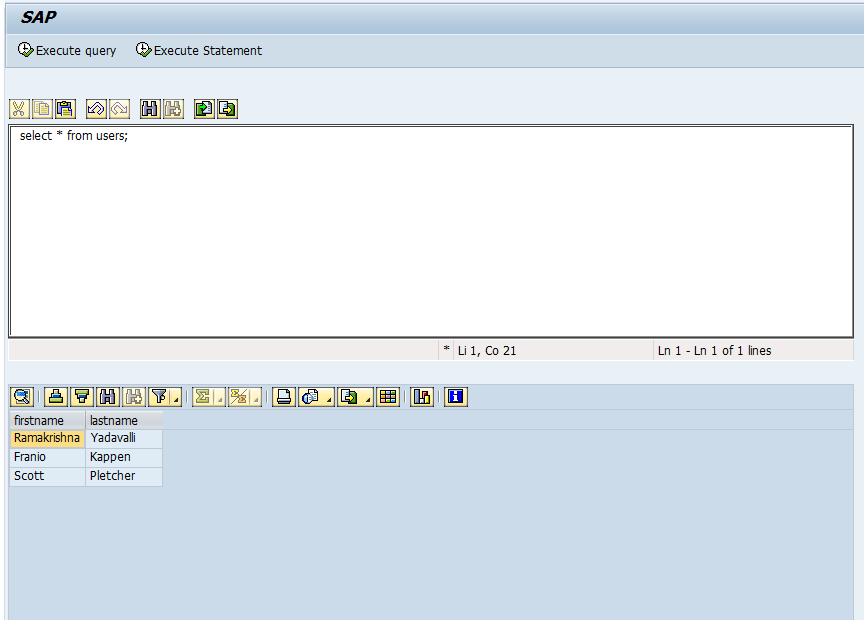
**Create transaction to execute SQL statements (ZAWS\_RSH\_PROCESS)**

You can create any abap program which consume SQL query to functional module ZAWS\_RSH\_PROCESS and display result in ALV GRID.



Source Code

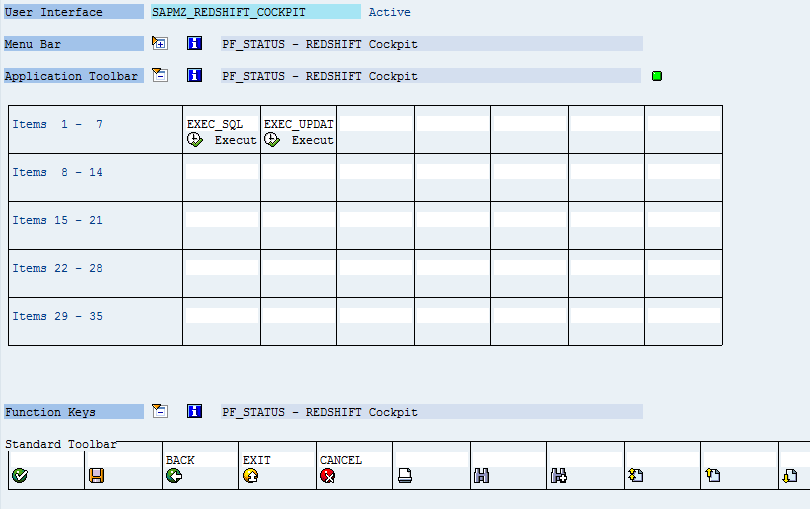
1. Main Program - SAPMZ\_REDSHIFT\_COCKPIT

\*&---------------------------------------------------------------------\*  
\*& Module Pool       SAPMZ\_REDSHIFT\_COCKPIT  
\*&  
\*&---------------------------------------------------------------------\*  
\*&  
\*&  
\*&---------------------------------------------------------------------\*  
  
  
INCLUDE ZSAPMZ\_REDSHIFT\_COCKPIT\_TOP.    " global Data  
  
INCLUDE zi\_redshift\_cockpit\_status\_o01.  
  
INCLUDE zi\_redshift\_cockpit\_userco\_i01.  
  
INCLUDE zi\_redshift\_cockpit\_performs.

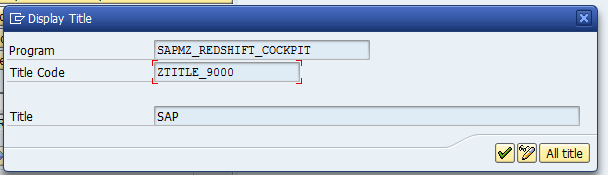
1. PBO

MODULE status\_9000 OUTPUT.  
  SET PF-STATUS 'ZRED\_9000'.  
  SET TITLEBAR 'ZTITLE\_9000'.  
  
\*  TEXTAREA  
  IF container is  INITIAL.  
    create OBJECT: container EXPORTING container\_name = 'TEXTAREA',  
                   editor EXPORTING parent = container.  
  
    editor->set\_text\_as\_stream( EXPORTING text = text\_tab ).  
  ENDIF.  
  IF o\_alv\_cont IS INITIAL.  
    CREATE OBJECT: o\_alv\_cont EXPORTING container\_name = 'ALV\_CONTAINER',  
                   o\_grid  EXPORTING i\_parent =  o\_alv\_cont.  
  ENDIF.  
  
  
  
ENDMODULE.

1. PF-STATUS



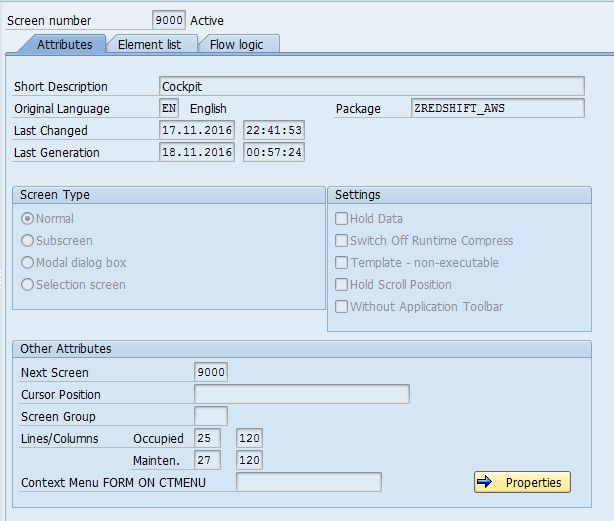
1. Title Bar

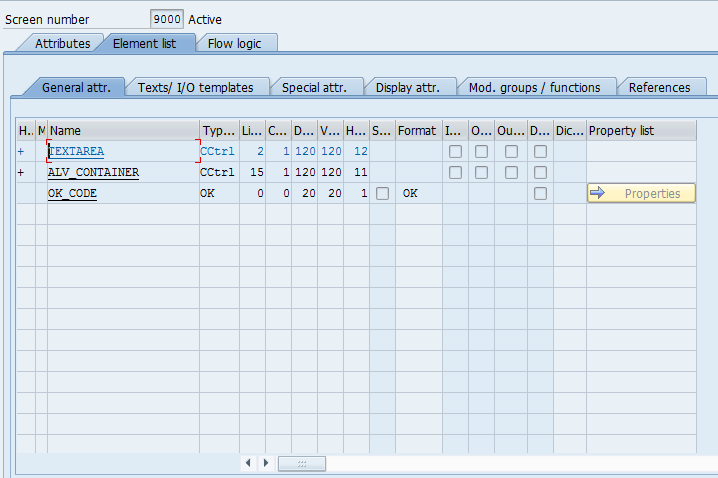


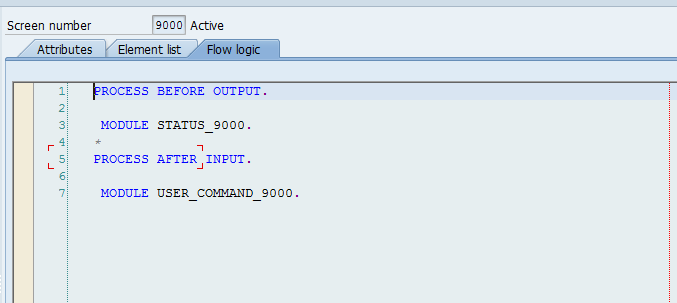
1. PAI

MODULE user\_command\_9000 INPUT.  
  
  ok\_code = sy-ucomm.  
  
  PERFORM process\_user\_input.  
  
  
ENDMODULE.

1. SCREEN 9000







1. Include - ZI\_REDSHIFT\_COCKPIT\_PERFORMS

\*----------------------------------------------------------------------\*  
\*\*\*INCLUDE ZI\_REDSHIFT\_COCKPIT\_PERFORMS.  
\*----------------------------------------------------------------------\*  
\*&---------------------------------------------------------------------\*  
\*&      Form  PROCESS\_USER\_INPUT  
\*&---------------------------------------------------------------------\*  
\*       text  
\*----------------------------------------------------------------------\*  
\*  -->  p1        text  
\*  <--  p2        text  
\*----------------------------------------------------------------------\*  
FORM process\_user\_input .  
  
  CASE ok\_code.  
    WHEN 'EXIT' OR 'CANCEL' OR 'BACK'.  
      LEAVE PROGRAM.  
  
    WHEN 'EXEC\_SQL'.  
      PERFORM send\_query\_redshift.  
    WHEN 'EXEC\_UPDAT'.  
      PERFORM send\_update\_query.  
    WHEN OTHERS.  
  ENDCASE.  
  
ENDFORM.  
\*&---------------------------------------------------------------------\*  
\*&      Form  SEND\_QUERY\_REDSHIFT  
\*&---------------------------------------------------------------------\*  
\*       text  
\*----------------------------------------------------------------------\*  
\*  -->  p1        text  
\*  <--  p2        text  
\*----------------------------------------------------------------------\*  
FORM send\_query\_redshift .  
  
  DATA: lv\_sql       TYPE string,  
        lv\_line      TYPE line,  
        lv\_rows      TYPE i,  
        lt\_fcat      TYPE lvc\_t\_fcat,  
        ls\_fcat      TYPE lvc\_s\_fcat,  
        ls\_meta\_data TYPE zaws\_rsh\_s\_metadata.  
  
  FIELD-SYMBOLS: <lfs\_table> TYPE table.  
  
  editor->get\_text\_as\_stream( IMPORTING text = text\_tab ).  
  
  CALL FUNCTION 'CONVERT\_TABLE\_TO\_STRING'  
    EXPORTING  
      i\_tabline\_length = 255  
    IMPORTING  
      e\_string         = lv\_sql  
    TABLES  
      it\_table         = text\_tab.  
  
  
  CALL FUNCTION 'ZAWS\_RSH\_PROCESS'  
    EXPORTING  
      iv\_nonquery = ''  
      iv\_sql      = lv\_sql  
    IMPORTING  
      et\_metadata = t\_metadata  
      eo\_data     = o\_data  
      ev\_rows     = lv\_rows.  
  
  
  ASSIGN o\_data->\* TO <lfs\_table>.  
  IF <lfs\_table> IS NOT ASSIGNED.  
    MESSAGE 'RFC failed to return data' TYPE 'I'.  
    RETURN.  
  ENDIF.  
  
  LOOP AT t\_metadata INTO ls\_meta\_data.  
    CLEAR ls\_fcat.  
    ls\_fcat-fieldname = ls\_meta\_data-fieldname.  
    ls\_fcat-scrtext\_l = ls\_meta\_data-fielddescr.  
    APPEND ls\_fcat TO lt\_fcat.  
  ENDLOOP.  
  
  CALL METHOD o\_grid->set\_table\_for\_first\_display  
    CHANGING  
      it\_outtab                     = <lfs\_table>  
      it\_fieldcatalog               = lt\_fcat  
    EXCEPTIONS  
      invalid\_parameter\_combination = 1  
      program\_error                 = 2  
      too\_many\_lines                = 3  
      OTHERS                        = 4.  
  IF sy-subrc <> 0.  
    MESSAGE 'Failed to display output' TYPE 'I'.  
  ENDIF.  
  
  
ENDFORM.  
\*&---------------------------------------------------------------------\*  
\*&      Form  SEND\_UPDATE\_QUERY  
\*&---------------------------------------------------------------------\*  
\*       text  
\*----------------------------------------------------------------------\*  
\*  -->  p1        text  
\*  <--  p2        text  
\*----------------------------------------------------------------------\*  
FORM send\_update\_query .  
  
  
  DATA: lv\_sql  TYPE string,  
        lv\_rows TYPE i.  
  
  
  editor->get\_text\_as\_stream( IMPORTING text = text\_tab ).  
  
  CALL FUNCTION 'CONVERT\_TABLE\_TO\_STRING'  
    EXPORTING  
      i\_tabline\_length = 255  
    IMPORTING  
      e\_string         = lv\_sql  
    TABLES  
      it\_table         = text\_tab.  
  
  
  CALL FUNCTION 'ZAWS\_RSH\_PROCESS'  
    EXPORTING  
      iv\_nonquery = 'X'  
      iv\_sql      = lv\_sql  
    IMPORTING  
      et\_metadata = t\_metadata  
      eo\_data     = o\_data  
      ev\_rows     = lv\_rows.  
  
  IF lv\_rows < 1.  
    MESSAGE 'RFC failed to update' TYPE 'I'.  
    RETURN.  
  ENDIF.  
  
  
ENDFORM.

1. Include - ZSAPMZ\_REDSHIFT\_COCKPIT\_TOP

\*&---------------------------------------------------------------------\*  
\*& Include ZSAPMZ\_REDSHIFT\_COCKPIT\_TOP                       Module Pool      SAPMZ\_REDSHIFT\_COCKPIT  
\*&  
\*&---------------------------------------------------------------------\*  
PROGRAM sapmz\_redshift\_cockpit.  
  
DATA: ok\_code    TYPE syucomm,  
      init,  
\*-- Text Area  
      container  TYPE REF TO cl\_gui\_custom\_container,  
      editor     TYPE REF TO cl\_gui\_textedit,  
      text\_tab   LIKE STANDARD TABLE OF line,  
      t\_metadata TYPE zaws\_rsh\_t\_metadata,  
      o\_data     TYPE REF TO data,  
\*-- ALV Display Container  
      o\_alv\_cont TYPE REF TO cl\_gui\_custom\_container,  
      o\_grid     TYPE REF TO cl\_gui\_alv\_grid.  
FIELD-SYMBOLS: <fs\_out\_table> TYPE table.