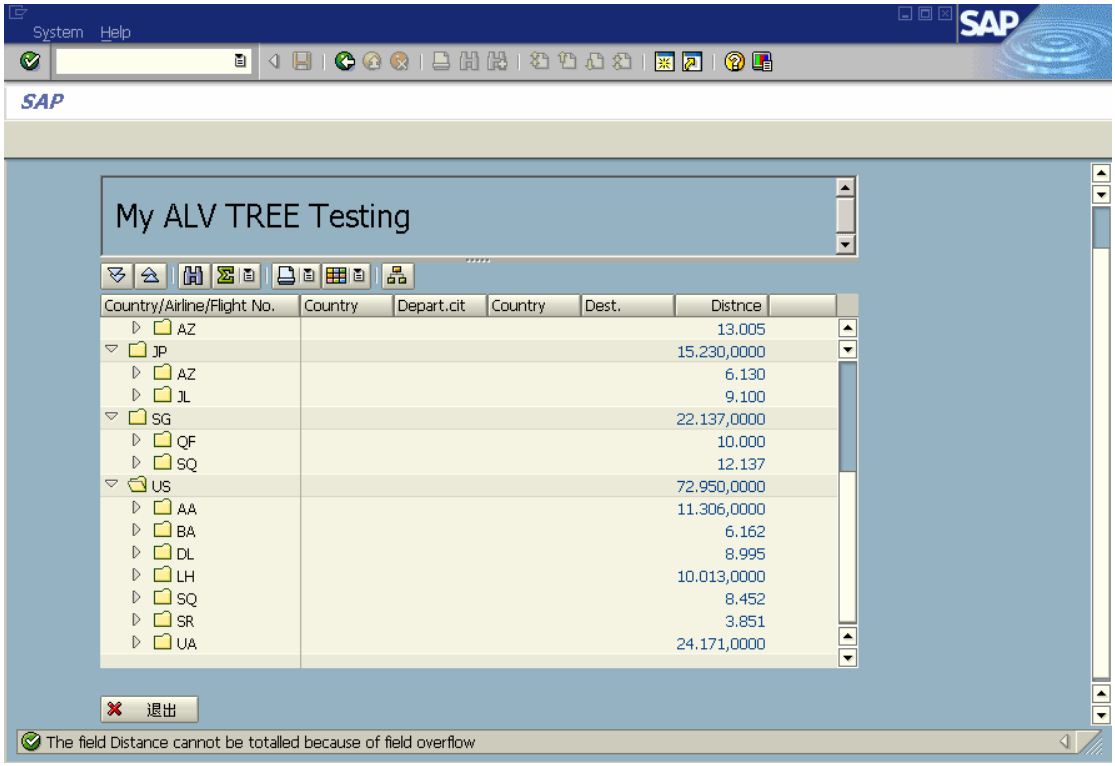


图 8-10

以航班表（SPFLI）为例：

（1）按国家、航线、航班分类；

（2）在右半屏对距离字段进行数量汇总；

（3）在右屏输出起飞国家、起飞城市、目的国家、目的城市、距离字段，并控制其输出长度。

处理过程：

（1）建立程序和屏幕，在屏幕上建立定制控制对象，定义好工作流；

（2）建立好 PAI、PBO 事件；

（3）在 PAI 中定义建立定制控制对象，并建立 AVL Tree 对象；

（4）建立 ALV Tree 对象的标题；

（5）建立右屏输出字段清单、字段长度、汇总字段等内容；

（6）建立 Tree 分类字段清单及输出先后顺序；

（7）显示 ALV Tree 对象。

主程序：

```
REPORT  YTEST32.
DATA: OK_CODE TYPE SY-UCOMM,
      SAVE_OK TYPE SY-UCOMM.

DATA: GB_FIELDCAT TYPE LVC_T_FCAT.
DATA: GB_SORTFLD TYPE LVC_T_SORT.

DATA WA_SPFLI TYPE TABLE OF SPFLI.

SELECT * INTO TABLE WA_SPFLI FROM SPFLI.
```

```
DATA: WA_CONTAINER TYPE SCRFNAME VALUE 'ALVDATA',
      ALV_GRID TYPE REF TO CL_GUI_ALV_TREE_SIMPLE,
      WA_CUSTOM_CONTAINER TYPE REF TO CL_GUI_CUSTOM_CONTAINER.

CALL SCREEN 100.
*&---------------------------------------------------------------------*
*&      Module  STATUS_0100  OUTPUT
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
MODULE STATUS_0100 OUTPUT.
  SET PF-STATUS 'STATUS1'.
  IF WA_CUSTOM_CONTAINER IS INITIAL.
    DATA LS_LIST_COMM TYPE SLIS_T_LISTHEADER.
    DATA LS_ALIST_COMM TYPE SLIS_LISTHEADER.
    LS_ALIST_COMM-TYP = 'H'.
    LS_ALIST_COMM-INFO = 'My ALV TREE Testing'.
    APPEND LS_ALIST_COMM TO LS_LIST_COMM.

    PERFORM BLDCAT.
    PERFORM BLDSORTFLD.

    CREATE OBJECT WA_CUSTOM_CONTAINER
      EXPORTING CONTAINER_NAME = WA_CONTAINER.

    CREATE OBJECT ALV_GRID
      EXPORTING I_PARENT = WA_CUSTOM_CONTAINER.

    CALL METHOD ALV_GRID->SET_TABLE_FOR_FIRST_DISPLAY
      EXPORTING
        IT_LIST_COMMENTARY = LS_LIST_COMM
        I_STRUCTURE_NAME   = 'SPFLI'
      CHANGING
        IT_SORT            = GB_SORTFLD
        IT_FIELDCATALOG    = GB_FIELDCAT
        IT_OUTTAB          = WA_SPFLI.

    CALL METHOD ALV_GRID->EXPAND_TREE
      EXPORTING
        I_LEVEL = 1.
  ENDIF.
ENDMODULE.                    " STATUS_0100  OUTPUT
*&---------------------------------------------------------------------*
```

```
*&      Module  USER_COMMAND_0100  INPUT
*&---------------------------------------------------------------------*
*       text
*---------------------------------------------------------------------*
MODULE USER_COMMAND_0100 INPUT.
  SAVE_OK = OK_CODE.
  CLEAR OK_CODE.
  CASE SAVE_OK.
    WHEN 'EXIT'.
      LEAVE PROGRAM.
  ENDCASE.


ENDMODULE.                  " USER_COMMAND_0100  INPUT
*&---------------------------------------------------------------------*
*&      Form  BLDCAT
*&---------------------------------------------------------------------*
*       text
*---------------------------------------------------------------------*
*  -->  p1        text
*  <--  p2        text
*---------------------------------------------------------------------*
FORM BLDCAT .
  CALL FUNCTION 'LVC_FIELDCATALOG_MERGE'
    EXPORTING
      I_STRUCTURE_NAME = 'SPFLI'
    CHANGING
      CT_FIELDCAT      = GB_FIELDCAT.

  DATA LS_FLDCAT TYPE LVC_S_FCAT.
  LOOP AT GB_FIELDCAT INTO LS_FLDCAT.
    CASE LS_FLDCAT-FIELDNAME.
      WHEN 'COUNTRYFR' OR 'CITYFROM' OR 'COUNTRYTO'
           OR 'CITYTO' OR 'DISTANCE'.
        LS_FLDCAT-OUTPUTLEN = 15.
      WHEN OTHERS.
        LS_FLDCAT-NO_OUT = 'X'.
    ENDCASE.

    IF LS_FLDCAT-FIELDNAME = 'DISTANCE'.
      LS_FLDCAT-DO_SUM = 'X'.
    ENDIF.
    MODIFY GB_FIELDCAT FROM LS_FLDCAT.
  ENDLOOP.
```

```
ENDFORM.                    " BLDCAT
*&---------------------------------------------------------------------*
*&      Form  BLDSORTFLD
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
*  -->  p1        text
*  <--  p2        text
*----------------------------------------------------------------------*
FORM BLDSORTFLD .
  DATA LS_SORTFLD TYPE LVC_S_SORT.

  LS_SORTFLD-SPOS = 1.
  LS_SORTFLD-FIELDNAME = 'COUNTRYFR'.
  LS_SORTFLD-UP = 'X'.
  LS_SORTFLD-SUBTOT = 'X'.
  APPEND LS_SORTFLD TO GB_SORTFLD.

  LS_SORTFLD-SPOS = 2.
  LS_SORTFLD-FIELDNAME = 'CARRID'.
  LS_SORTFLD-UP = 'X'.
  LS_SORTFLD-SUBTOT = 'X'.
  APPEND LS_SORTFLD TO GB_SORTFLD.

  LS_SORTFLD-SPOS = 3.
  LS_SORTFLD-FIELDNAME = 'CONNID'.
  LS_SORTFLD-UP = 'X'.
  LS_SORTFLD-SUBTOT = 'X'.
  APPEND LS_SORTFLD TO GB_SORTFLD.
ENDFORM.                    " BLDSORTFLD
```

逻辑流程序：

```
PROCESS BEFORE OUTPUT.
 MODULE STATUS_0100.

PROCESS AFTER INPUT.
 MODULE USER_COMMAND_0100.
```