CSYSTEMS ™

PROGRAMSKI PAKET ZA KNJIGOVODSTVO

COBA Systems

Alaska Xbase++
DBF to PostgreSQL database application

(PART 2)
DBF file to SQL script file

20.10.2023

UVOD
INTODUCTION

U delu 1 ove knjige opisan je PGDBE (PostGreDatabaseEngine) za rad Alaska Xbase++ aplikacija sa PostgerSQL databazom. Opisana je i UPSIZE tehnologija koja se može primeniti uz PGDBE, tako da se sa PostgreSQL databazom može komunicirati putem ISAM komandi i funkcija i putem SQL komandi i funkcija. Navedena su 3 programa koji se koriste u COBA Systems poslovnim aplikacijama za rad sa PostgreSQL bazom podataka.

Part 1 of this book describes PGDBE (PostGreDatabaseEngine) for the operation of Alaska Xbase++ applications with the PostgerSQL database. UPSIZE technology, which can be applied with PGDBE, is also described, so that it is possible to communicate with the PostgreSQL database through ISAM commands and functions and through SQL commands and functions. There are 3 programs used in COBA Systems business applications for working with the PostgreSQL database.

U ovom delu 2 ove knjige je opisan i dokumentovan prvi od ta tri programa:
In this part 2 of this book, the first of those three programs is described and documented:

C-DBF2SQLFILE.EXE (developer service program)

Ovo je servisni program za izvršenje operacije prelaska sa DBF baze podataka na postgreSQL bazu podataka - uz prenos podataka iz DBF baze podataka u postgreSQL bazu podataka. Ovaj program izvršava se samo jednom i nikada više. Kada se DBF fajlovi, koji sačinjavaju DBF bazu podataka, transformišu u postgreSQL table u neku od postgreSQL baza podataka, to se više ne ponavlja. Međutim, preuzimanje podataka iz DBF fajlova u postgreSQL table može se vršiti i naknadno pa i u toku celog života postgreSQL baze podataka.
Dakle, ovaj program pomaže programeru i korisniku programa da izvrše prelaz sa Alaska Xbase++ DBF database aplikacije na Alaska Xbase++ SQL database aplikaciju.

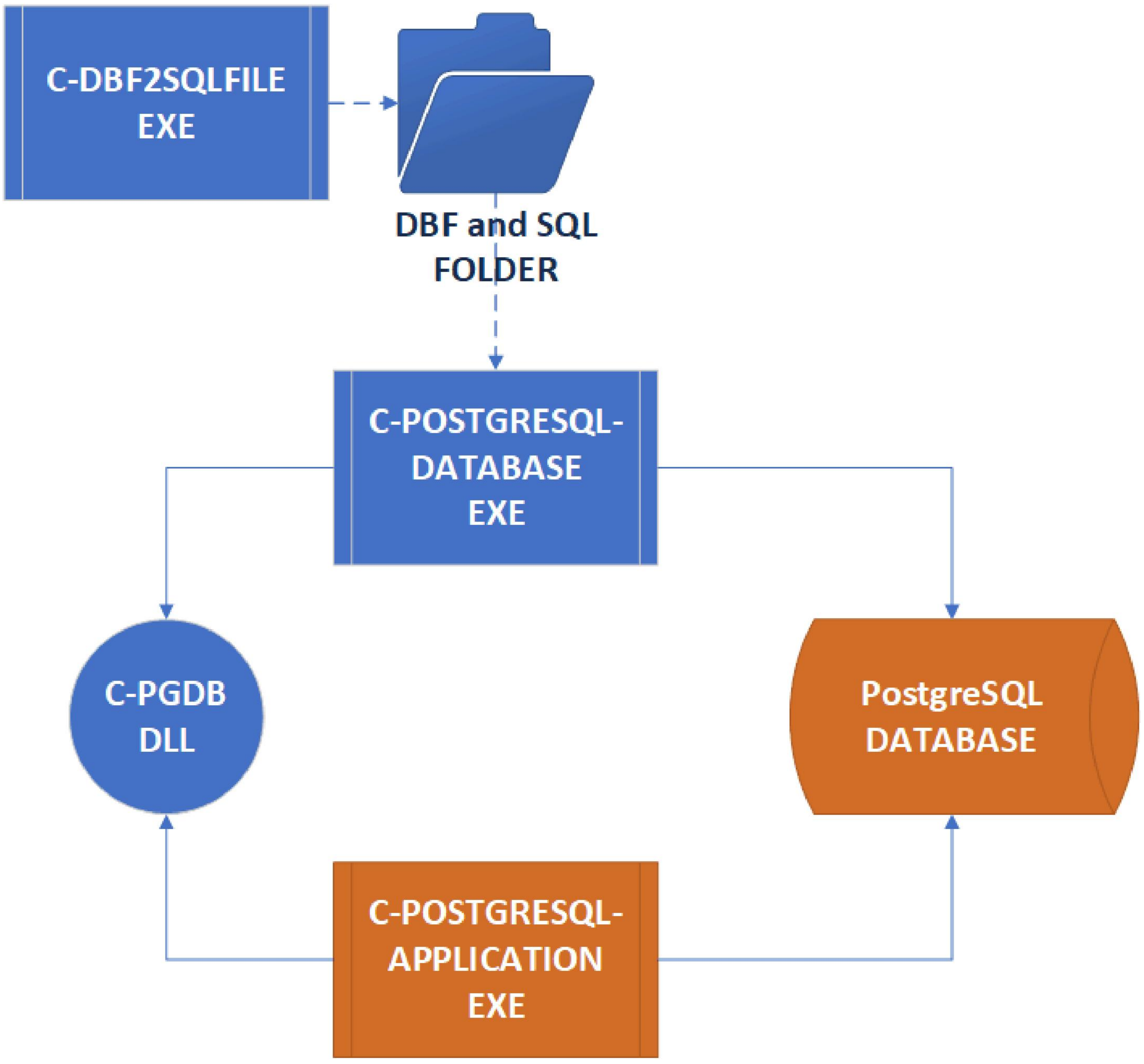
This is a service program for performing the operation of switching from the DBF database to the postgreSQL database - with the transfer of data from the DBF database to the postgreSQL database. This program is executed only once and never again. When the DBF files, which make up the DBF database, are transformed into postgreSQL tables in one of the postgreSQL databases, this is no longer repeated. However, downloading data from DBF files to postgreSQL tables can be done later and during the entire life of the postgreSQL database.
So, this program helps the developer and user of the program to make the transition from Alaska Xbase++ DBF database application to Alaska Xbase++ SQL database application.

U delu 3 ove knjige biće opisan i dokumentovan program
Part 3 of this book will describe and document the program
C-POSTRGRESQL-DATABASE.EXE (supplementary program with each EXE application)

U delu 4 ove knjige biće opisan i dokumentovan program
Part 4 of this book will describe and document the program
C-PGDB.DLL (common procedures and functions for applications)

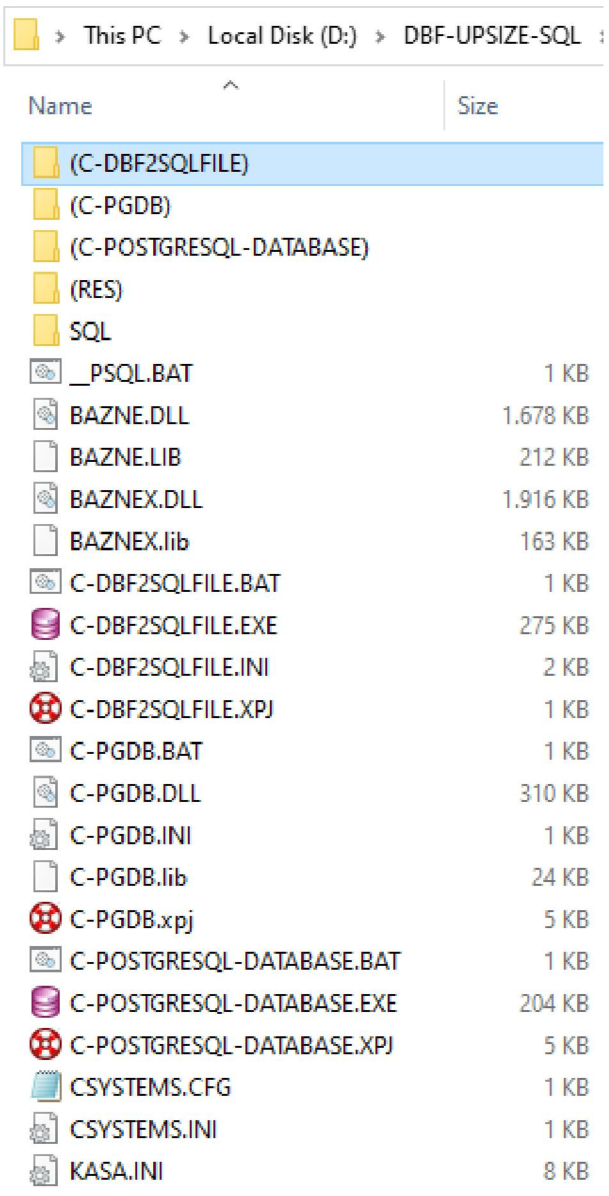
PRILOG: ŠEMA ORGANIZACIJE PROGRAMA ZA RAD SA UPSIZE POSTGRESQL DATABAZOM

PROGRAM ORGANIZATION SCHEME FOR WORKING WITH THE UPSIZE POSTGRESQL DATABASE



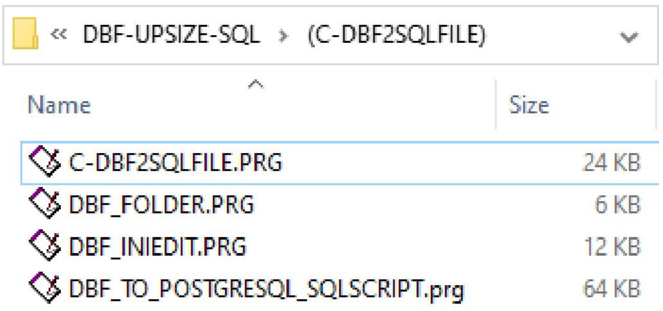
ORGANIZACIJA PROJEKTA: Alaska Xbase++ To PostgreSQL

```
C-DBF2SQLFILE.EXE              (developer service program)
C-POSTRGRESQL-DATABASE.EXE     (supplementary program with each EXE application)
C-PGDB.DLL                     (common procedures and functions for applications)
```



> This PC › Local Disk (D:) › DBF-UPSIZE-SQL

Name	Size
(C-DBF2SQLFILE)	
(C-PGDB)	
(C-POSTGRESQL-DATABASE)	
(RES)	
SQL	
_PSQL.BAT	1 KB
BAZNE.DLL	1.678 KB
BAZNE.LIB	212 KB
BAZNEX.DLL	1.916 KB
BAZNEX.lib	163 KB
C-DBF2SQLFILE.BAT	1 KB
C-DBF2SQLFILE.EXE	275 KB
C-DBF2SQLFILE.INI	2 KB
C-DBF2SQLFILE.XPJ	1 KB
C-PGDB.BAT	1 KB
C-PGDB.DLL	310 KB
C-PGDB.INI	1 KB
C-PGDB.lib	24 KB
C-PGDB.xpj	5 KB
C-POSTGRESQL-DATABASE.BAT	1 KB
C-POSTGRESQL-DATABASE.EXE	204 KB
C-POSTGRESQL-DATABASE.XPJ	5 KB
CSYSTEMS.CFG	1 KB
CSYSTEMS.INI	1 KB
KASA.INI	8 KB



> « DBF-UPSIZE-SQL › (C-DBF2SQLFILE)

Name	Size
C-DBF2SQLFILE.PRG	24 KB
DBF_FOLDER.PRG	6 KB
DBF_INIEDIT.PRG	12 KB
DBF_TO_POSTGRESQL_SQLSCRIPT.prg	64 KB

```
// C-DBF2SQLFILE.XPJ --- START

[PROJECT]
    COMPILE       = xpp
    COMPILE_FLAGS = -q
    DEBUG         = no
    GUI           = yes
    LINKER        = alink
    LINK_FLAGS    =
    RC_COMPILE    = arc
    RC_FLAGS      = -v
    OBJ_DIR       = __________C-DBF2SQLFILE.EXE
    C-DBF2SQLFILE.XPJ

[C-DBF2SQLFILE.XPJ]

   C-DBF2SQLFILE.EXE

[C-DBF2SQLFILE.EXE]
// $START-AUTODEPEND

// $STOP-AUTODEPEND

    (RES)\C-DBF2SQLFILE.RES

    dclipx.lib                // eXpress++
    XBTBASE1.LIB              // XbToolsIII
    XBTBASE2.LIB              // XbToolsIII

    bazne.lib                 // COBA Systems
    baznex.lib                // COBA Systems

    //--------------------
    // C-DBF2SQLFILE.INI
    //--------------------

    (C-DBF2SQLFILE)\C-DBF2SQLFILE.PRG       // main
    (C-DBF2SQLFILE)\DBF_FOLDER.PRG
    (C-DBF2SQLFILE)\DBF_TO_POSTGRESQL_SQLSCRIPT.PRG
    (C-DBF2SQLFILE)\DBF_INIEDIT.PRG

// C-DBF2SQLFILE.XPJ --- END
```

```
:: C-DBF2SQLFILE.BAT --- START

PBUILD C-DBF2SQLFILE.XPJ > __________.TXT
NOTEPAD __________.TXT

:: C-DBF2SQLFILE.BAT --- END
```

```
;; C-DBF2SQLFILE.INI --- START

[COBA Systems]
[DATABASE]
work_database=POSTGRESQL
work_dir=SQL
[POSTGRESQL]
```

```
work_user=postgres
work_pass=coba#1949
work_data=YES DATA
work_db=kasa
work_sema=public

;; C-DBF2SQLFILE.INI --- END
```

C-DBF2SQLFILE.PRG

```
//////////////////////////////////////////////////////////////////////////
//                                                                      //
//                                                                      //
//   C-DBF2SQLFILE.PRG
//                                                                      //
//   23-10-2023                                                         //
//                                                                      //
//   www.cobasystems.com --- COBA Systems --- Slobodan Stanojević Coba  //
//   Open Source Project BAST Business Account Software Technology      //
//   www.Alaska-Software.com --- Alaska Xbase++ version 2.0.1503        //
//   www.Donnay-software.com --- eXpress++      version 2.0.268         //
//   SAP --- Advantage Database Server ADS      version 10.10.0.49      //
//   Sergej Spirin --- FastReport for Xbase++   version 27.03.2015      //
//                                                                      //
//                                                                      //
//////////////////////////////////////////////////////////////////////////

* COMMON FUNCTION LOCATION:
* RUN_STOP()       // BAZNE.DLL
* GDE_EXE()        // BAZNE.DLL
* GDE_VER()        // BAZNE.DLL
* INI_READ()       // BAZNE.DLL
* INI_WRITE()      // BAZNE.DLL
* CTXT()           // BAZNE.DLL
* C__PROCITAJ()    // BAZNE.DLL
* C__PORUKA()      // BAZNE.DLL
* C__GRESKA()      // BAZNE.DLL
* C__NEDA()        // BAZNE.DLL
* C_PORUKA()       // BAZNE.DLL
* C_GRESKA()       // BAZNE.DLL
* FONT_CODEPAGE()  // BAZNE.DLL
* I2018()          // BAZNEX.DLL
* C_EDIT()         // BAZNEX.DLL

* APP FUNCTION IN PRG:
* PROCEDURE APPSYS()
* PROCEDURE MAIN(xffx)
* FUNCTION sql_database_parameters()
* FUNCTION DRAG_AND_DROP(xffx)
* FUNCTION cStr()
* FUNCTION USER_PASS()
* FUNCTION sql_database_parameters()
* STATIC FUNCTION help_programa()
* FUNCTION XSample_45

#pragma Library( "XppUI2.LIB" )
#pragma Library( "Adac20b.lib" )

#include "Appevent.ch"
#include "Common.ch"
#include "Xbp.ch"
#include "xbtsys.ch"

#INCLUDE "DCDIALOG.CH"
```

```
*****************************************************************************
PROCEDURE APPSYS()
*****************************************************************************
RETURN

*****************************************************************************
PROCEDURE MAIN(xffx)
*****************************************************************************
* xffx = DBF file name


     Q := RUN_STOP("GUI")          // bazne.dll
     nOldIcon := DC_ICONDEFAULT(1)
     PUBLIC KORISNIK := 0
     SET CHARSET TO ANSI
     DC_DotHotKey(180)             // DotPrompt Alt+D -> chr(180)
     SetCancel(.F.)                // turn off Alt+C
     PUBLIC EXEnaziv := "DBF-TO-SQL-SCRIPT"




          IF FILE(gde_exe()+"\SQL","D")=.F.
             dirmake( gde_exe()+"\SQL" )
          ENDIF
         *************************
         sql_database_parameters() // here
         *************************
         * PUBLIC work_database, work_user,
         * work_pass, work_data, work_sema

         *******************
         DRAG_AND_DROP(xffx)  // here
         *******************

RETURN



**************************************************************************
FUNCTION DRAG_AND_DROP(xffx)
**************************************************************************
* xffx = DBF file name

LOCAL radno:=SELECT()
LOCAL GetList := {}, oFileName, oBrowse, oDlg, GetOptions, oMsgBox

LOCAL tsize := 55
LOCAL bsize := tsize/5

DEFAULT xffx TO "FILE NOT SELECTED"

PUBLIC cFile := var2char(xffx)
PUBLIC cStr  := cStr() // here
```

```
@ 0,2 DCSAY PADC("SELECT DBF FILE or drop DBF file to EXE file icon",tsize-2) ;
            SAYSIZE tsize  COLOR GRA_CLR_DARKRED,GRA_CLR_WHITE ;
            SAYFONT "10.Consolas Bold"


@ 2,2 DCSTATIC TYPE XBPSTATIC_TYPE_GROUPBOX CAPTION "DBF FILE" ;
      SIZE tsize,10 OBJECT o1

@ 1,2 DCSTATIC TYPE XBPSTATIC_TYPE_TEXT SIZE tsize-4,8 PARENT o1 OBJECT o2;
      CAPTION {|| cStr() };  // here
      OPTIONS XBPSTATIC_TEXT_WORDBREAK COLOR GRA_CLR_DARKBLUE;
      FONT "10.Consolas"




 PRIVATE co0,co1,co2,co3,co4,of1,of2,of3,of4,aCUR,oCUR
 I2018(@co0,@co1,@co2,@co3,@co4,@of1,@of2,@of3,@of4,@aCUR,@oCUR) // baznex.dll

RESTART:=.F.

*********************************************************************
@ 10+3,2 DCTOOLBAR oToolBar SIZE tsize,3 BUTTONSIZE bsize,3
*********************************************************************




PRIVATE aa:={}
//-------------------- SELECT

    DCADDBUTTONXP CAPTION "SELECT;DBF FILE" ;
    ACTION {|| aa := FILE_DIALOG_(o2),;  // DBF_FOLDER.PRG
               cFile := aa[2]        ,;
               DC_GetRefresh(GetList) };
       MESSAGE "" ;
       PARENT oToolBar ;
       CURSOR aCUR OUTLINE COLOR co1,co2 CLICKCOLOR co3,co4

//--------------------

//-------------------- MAKE

    DCADDBUTTONXP CAPTION "MAKE;SQL FILE" ;
       ACTION {|| DBF_TO_POSTGRESQL_SQLSCRIPT(cFile),;  // DBF_TO_POSTGRESQL_SQLSCRIPT.PRG
                  DC_GetRefresh(GetList) } ;
       MESSAGE "";
       PARENT oToolBar ;
       CURSOR aCUR OUTLINE COLOR co1,co2 CLICKCOLOR co3,co4


//--------------------

//-------------------- CONFIG

    DCADDBUTTONXP CAPTION "CONFIG;SQL FILE" ;
       ACTION {|| DBF_INIEDIT(),;  // DBF_INIEDIT.PRG
                  DC_GetRefresh(GetList)} ;
       MESSAGE "";
       PARENT oToolBar ;
       CURSOR aCUR OUTLINE COLOR co1,co2 CLICKCOLOR co3,co4

//--------------------

//--------------------  HELP

//  DCADDBUTTON CAPTION " " PARENT oToolbar SIZE (bsize*6.5) HIDE {||.T.}

    DCADDBUTTONXP CAPTION "Help" ;
    PARENT oToolbar;
      ACCELKEY xbeK_F1 ;
      ACTION {|| help_programa() } ;
      MESSAGE "Help [F1]" ;
      CURSOR aCUR OUTLINE COLOR co1,co2 CLICKCOLOR co3,co4
//--------------------

//--------------------  EXIT

    DCADDBUTTONXP CAPTION "Exit" ;
      PARENT oToolbar ;
      ACCELKEY xbeK_ESC ;
      ACTION {|| DC_ReadGuiEvent(DCGUI_EXIT_OK,GetList) } ;
      MESSAGE "KRAJ [Esc]" ;
      CURSOR aCUR OUTLINE COLOR co1,co2 CLICKCOLOR co3,co4
//--------------------



  DCGETOPTIONS ICON 1 AUTORESIZE TABSTOP ;
  FONT "10.Archivo Narrow Bold"

  DCREAD GUI TITLE appname()+"  |  "+gde_ver() ;  // bazne.dll
   OPTIONS GetOptions ;
   FIT ;
   SETFOCUS @oBrowse ;
   PARENT @oDlg ;
   MODAL  ;
   EVAL {|o|SetAppWindow(o) }

   IF RESTART = .T.
     ***************
      MAIN(xffx)
     ***************
   ENDIF

RETURN NIL


SELECT(radno)
RETURN NIL

*************************************************************************
FUNCTION cStr()
*************************************************************************
* iz main():
* PUBLIC cstr
* PUBLIC cFile := "C\DATABASE\customer.dbf"   // dbf file path

cDBFpath := lower(cFile)
```

```
_tdbf_   := dc_path(cDBFpath,.T.)           // dbf file name   "customer.dbf"
_tsql_   := strtran(_tdbf_,".dbf")          // sql table name  "customer"
cSQLpath := gde_exe()+"\SQL\"+_tsql_+".sql"  // path - sql fajl "C\APP\customer.sql"
cTXTpath := gde_exe()+"\SQL\"+_tsql_+".txt"  // path - txt fajl "C\APP\customer.txt"


IF cDBFpath = "file not selected"
cSQLpath := cDBFpath; cTXTpath := cDBFpath
ENDIF

      PUBLIC ;
      cStr := ;
      chr(13)+;
      "Transfer Alaska Xbase++ DBF files to SQL scripts for creating SQL
tables"+chr(13)+chr(13)+;
      "From DBF file:"+chr(13)+;
      "    "+cDBFpath+chr(13)+chr(13)+;
      "files are created:"+chr(13)+;
      "    "+cSQLpath+chr(13)+;
      "    "+cTXTpath

RETURN cStr




*************************************************************************
FUNCTION USER_PASS()
*************************************************************************
* is used from: DBF_TO_POSTGRESQL_SQLSCRIPT.PRG
PUBLIC user, pass, data
**************************
sql_database_parameters()  // here
**************************
user:=alltrim(work_user)
pass:=alltrim(work_pass)
data:=UPPER(work_data)
RETURN NIL



*************************************************************************
FUNCTION sql_database_parameters()
*************************************************************************
 * sql_database_parameters()
 * PUBLIC work_database, work_user,
 * work_pass, work_data, work_db, work_sema


PUBLIC ini := gde_exe()+"\"+"C-DBF2SQLFILE.INI"

PUBLIC ;
work_database := "POSTGRESQL" ,; // ili "ORACLE" za koju bazu podataka se pravi SQL script
work_user     := "postgres"   ,; // username za bazu podataka
work_pass     := "coba"       ,; // password za bazu podataka
work_data     := "YES DATA"   ,; // work_data="YES DATA" prenesi tablu i podatke, "NO DATA"
prenesi smo tablu
work_db       := "test"       ,; // naziv baze podataka
work_sema     := "public"     ,; // naziv scheme iz baze podataka
```

```
work_dir      := "SQL"           // naziv foldera sa SQL fajlovima
                                 // koristi se u DBF_INIEDIT.PRG

//-------------------------------------------------------------------
   work_database := INI_READ("C","DATABASE","work_database",ini)

   IF empty(work_database)
      * work_database = "ORACLE" ili "POSTGRESQL"
      INI_WRITE("C","DATABASE","work_database","POSTGRESQL",ini)
      work_database := INI_READ("C","DATABASE","work_database",ini)
   ENDIF
   IF empty( INI_READ("C","DATABASE","work_dir",ini) )
      INI_WRITE("C","DATABASE","work_dir","SQL",ini)
   ENDIF
//-------------------------------------------------------------------

//-------------------------------------------------------------------

   IF empty( INI_READ("C","POSTGRESQL","work_user",ini) )
      INI_WRITE("C","POSTGRESQL","work_user","postgres",ini)
   ENDIF
   IF empty( INI_READ("C","POSTGRESQL","work_pass",ini) )
      INI_WRITE("C","POSTGRESQL","work_pass","coba",ini)
   ENDIF
   IF empty( INI_READ("C","POSTGRESQL","work_data",ini) )
      INI_WRITE("C","POSTGRESQL","work_data","YES DATA",ini)
   ENDIF
   IF empty( INI_READ("C","POSTGRESQL","work_db",ini) )
      INI_WRITE("C","POSTGRESQL","work_db","test",ini)
   ENDIF
   IF empty( INI_READ("C","POSTGRESQL","work_sema",ini) )
      INI_WRITE("C","POSTGRESQL","work_sema","public",ini)
   ENDIF

//-------------------------------------------------------------------



   work_user     := INI_READ("C","POSTGRESQL","work_user",ini)
   work_pass     := INI_READ("C","POSTGRESQL","work_pass",ini)
   work_data     := INI_READ("C","POSTGRESQL","work_data",ini)
   work_db       := INI_READ("C","POSTGRESQL","work_db"  ,ini)
   work_sema     := INI_READ("C","POSTGRESQL","work_sema",ini)



RETURN NIL


***************************************************************************
STATIC FUNCTION help_programa()
***************************************************************************
LOCAL txt := "", cr := chr(13)+chr(10)

program := appname()+" verzija "+gde_ver()

TEXT INTO txt WRAP cr //TRIMMED

   &program
```

```
---------------------------------------------------------------
SERVISNI PROGRAM ZA GENERISANJE SQL SCRIPT FAJLOVA IZ DBF FAJLA
---------------------------------------------------------------

Ovo je servisni program koji zahteva preuzimanje DBF fajla.
DBF fajl se može preuzeti komandnim dugmetom:

           [SELECT DBF FILE]

ili se može preuzeti prevlačenjem ikone DBF fajla na ikonu
ovog servisnog programa (drag and drop ikone)


Ovaj servisni program od preuzetog DBF fajla generiše
dva sql fscript fajla komandnim dugmetom:

           [MAKE SQL FILE]

Primer:

   ako je preuzet DBF fajl:

           'customer.dbf'

   biće napravljeni sql script fajlovi:

           'customer.txt'
           'customer.sql'

   i smešteni u folder u kome je i ovaj EXE program.

Upotreba:

   customer.txt
   ------------

   Fajl 'customer.txt' je sql script koji služi za kreiranje
   postgreSQL table čiji naziv je 'customer'.
   Tabla će biti kreirana u postgreSQL databazi i šemi koje
   su zadate u konfiguraciji komandnim dugmetom:

             [CONFIG SQL FILE]

   Ovaj script se upotrebljava tako što se iz njega kopira
   (copy) kod koji se nalazi između crta:

   -- Kreiranje nove table ----------------------- START
      code...
   -- Kreiranje nove table ----------------------- END

   i dobijeni string se upisuje (paste) u program:

     __tab__create__all.prg

   u funkciju:

     __tab__create_all()

   Program: __tab__create__all.prg  je sastavni deo svake
            Xbase++ aplikacije koja radi sa PostgreSQL
```

bazom podataka (sa PGDBE.DLL mašinom)

customer.sql

Fajl 'customer.sql' je sql fajl koji služi za punjenje podacima
postgreSQL table 'customer'.

Fajl 'sustomer.sql' koristi se od strane one aplikacije u
čiji program __tab__create__all.prg je upisan 'customer.txt'
na osnovu čega je ta aplikacija kreirala tablu 'customer'.
Aplikacija traži fajl 'customer.sql' u subfolderu aplikacije:

```
                        \SQL
```

i ako ga nađe, tada podacima koji su upisani u taj fajl, puni
PostgreSQL tablu 'customer'
Prethodno aplikacija isprazni tablu 'customer'.

Ova operacija punjenja PostgreSQL table podacima iz sql fajla
obavlja se samo na zahtev korisnika (klikom na komandno dugme)
a izvršava je funkcija:

```
                     __tab__create()
iz programa:
                     __tab__create.prg
koji se nalazi u:
                     C_PGDB.DLL/LIB biblioteci
```

Note:

Svaka Xbase++ PostgreSQL aplikacija firme COBA Systems ima svoj
pomoćni-dodatni program za održavanje PosgreSQL baze podataka.
Ako je naziv aplikacije:

```
                CUSTOMERS_REGISTER.EXE
```

Naziv pomoćnog-dodatnog programa je:

```
                CUSTOMERS_REGISTER_DATABASE.EXE
```

Ovaj pomoćni-dodatni program sadrži fajlove i funkcije:

```
                __tab__create__all.prg  -> __tab__create__all()
                __tab__create.prg       -> __tab__create()
```

```
gde u prg modul:
                __tab__create__all.prg
treba ugraditi:
                'customer.txt'
a prg modul:
                __tab__create.prg
će koristiti:
                '\SQL\customer.sql'
```

```
ENG


   ------------------------------------------------- -------------
   SERVICE PROGRAM FOR GENERATION OF SQL SCRIPT FILES FROM DBF FILE
   ------------------------------------------------- -------------

   This is a service program that requires the download of a DBF file.
   The DBF file can be downloaded with the command button:

              [SELECT DBF FILE]

   or it can be downloaded by dragging the DBF file icon to the icon
   of this service program (drag and drop icons)


   This service program generates from the downloaded DBF file
   two sql fscript files with the command button:

              [MAKE SQL FILE]

   Example:

      if a DBF file is downloaded:

              'customer.dbf'

      sql script files will be created:

              'customer.txt'
              'customer.sql'

      and placed in the folder where this EXE program is also.



   Usage:


      customer.txt
      ------------

      The file 'customer.txt' is a sql script used to create
      postgreSQL table named 'customer'.
      The table will be created in the postgreSQL database and schema which
      are set in the command button configuration:

                [CONFIG SQL FILE]

      This script is used by copying from it
      (copy) the code between the lines:

      -- Creating a new board ------------------------------------ START
         codes...
      -- Creating a new board ------------------------------------ END

      and the resulting string is written (pasted) into the program:
        __tab__create__all.prg

      into function:

        __tab__create_all()

      Program: __tab__create__all.prg is an integral part of each
               Xbase++ application that works with PostgreSQL
               database (with PGDBE.DLL engine)




      customer.sql
      ------------

      The file 'customer.sql' is a sql file used to load data
      postgreSQL table 'customer'.

      The file 'sustomer.sql' is used by that application in
      whose program __tab__create__all.prg is written 'customer.txt'
      based on which that application created the 'customer' panel.
      The application looks for the file 'customer.sql' in the application subfolder:

                            \SQL

      and if it finds it, then it fills with the data written in that file
      PostgreSQL panel 'customer'
      Previously, the application empties the 'customer' board.

      This operation of filling the PostgreSQL table with data from the sql file
      performed only at the request of the user (by clicking on the command button)
      and is executed by the function:
                         __tab__create()
      from the program:
                         __tab__create.prg
      located in:
                         C_PGDB.DLL/LIB library



   Notes:

      Each Xbase++ PostgreSQL application from COBA Systems has its own
      auxiliary-additional program for maintenance of PosgreSQL database.
      If the app name is:

                    CUSTOMERS_REGISTER.EXE

      The name of the auxiliary-additional program is:

                    CUSTOMERS_REGISTER_DATABASE.EXE

      This plug-in contains files and functions:

                    __tab__create__all.prg -> __tab__create__all()
                    __tab__create.prg -> __tab__create()

      where in prg module:
                    __tab__create__all.prg
```

```
       should be installed:
                       'customer.txt'
       and the prg module:
                       __tab__create.prg
       will use:
                       '\SQL\customer.sql'


ENDTEXT
  TxtFile:=cTxt()
  MEMOWRIT(TxtFile,txt)
  *******************************************
  off := Font_codepage(10,"Consolas",238,.f.) // baznex.dll
  C_EDIT(TxtFile,"DBF TO POSTGRESQL SQL SCRIPT",;
               "DBF TO POSTGRESQL SQL SCRIPT",,,,,,off) // baznex.dll
  *******************************************
   CLEAR TYPEAHEAD

RETURN NIL






*---------------------------------------------------------------  START
*      Ovo je Roger Donnay primer za drag-and-drop.
*      Treba na osnovu njega napraviti funkciju koja će
*      da izabrani dbf fajl unese u DCSTATIC objekat
*      aplikacije sa drag-and-drop (to ostavljam za kasnije)
*
*      This is a Roger Donnay drag-and-drop example.
*      It is necessary to create a function based on it that will
*      to import the selected dbf file into the DCSTATIC object
*      apps with drag-and-drop (I'll leave that for later)
*---------------------------------------------------------------



FUNCTION XSample_45

*
* Drag and Drop
*
* This sample demonstrates dragging a value from a row/column in
* an array browse to another row/column.  The value in the cell
* that is grabbed is swapped with the value in the cell dropped.
*

LOCAL GetList := {}, oBrowse, aDirectory, i

aDirectory := Directory()
FOR i := 1 TO Len(aDirectory)
  aDirectory[i,1] := Upper(aDirectory[i,1])
NEXT
ASort(aDirectory,,,{|x,y|x[1]<y[1]})

@ 3,1 DCBROWSE oBrowse DATA aDirectory ;
      SIZE 43,12 FONT '10.Helv Bold' ;
      PRESENTATION DC_BrowPres()

DCBROWSECOL ELEMENT 1 WIDTH 30 HEADER "File Name" PARENT oBrowse

DCREAD GUI ;
   FIT ;
   MODAL ;
   BUTTONS DCGUI_BUTTON_EXIT ;
   TITLE 'Drag a Value from one Row to another' ;
   HANDLER _XSample_45 REFERENCE @oBrowse

RETURN nil



STATIC FUNCTION ;
  _XSample_45 ( nEvent, mp1, mp2, oXbp, oDlg, GetList, oBrowse )

STATIC lButtonDown := .f., nTopRow, nGrabRow, cFileName, oCellGroup

LOCAL oColumn, nRowPos, nElement, lHitBottom := .f.

IF Valtype(oXbp) = 'O'

  /* -- Left button pressed in a cell -- */
  IF nEvent = xbeM_LbDown .AND. oXbp:ClassName() = 'XbpCellGroup'

    nRowPos := Int((oXbp:currentSize()[2]-mp1[2]) / (oXbp:CellRect(1)[4]-oXbp:CellRect(1)[2]))
    oBrowse:forcestable()
    nElement := oBrowse:cargo[4]  // Get current array pointer
    nTopRow := nElement - nRowPos
    nGrabRow := nElement
    oBrowse:setPointer( nil, POINTER_MOVE_1, 1)
    lButtonDown := .t.
    oCellGroup := oXbp

    /* -- Left button released in new cell -- */
  ELSEIF nEvent = xbeM_LbUp .AND. oXbp:ClassName() = 'XbpCellGroup' .AND. ;
         oXbp == oCellGroup

    nRowPos := Int((oXbp:currentSize()[2]-mp1[2]) / (oXbp:CellRect(1)[4]-oXbp:CellRect(1)[2]))
    nElement := nTopRow + nRowPos
    cFileName := oBrowse:cargo[5,nElement,1]
    oBrowse:cargo[5,nElement,1] := oBrowse:cargo[5,nGrabRow,1]
    oBrowse:cargo[5,nGrabRow,1] := cFileName
    oBrowse:refreshAll()
    oBrowse:setPointer( nil, 1, 1 )
    lButtonDown := .f.

  ELSEIF lButtonDown

    oBrowse:setPointer( nil, POINTER_MOVE_1, 1)

    /* -- Mouse moved in Bottom ScrollBar area -- */
    IF nEvent = xbeM_Motion .AND. oXbp:ClassName() = 'XbpScrollbar' .AND. ;
       oXbp:type = XBPSCROLL_HORIZONTAL
      DO WHILE oBrowse:cargo[4] < Len(oBrowse:cargo[5])
        IF oBrowse:RowPos+1 = oBrowse:RowCount
           nTopRow++
        ENDIF
        oBrowse:down()
        oBrowse:refreshAll()
        nEvent := AppEvent( @mp1, @mp2, @oXbp, .1 )
        IF nEvent = xbeM_LbUp
           EXIT
        ELSEIF nEvent = xbeP_None .OR. Valtype(oXbp) # 'O' .OR. oXbp:ClassName() =
'XbpScrollbar'
           Sleep(7)
           LOOP
        ENDIF
        EXIT
      ENDDO
    /* -- Mouse moved in Header area -- */
    ELSEIF nEvent = xbeM_Motion .AND. oXbp:ClassName() = 'XbpCellGroup' .AND. ;
           !(oXbp==oCellGroup) .AND. oXbp:setParent()==oCellGroup:setParent()
      DO WHILE oBrowse:cargo[4] > 1
        IF oBrowse:RowPos = 1
           nTopRow--
        ENDIF
        oBrowse:up()
        oBrowse:refreshAll()
        nEvent := AppEvent( @mp1, @mp2, @oXbp, .1 )
        IF nEvent = xbeM_LbUp
           EXIT
        ELSEIF nEvent = xbeP_None .OR. Valtype(oXbp) # 'O' .OR. ;
           (oXbp:ClassName() = 'XbpCellGroup' .AND. ;
            !(oXbp==oCellGroup) .AND. oXbp:setParent()==oCellGroup:setParent())
           Sleep(7)
           LOOP
        ENDIF
        EXIT
      ENDDO
    ENDIF

  ENDIF

ENDIF
IF nEvent = xbeM_LbUp
   lButtonDown := .f.
ENDIF

RETURN DCGUI_NONE
*** END OF EXAMPLE ***
*-----------------------------------------------------------------
*      Ovo je Roger Donnay primer za drag-and-drop.
*      Treba na osnovu njega napraviti funkciju koja će
*      da izabrani dbf fajl unese u DCSTATIC objekat
*      aplikacije sa drag-and-drop (to ostavljam za kasnije)
*
*      This is a Roger Donnay drag-and-drop example.
*      It is necessary to create a function based on it that will
*      to import the selected dbf file into the DCSTATIC object
*      apps with drag-and-drop (I'll leave that for later)
*-----------------------------------------------------------------  END
```

DBF_FOLDER.PRG

```
//////////////////////////////////////////////////////////////////////////
//                                                                    //
//                                                                    //
//   DBF_FOLDER.PRG
//                                                                    //
//   23-10-2023                                                       //
//                                                                    //
//   www.cobasystems.com --- COBA Systems --- Slobodan Stanojević Coba  //
//   Open Source Project BAST Business Account Software Technology    //
//   www.Alaska-Software.com --- Alaska Xbase++ version 2.0.1503      //
//   www.Donnay-software.com --- eXpress++      version 2.0.268       //
//   SAP --- Advantage Database Server ADS      version 10.10.0.49    //
//   Sergej Spirin --- FastReport for Xbase++   version 27.03.2015    //
//                                                                    //
//                                                                    //
//////////////////////////////////////////////////////////////////////////



// Uređivač teksta sa dijalogom datoteke i menijem datoteka
//----------------------------------------------------------
// Ovaj primjer prikazuje uređivač teksta koji radi u an
// XbpCrt prozor. Prozor ima traku menija koja
// uključuje podmeni za obradu tekstualnih datoteka. Ovo
// meni uključuje stavke "Novo", "Otvori", "Sačuvaj" i
// "Sačuvaj kao". Za ove menije se koristi dijalog fajla
// opcije. Pravi uređivač teksta obezbeđuje an
// XbpMLE objekt čija varijabla instance :cargo sadrži
// naziv datoteke tekuće datoteke.

// Text editor with file dialog and file menu
//----------------------------------------------------------
// This example shows a text editor that runs in an
// XbpCrt window. The window has a menu bar that
// includes a submenu for processing text files. This
// menu includes the items "New", "Open", "Save" and
// "Save As". A file dialog is used for these menu
// options. The actual text editor is provided by an
// XbpMLE object whose :cargo instance variable contains
// the file name of the current file.


* COMMON FUNCTION LOCATION:
* GDE_EXE()       // BAZNE.DLL

* APP FUNCTION IN PRG:
* FUNCTION FILE_DIALOG()
* FUNCTION TextFileMenu()
* PROCEDURE ReadFile()
* PROCEDURE WriteFile()

#include "Xbp.ch"
#include "Appevent.ch"
#pragma library("XppUi2")
```

```
*************************************************************************
FUNCTION FILE_DIALOG_()
*************************************************************************

      LOCAL nEvent, mp1, mp2, oXbp
      LOCAL oMLE, oFileDlg

       // Create file dialog with desktop as parent
      oFileDlg   := XbpFileDialog():new()
      oFileDlg:create( AppDesktop() )

      IF UPPER(work_dir)=="SQL"
         lokacija := gde_exe()+"\SQL\*.*"   // default: GDE_EXE()
         cFile := oFileDlg:open(lokacija)
      ELSE
         cFile := oFileDlg:open("*.DBF")
      ENDIF

      IF cFile = NIL // nije promenjena lokacija DBF
          RETURN {"FILE NOT SELECTED","FILE NOT SELECTED"}
      ENDIF

      // MsgBox(cFile)
      nKraj := RAT("\",cFile)
      cFolder := SUBSTR(cFile,1,nKraj-1)

      //MsgBox("=="+folder+"==")

      MemoWrit(gde_exe()+"\folder",cFolder)

   RETURN {cFolder,cFile}

   // *************************************************
   // This function creates a default menu for opening
   // and saving text files
   //

   FUNCTION TextFileMenu( oMenuBar, oFileDlg, oMLE )
      LOCAL oMenu := XbpMenu():new(oMenuBar):create()

      oMenu:title := "~File"

      oMenu:addItem( { "~New", ;
         {|| oMLE:setData(""), oMLE:cargo := "", ;
             SetAppWindow():setTitle( "No name" ), ;
             oMenu:disableItem(4) } } )
      oMenu:addItem( { "~Open...", ;
         {|| ReadFile( oFileDlg, oMLE ), ;
             IIf( Empty(oMLE:cargo), NIL, oMenu:enableItem(4)) } } )


      oMenu:addItem( { ,, XBPMENUBAR_MIS_SEPARATOR,0 } )

      oMenu:addItem( { "~Save", ;
         {|| WriteFile( oMLE:cargo, oMLE ) }, ;
         0, XBPMENUBAR_MIA_DISABLED } )

      oMenu:addItem( { "Save ~as...", ;
         {|| WriteFile( oFileDlg:saveAs( oMLE:cargo ), oMLE ) } } )

      oMenuBar:addItem( {oMenu, NIL} )
   RETURN oMenu

   // *******************************************
   // The procedure activates the file dialog for

   // selecting a file. Reads selected files and
   // copies into MLE.
   //
   PROCEDURE ReadFile( oFileDlg, oMLE )
      LOCAL cFile := oFileDlg:open( oMLE:cargo )

      IF ! Empty( cFile )
         oMLE:cargo := cFile
         oMLE:setData( MemoRead( cFile ) )
         SetAppWindow():setTitle( cFile )
      ENDIF
   RETURN

   // *************************************
   // Write text buffer of the MLE to file
   //
   PROCEDURE WriteFile( cFile, oMLE )

      IF ! Empty( cFile )
         oMLE:cargo := cFile
         MemoWrit( cFile, oMLE:getData() )
         SetAppWindow():setTitle( cFile )
      ENDIF

   RETURN
```

DBF_INIEDIT.PRG

```
/////////////////////////////////////////////////////////////////////////
//                                                                     //
//                                                                     //
//  DBF_INIEDIT.PRG                                                    //
//                                                                     //
//   23-10-2023                                                        //
//                                                                     //
//   www.cobasystems.com --- COBA Systems --- Slobodan Stanojević Coba  //
//   Open Source Project BAST Business Account Software Technology     //
//   www.Alaska-Software.com --- Alaska Xbase++ version 2.0.1503       //
//   www.Donnay-software.com --- eXpress++      version 2.0.268        //
//   Sergej Spirin --- FastReport for Xbase++   version 27.03.2015     //
//                                                                     //
//   Database Server PostgreSQL                 version 9.4.4.         //
//                                                                     //
//                                                                     //
/////////////////////////////////////////////////////////////////////////


* COMMON FUNCTION LOCATION:
* GDE_EXE()             // BAZNE.DLL
* FONT_CODEPAGE()       // BAZNE.DLL
* C__PROCITAJ()         // BAZNE.DLL
* C__PORUKA()           // BAZNE.DLL
* C__SIFRA()            // BAZNEX.DLL

* APP FUNCTION IN PRG:
* FUNCTION DBF_INIEDIT()
* FUNCTION ini___read()
* FUNCTION ini___default(nn)
* FUNCTION ini___write()
* STATIC FUNCTION xpassword()

#include "Appevent.ch"
#include "Xbp.ch"
#include "common.ch"
#INCLUDE "dcdialog.ch"


************************************************************************
FUNCTION DBF_INIEDIT()
************************************************************************
LOCAL GetList := {}, oDlg
LOCAL xx
PRIVATE xpassword := .F.

SET CHARSET TO ANSI

co0 := GraMakeRGBColor({100,30,50}) // CRNO VINO
co1 := GRA_CLR_WHITE
co2 := GRA_CLR_DARKGREEN
co3 := GRA_CLR_BLACK
co4 := GRA_CLR_YELLOW
of1  := Font_codepage(14,"Archivo Narrow",238,.t.)
of2  := Font_codepage(12,"Archivo Narrow",238,.f.)
```

```
of22  := Font_codepage(12,"Archivo Narrow",238,.t.)
of3  := Font_codepage(11,"Archivo Narrow",238,.t.)
of4  := Font_codepage(11,"Consolas",238,.t.)
aCUR := {"user32.dll",114}
oCUR := {"user32.dll",112}

*****************
 ini___read()
*****************
* formirane su:
PUBLIC ;
work_database,; //:= "POSTGRESQL" ,; // or "ORACLE" for which database is the sql script
created
work_user    ,; //:= "postgres"   ,; // username
work_pass    ,; //:= "coba"       ,; // password
work_data    ,; //:= "YES DATA"   ,; // "YES DATA" table with data, "NO DATA" table without
data
work_db      ,; //:= "test"       ,; // naziv baze podataka - database name
work_sema       //:= "public"        // naziv scheme iz baze podataka - scheme name


xx:=0

  @ xx,1  DCSAY "DATABASE PARAMETERS" FONT of22 SAYSIZE 0

xx:=xx+1+1
  work_database := PADR(work_database,36)
  @ xx,1  DCSAY "Database Server (POSTGRESQL or ORACLE)" FONT of2 SAYSIZE 0
xx:=xx+1
  @ xx,1  DCGET work_database FONT of4 WHEN {|| .F. }

xx:=xx+1
  work_user := PADR(work_user,36)
  @ xx,1  DCSAY "User name (default=postgres)" FONT of2 SAYSIZE 0
xx:=xx+1
  @ xx,1  DCGET work_user FONT of4 WHEN {|| xpassword }

xx:=xx+1
  work_pass := PADR(work_pass,36)
  @ xx,1  DCSAY "Password (default=postgres)" FONT of2 SAYSIZE 0
xx:=xx+1
  @ xx,1  DCGET work_pass FONT of4 WHEN {|| xpassword }

xx:=xx+1
  work_data := PADR(work_data,36)
  @ xx,1  DCSAY "Filling SQL table with data from DBF file" FONT of2 SAYSIZE 0
xx:=xx+1
  @ xx,1  DCGET work_data FONT of4 WHEN {|| .F. } // xpassword }


xx:=xx+1
  work_db := PADR(work_db,36)
  @ xx,1  DCSAY "Database (default=postgres)" FONT of2 SAYSIZE 0
xx:=xx+1
  @ xx,1  DCGET work_db FONT of4 WHEN {|| xpassword }


xx:=xx+1
  work_sema := PADR(work_sema,36)
  @ xx,1  DCSAY "Scheme (default=public)" FONT of2 SAYSIZE 0
```

```
xx:=xx+1
  @ xx,1  DCGET work_sema FONT of4 WHEN {|| xpassword }


//----------------------------------------------------------------------------

xx:=xx+2
  @ xx,01 ;
  DCPUSHBUTTONXP CAPTION "Unlock;for changes" ;
         SIZE 14,3 ACTION {||xpassword(),DC_getrefresh(GetList)} ;
         CURSOR aCUR OUTLINE COLOR co1,co2 CLICKCOLOR co3,co4 FONT of22

  @ xx,14 ;
  DCPUSHBUTTONXP CAPTION "OK" ;
         SIZE 10,3 ACTION {||DC_ReadGuiEvent(DCGUI_EXIT_OK,GetList)} ;
         CURSOR aCUR OUTLINE COLOR co1,co2 CLICKCOLOR co3,co4 FONT of22

  @ xx,14+10 ;
  DCPUSHBUTTONXP CAPTION "Help" ACCELKEY xbeK_F1 ;
         SIZE 10,3 ACTION {||;
         c__procitaj(;
         "Treba upisati sve zadate parametre."  +chr(59)+;
         "Program neke od njih ugrađuje u SQL"  +chr(59)+;
         "i TXT script fajlove, koje pravi iz"  +chr(59)+;
         "preuzetog DBF fajla."                 +chr(59)+;
         " "                                    +chr(59)+;
         "Ovi SQL-TXT script fajlovi koriste "  +chr(59)+;
         "se iz posebne EXE aplikacije kojom "  +chr(59)+;
         "se kreira i puni podacima PostgreSQL" +chr(59)+;
         "tabla koja je SQL klon DBF fajlA"     +chr(59)+;
         " "                                           +chr(59)+;
         "All default parameters should be entered."   +chr(59)+;
         "Program embeds some of them in SQL"          +chr(59)+;
         "and TXT script files, which it creates from" +chr(59)+;
         "downloaded DBF file."                        +chr(59)+;
         " "                                           +chr(59)+;
         "These SQL-TXT script files use "             +chr(59)+;
         "from a special EXE application that "        +chr(59)+;
         "created and loaded with PostgreSQL data"     +chr(59)+;
         "table which is SQL clone DBF fileA"          +chr(59)+;
         " ","DATABASE PARAMETERS")} ;
         CURSOR aCUR OUTLINE COLOR co1,co2 CLICKCOLOR co3,co4 FONT of22

  @ xx,14+10+10 ;
  DCPUSHBUTTONXP CAPTION "Exit" ACCELKEY xbeK_ESC;
         SIZE 10,3 ACTION {||DC_ReadGuiEvent(DCGUI_EXIT_OK,GetList)};
         CURSOR aCUR OUTLINE COLOR co1,co2 CLICKCOLOR co3,co4 FONT of22
//----------------------------------------------------------------------------

 DCGETOPTIONS NOMAXBUTTON NOMINBUTTON noescapekey ;
 COLORGETS { { GRA_CLR_WHITE, GraMakeRGBColor({208,82,92}) },{ GRA_CLR_DARKRED, GRA_CLR_WHITE }
}

 DCREAD GUI OPTIONS GetOptions ENTEREXIT PARENT @oDlg MODAL ;
 FIT TITLE "DATABASE PARAMETERS"

     ******************
       ini___write()    // here
     ******************
     c__poruka("UPISANO-WRITTEN","OK",,"G3")  // bazne.dll
```

```
RETURN(.T.)

****************************
 STATIC FUNCTION xpassword()
****************************
// xpassword := .F.
IF C__sifra(2,"(Password=22)")=.F.
   RETURN NIL
ENDIF
xpassword := .T.
c_poruka("UNLOCKED","OK")
RETURN NIL

***************************************************************************
FUNCTION ini___read()
***************************************************************************
PUBLIC ini := gde_exe()+"\"+"C-DBF2SQLFILE.INI"

PUBLIC ;
work_database,; //:= "POSTGRESQL" ,; // ili "ORACLE" za koju bazu podataka se pravi SQL script
work_user    ,; //:= "postgres"   ,; // username za bazu podataka
work_pass    ,; //:= "coba"       ,; // password za bazu podataka
work_data    ,; //:= "YES DATA"   ,; // work_data="YES DATA" prenesi tablu i podatke, "NO DATA"
prenesi smo tablu
work_db      ,; //:= "test"       ,; // naziv baze podataka
work_sema       //:= "public"        // naziv scheme iz baze podataka

 //--- učitaj podatak iz INI fajla
 //--- load data from the INI file
 work_user := INI_READ("C","POSTGRESQL","work_user",ini)

 IF EMPTY(ALLTRIM(work_user))
    // ako nema podatka - upiši u INI sve oznake u default vrednostima
    // if there is no data - write in INI all tags in default values
    ***********************
    ini___default()           // here
    ***********************
 ENDIF

 //--- ponovo učitaj sve podatke iz INI fajla
 //--- reload all the data from the INI file
   work_database := INI_READ("C","DATABASE"  ,"work_database",ini)
   work_dir      := INI_READ("C","DATABASE"  ,"work_dir" ,ini)
   work_user     := INI_READ("C","POSTGRESQL","work_user",ini)
   work_pass     := INI_READ("C","POSTGRESQL","work_pass",ini)
   work_data     := INI_READ("C","POSTGRESQL","work_data",ini)
   work_db       := INI_READ("C","POSTGRESQL","work_db"  ,ini)
   work_sema     := INI_READ("C","POSTGRESQL","work_sema",ini)

RETURN NIL


***************************************************************************
FUNCTION ini___default(nn)
***************************************************************************
* ako ne postoji INI fajl, biće napravljen i u njega će biti upisane ove
* default vrednosti. Ako pak postoji INI fajl u njega će biti upiane ove
* default vrednosti
*
```

```
* if there is no INI file, it will be created and these will be written
* into it default values. If there is an INI file, these will be absorbed
* into it default values

PUBLIC ini := gde_exe()+"\"+"C-DBF2SQLFILE.INI"

   * work_database = "ORACLE" ili "POSTGRESQL"
   INI_WRITE("C","DATABASE"  ,"work_database","POSTGRESQL",ini)
   INI_WRITE("C","DATABASE"  ,"work_dir","SQL",ini)  // "ALL","SQL"
   INI_WRITE("C","POSTGRESQL","work_user","postgres",ini)
   INI_WRITE("C","POSTGRESQL","work_pass","coba",ini)
   INI_WRITE("C","POSTGRESQL","work_data","YES DATA",ini) // "NO DATA"
   INI_WRITE("C","POSTGRESQL","work_db","test",ini)
   INI_WRITE("C","POSTGRESQL","work_sema","public",ini)

* work_dir se koristi u DBF_FOLDER.PRG i ako je: work_dir="SQL"
* tada se file dialog za izbor DBF fajla za translaciju u SQL skript
* stalno otvara u subfolderu \SQL foldera exe aplikacije
* U protivnom otvara se zadnji folder iz koga je izabran neki DBF
* Ovo se može podesiti samo ručnim upisom u INI fajl
*
* work_dir is used in DBF_FOLDER.PRG and if: work_dir="SQL"
* then the file dialog for selecting the DBF file for translation into
* SQL script constantly opens in the \SQL subfolder of the exe application
* folder Otherwise, the last folder from which a DBF was selected is opened
* This can only be set manually in the INI file

RETURN NIL


***************************************************************************
FUNCTION ini___write()
***************************************************************************
PUBLIC ini := gde_exe()+"\"+"C-DBF2SQLFILE.INI"

  //--- upiši podatke u INI fajl
  //--- write the data in the INI file
   INI_WRITE("C","DATABASE"  ,"work_database",work_database,ini)
   INI_WRITE("C","DATABASE"  ,"work_dir","SQL",ini) // "ALL","SQL"
   INI_WRITE("C","POSTGRESQL","work_user",work_user,ini)
   INI_WRITE("C","POSTGRESQL","work_pass",work_pass,ini)
   INI_WRITE("C","POSTGRESQL","work_data",work_data,ini)
   INI_WRITE("C","POSTGRESQL","work_db"  ,work_db,ini)
   INI_WRITE("C","POSTGRESQL","work_sema",work_sema,ini)

RETURN NIL
```

DBF_TO_POSTGRESQL_SQLSCRIPT.PRG

```
//////////////////////////////////////////////////////////////////////////
//                                                                      //
//                                                                      //
//   DBF_TO_POSTGRESQL_SQLSCRIPT.PRG
//                                                                      //
//   23-10-2023                                                         //
//                                                                      //
//   www.cobasystems.com --- COBA Systems --- Slobodan Stanojević Coba  //
//   Open Source Project BAST Business Account Software Technology      //
//   www.Alaska-Software.com --- Alaska Xbase++ version 2.0.1503        //
//   www.Donnay-software.com --- eXpress++      version 2.0.268         //
//   SAP --- Advantage Database Server ADS      version 10.10.0.49      //
//   Sergej Spirin --- FastReport for Xbase++   version 27.03.2015      //
//                                                                      //
//                                                                      //
//////////////////////////////////////////////////////////////////////////


* COMMON FUNCTION LOCATION:
* GDE_EXE()     // BAZNE.DLL
* C_GRESKA()    // BAZNE.DLL
* C__NEDA()     // BAZNE.DLL
* EC_P()        // BAZNE.DLL
* EC_K()        // BAZNE.DLL
* PROGRES1()    // BAZNE.DLL
* PROGRES2()    // BAZNE.DLL
* PROGRES3()    // BAZNE.DLL


* APP FUNCTION IN PRG:
* FUNCTION DBF_TO_POSTGRESQL_SQLSCRIPT()
* FUNCTION PODACI_FROM_DBF_TO_SQL()
* FUNCTION DBF_polja_Xbase()
*
* FUNCTION __ORACLE__()                         // not used in this application
* FUNCTION test__oracle__dbf_file_to_sql_table() // not used in this application





* ========================================================================= start
* ŠTA RADI OVAJ PROGRAM:
* =========================================================================
* OVAJ PROGRAM ČITA DbaseIII DBF/DBT FAJL (TABELU) I GENERIŠE SQL FAJL
*      za ORACLE       (sa DBF podacima ili bez DBF podataka)
*      za PostgreSQL   (sa DBF podacima ili bez DBF podataka)
*
* SQL fajl za ORACLE
*-------------------
* ČIJIM ĆE STARTOM IZVRŠENIM U ORACLE BAZI PODATAKA (SQLplus/SQLdeveloper)
* BITI IZVRŠEN PRENOS TE DbaseIII DBF TABELE U ISTOIMENU ORACLE 11g TABELU
* (Biće preneta struktura tabele - malo modifikovana - i podaci iz tabele)
*
```

```
* SQL fajl za PostgreSQL
*-----------------------
* ČIJIM ĆE STARTOM IZVRŠENIM U PostgreSQL BAZI PODATAKA (PL/PgSQL)
* BITI IZVRŠEN PRENOS TE DbaseIII DBF TABELE U ISTOIMENU PostgreSQL TABELU
* (Biće preneta struktura tabele - malo modifikovana - i podaci iz tabele)
*
*
* ORACLE ili PostgreSQL
*----------------------
* Za koju bazu podataka će ovaj program praviti SQL script odlučuje se
* učitavanjem parametra iz C-DBF2SQLFILE.INI putem funkcije:
*    sql_database_parameters() // DBF-TO-POSTGRESQL.PRG
*    PUBLIC work_database,;
*           work_user, work_pass, work_data, work_db, work_sema
*


* ======================================================================== start
* WHAT THIS PROGRAM DOES:
* ========================================================================
* THIS PROGRAM READS A DbaseIII DBF/DBT FILE (TABLE) AND GENERATES A SQL FILE
* for ORACLE (with or without DBF data)
* for PostgreSQL (with or without DBF data)
*
* SQL file for ORACLE
*-------------------
* WHICH WILL BE STARTED IN THE ORACLE DATABASE (SQLplus/SQLdeveloper)
* THE TRANSFER OF THAT DbaseIII DBF TABLE TO THE ORACLE 11g TABLE OF THE SAME NAME BE PERFORMED
* (The structure of the table will be transferred - slightly modified - and data from the
table)
*
* SQL file for PostgreSQL
*---------------------
* WHICH WILL BE STARTED IN THE PostgreSQL DATABASE (PL/PgSQL)
* THE TRANSFER OF THAT DbaseIII DBF TABLE TO THE SAME NAMED PostgreSQL TABLE BE PERFORMED
* (The structure of the table will be transferred - slightly modified - and data from the
table)
*
*
* ORACLE or PostgreSQL
*--------------------
* It is decided for which database this program will create the SQL script
* by loading the parameter from C-DBF2SQLFILE.INI via the function:
* sql_database_parameters() // DBF-TO-POSTGRESQL.PRG
* PUBLIC work_database,;
* work_user, work_pass, work_data, work_db, work_sema
*


* ini := gde_exe()+"\"+"C-DBF2SQLFILE.INI"
* Po defaultu:
* work_database := "POSTGRESQL" ,; // or "ORACLE"
* work_user     := "postgres"   ,; // username
* work_pass     := "coba"       ,; // password
* work_data     := "YES DATA"   ,; // work_data="YES DATA"
* work_db       := "test"       ,; // database name
* work_sema     := "public"        // scheme name
*
* IF work_database == "ORACLE"
*
```

```
*   work_user     := INI_READ("C","ORACLE","work_user",ini)
*   work_pass     := INI_READ("C","ORACLE","work_pass",ini)
*   work_data     := INI_READ("C","ORACLE","work_data",ini)
*   work_db       := INI_READ("C","ORACLE","work_db"  ,ini)
*   work_sema     := INI_READ("C","ORACLE","work_sema",ini)
*
* ELSE // work_database == "POSTGRESQL"  // po defaultu
*
*   work_user     := INI_READ("C","POSTGRESQL","work_user",ini)
*   work_pass     := INI_READ("C","POSTGRESQL","work_pass",ini)
*   work_data     := INI_READ("C","POSTGRESQL","work_data",ini)
*   work_db       := INI_READ("C","POSTGRESQL","work_db"  ,ini)
*   work_sema     := INI_READ("C","POSTGRESQL","work_sema",ini)
*
* ENDIF
* =========================================================================
*
* Ako se izabere bilo koji DBF/DBT fajl, iz liste mogućih DBF/DBT fajlova
* u ovom programu ili se prosledi ovoj funkciji pun path do DBF/DBT fajla
* kao parametar, njegova struktura će biti prevedena u strukturu Oracle
* tabele ili PostgreSQL tabele i upisana u SQL script fajl koji će poslužiti
* za kreiranje iste takve Oracle tabele i iste takve PostgreSQL tabele za
* Oracle Databaze Usera ili PostgreSQL usera prema parametrima:
*   PUBLIC work_user, work_pass, work_data, work_db, work_sema
*
*
* If any DBF/DBT file is selected, from the list of possible DBF/DBT files
* in this program or the full path to the DBF/DBT file is passed to this function
* as a parameter, its structure will be translated into an Oracle structure
* tables or PostgreSQL tables and written into the SQL script file that will serve
* to create the same such Oracle table and the same such PostgreSQL table for
* Oracle Database User or PostgreSQL user according to parameters:
* PUBLIC work_user, work_pass, work_data, work_db, work_sema
*
* FILL DATABASE
* -------------
* Ako je parametar work_data = "YES DATA"
* tada će svi podaci iz DBF/DBT fajla biti prepisani u SQL script fajl,
* pa će posle kreiranja Oracle ili PostgreSQL tabele ista tabela biti
* napunjena tim podacima. Ovim se vrši potpun prenos podataka iz DBF/DBT
* fajlova u Oracle i PostgreSQL tabele.
*
* FILL DATABASE
* -------------
* If parameter work_data = "YES DATA"
* then all the data from the DBF/DBT file will be copied into the SQL script file,
* so after creating an Oracle or PostgreSQL table, the same table will be
* loaded with that data. This completes the transfer of data from DBF/DBT
* files in Oracle and PostgreSQL tables.
*
* Pravila su sledeća:
*--------------------
* - NUMERIK
* - Svaki numerik N iz DBF prebacuje se u Oracle kao NUMBER(20,x) gde je
*   20 fiksiran ukupan broj mesta za numerik bez obzira koliki je on bio
*   u DBF fajlu, a x je broj decimalnih mesta preuzet iz DBF fajla.
*   Samo ako je decimalni x = 0, ako je u DBF fajlu to bio integer, tada
*   se u Oracle upisuje kao NUMBER(10) odnosno kao integer fiksiran na 10
*   Ovo su obično u DBF fajlu bili brojači slogova ili ID šifre artikala.
*   (Oracle databaze polje NUMBER ima najviše 38 mesta za cifre).
```

```
*
* - STRING
* - Svaki string C iz DBF prebacuje se u Oracle kao NVARCHAR2(x) gde je x
*   najveća moguća dužina promenljivog unicode stringa u karakterima CHAR,
*   preuzeta iz DBF. Ako je x=50 a upisani string u polje je 20, biće u
*   databazi zauzeto samo 20 CHAR, a ne 50 CHAR
*
* - BOOLEAN
* - Boolean polje L iz DBF prebacuje se u Oracle kao NCHAR(1) odnosno kao
*   unicode string fiksne dužine 1 CHAR i vrednost mu može biti samo veliko
*   slovo T (true) ili F (false)
*
* - MEMO
* - Memo polje M iz DBF prebacuje se u Oracle kao NVARCHAR2(600) gde je
*   600 najveća moguća dužina stringa u unicode CHAR karakterima.
*
* - DATE
* - Date polje D prebacuje se kao string "31.12.2016" u Oracle kao DATE
*
*
* The rules are as follows:
*--------------------------
* - NUMERIC
* - Each numeric N from DBF is passed to Oracle as NUMBER(20,x) where
* 20 fixed total number of places for the number regardless of how big it was
* in the DBF file, and x is the number of decimal places taken from the DBF file.
* Only if the decimal x = 0, if in the DBF file it was an integer, then
* is entered in Oracle as NUMBER(10), i.e. as an integer fixed at 10
* These were usually syllable counters or item code IDs in the DBF file.
* (Oracle database field NUMBER has a maximum of 38 places for digits).
*
* - STRING
* - Each string C from DBF is passed to Oracle as NVARCHAR2(x) where x is
* maximum possible length of variable unicode string in CHAR characters,
* taken from DBF. If x=50 and the string entered in the field is 20, it will be u
* database occupied only 20 CHAR, not 50 CHAR
*
* - BOOLEAN
* - Boolean field L from DBF is transferred to Oracle as NCHAR(1) or as
* unicode string of fixed length 1 CHAR and its value can only be large
* letter T (true) or F (false)
*
* - MEMO
* - Memo field M from DBF is passed to Oracle as NVARCHAR2(600) where
* 600 maximum possible string length in unicode CHAR characters.
*
* - DATES
* - Date field D is passed as string "12/31/2016" to Oracle as DATE
*
* ========================================================================
* ŠTA RADI OVAJ PROGRAM:
* WHAT THIS PROGRAM DOES:
* ======================================================================== end
```

```
#pragma Library( "XppUI2.LIB" )
#pragma Library( "Adac20b.lib" )

#include "Appevent.ch"
#include "Common.ch"
#include "Xbp.ch"
#include "Directry.ch"  // za funkciju directory()
#include "Xbtsys.ch"

**************************************************************************
FUNCTION DBF_TO_POSTGRESQL_SQLSCRIPT(_cDBFpath)
**************************************************************************
* cDBFpath = full path to dbf file, example:  "C:\DATABASE\customer.dbf"


LOCAL radno := SELECT()

LOCAL aField, aType, aLen, aDec, user, password, data

PRIVATE cDBFpath := _cDBFpath, _tdbf_, _tsql_, cTXTpath, cSQLpath


if right( lower(cDBFpath), 4 ) == ".dbf"
   // OK
else
   c_greska("Dozvoljen je rad samo sa DBF fajlovima"+chr(13)+;
            "Allowed to work only with DBF files"   +chr(13)+;
            "","STOP")
   return nil
endif


//-------------------------------------------------------------  start
// DATABAZE USER I PASWORD
//-------------------------------------------------------------
 ************
 USER_PASS() // C-DBF2SQLFILE.PRG
 ************
 * PUBLIC work_database    // "ORACLE" or "POSTGRES"
 * PUBLIC work_data, data  // sql script sa podacima iz dbf fajla YES/NO
 * PUBLIC work_user, work_pass, work_db, work_sema, user, pass
//-------------------------------------------------------------
// DATABAZE USER I PASWORD
//-------------------------------------------------------------  end


//-------------------------------------------------------------  start
// PORUKA KORISNIKU message to user - IZBOR Choice YES-NO
//-------------------------------------------------------------

cDBFpath := lower(cDBFpath)            // path - dbf fajl
                                       // "C\DATABASE\customer.dbf"
 * --- dc_path()
 * nT := rat("\",cDBFpath)
 * _tdbf_ := alltrim(substr(cDBFpath,nT+1,100)) // dbf file name  "customer.dbf"
 * _tsql_ := strtran(_tdbf_,".dbf")             // sql table name "customer"
 * --- dc_path()
```

```
_tdbf_   := dc_path(cDBFpath,.T.)        // dbf file name   "customer.dbf"
_tsql_   := strtran(_tdbf_,".dbf")       // sql table name  "customer"
cSQLpath := gde_exe()+"\SQL\"+_tsql_+".sql"  // path - sql fajl "C\APP\customer.sql"
cTXTpath := gde_exe()+"\SQL\"+_tsql_+".txt"  // path - txt fajl "C\APP\customer.txt"


_she_  := work_sema                      // sql scheme name
_db_   := work_db                        // sql database name
_uid_  := work_user                      // user name
_pwd_  := work_pass                      // password

IF FILE(gde_exe()+"\SQL","D") = .F.
   dirmake( gde_exe()+"\SQL" )
ENDIF


cc:="Create SQL script for TABLE structure;      and plus all data (all DBF records)"
IF c__neda(;
         "Šta radi ovaj program:                           " +chr(59)+;
         "Iz ovde izabranog DBF fajla:"               +chr(59)+;
         "      "+cDBFpath                            +chr(59)+;
         "kreiraju se SQL script falovi: "            +chr(59)+;
         "      "+cSQLpath                            +chr(59)+;
         "      "+cTXTpath                            +chr(59)+;
         "koji služe za kreiranje postgreSQL table:"  +chr(59)+;
         "      "+_tsql_                              +chr(59)+;
         "koja odgovara izabranom DBF fajlu,"         +chr(59)+;
         "i popunjava se podacima iz tog DBF fajla."  +chr(59)+;
         "---------------------------------------------"  +chr(59)+;
         "Ovi SQL script fajlovi služi za ugradnju "     +chr(59)+;
         "u SQL izraz iz koga se kreira Pg SQL tabla."   +chr(59)+;
         "Kreira se PgSQL tabla strukture kao DBF tabla" +chr(59)+;
         "plus dodate Xbase UPSIZE-ISAM-PGDBE kolone,"   +chr(59)+;
         "zatim obavezni indeksi i forigen key polja,"   +chr(59)+;
         "po sistemu potrebnom za poslovne programe"     +chr(59)+;
         "COBA Systems pod postgreSQL PGDBE mašinom."    +chr(59)+;
         "---------------------------------------------"  +chr(59)+;
         "U SQL script fajl upisuje se pristup"          +chr(59)+;
         _db_+" bazi podataka (samo za oracle):"         +chr(59)+;
         "      Database  = "+_db_    +chr(59)+;
         "      Schema    = "+_she_   +chr(59)+;
         "      User      = "+_uid_   +chr(59)+;
         "      Pass      = "+_pwd_   +chr(59)+;
         "Create SQL script for TABLE structure and"    +chr(59)+;
         "plus all data from DBFfile (all DBF records)" +chr(59)+;
         " ",":: Generisanje SQL fajla iz DBF fajla",,"V","R") == .F.

   RETURN NIL
ENDIF

IF c__neda(;
         "What this program does:                          " +chr(59)+;
         "From DBF file selected here:"                +chr(59)+;
         "      "+cDBFpath                             +chr(59)+;
         "creating SQL script files: "                 +chr(59)+;
         "      "+cSQLpath                             +chr(59)+;
         "      "+cTXTpath                             +chr(59)+;
         "which are used to create a postgreSQL table:"+chr(59)+;
         "      "+_tsql_                               +chr(59)+;
         "corresponding to the selected DBF file, and" +chr(59)+;
         "is populated with data from that DBF file."  +chr(59)+;
         "-------------------------------------------------" +chr(59)+;
         "These SQL script files are used for installation" +chr(59)+;
         "in SQL statement from which the PgSQL table is"   +chr(59)+;
         "created."                                        +chr(59)+;
         "Creating PgSQL structure table as DBF table"      +chr(59)+;
         "plus added Xbase UPSIZE-ISAM-PGDBE columns,"      +chr(59)+;
         "then mandatory indexes and forigen key fields,"   +chr(59)+;
         "by system required for business programs"         +chr(59)+;
         "COBA Systems under the postgreSQL PGDBE engine."  +chr(59)+;
         "-------------------------------------------------" +chr(59)+;
         "In the SQL script file, access is entered"        +chr(59)+;
         _db_+" database (for oracle only):"               +chr(59)+;
         "     Database  = "+_db_    +chr(59)+;
         "     Schema    = "+_she_   +chr(59)+;
         "     User      = "+_uid_   +chr(59)+;
         "     Pass      = "+_pwd_   +chr(59)+;
         "Create SQL script for TABLE structure and"    +chr(59)+;
         "plus all data from DBFfile (all DBF records)" +chr(59)+;
         " ","::  Create SQL script files from DBF files",,"V","B") == .F.

   RETURN NIL

ENDIF

//------------------------------------------------------------
// PORUKA KORISNIKU message to user - IZBOR Choice YES-NO
//------------------------------------------------------------  end



* formira se upsize SQL tabela identično preslikana iz DBF tabele
* sa dodatim upsize poljima (kolonama) i bez indeksa

* an upsize SQL table identically copied from the DBF table is formed
* with added upsize fields (columns) and no index


//------------------------------------------------------------ start
// FORMIRAJ PATH (LOKACIJU I NAZIV) ZA DBF FAJL I ZA SQL FAJL
// FORM PATH (LOCATION AND NAME) FOR DBF FILE AND FOR SQL FILE
//------------------------------------------------------------
* L E G E N D
* _she_  := work_sema      // sql scheme name    public
* _db_   := work_database  // sql database name  test
* _uid_  := work_user      // database user name postgres
* _pwd_  := work_pass      // database password  postgres
* _tdbf_   := _tdbf_       // name - dbf fajl  kupci.dbf
* cDBFpath := cDBFpath     // path - dbf fajl  c:\database\kupci.dbf
* _tsql_   := _tsql_       // sql table name   kupci
* _tsql_   := _tsql_       // sql table name   kupci
* cTXTpath := cTXTpath     // path - txt fajl  c:\app\kupci.txt
* cSQLpath := cSQLpath     // path - sql fajl  c:\app\kupci.sql
//------------------------------------------------------------
// FORM PATH (LOCATION AND NAME) FOR DBF FILE AND FOR SQL FILE
//------------------------------------------------------------ end
```

```
//------------------------------------------------------------ start
// ČITANJE STRUKTURE DBF FAJLA RADI PREVOĐENJA U SQL TABELU
// READING STRUCTURE OF DBF FILE FOR TRANSLATION INTO SQL TABLE
//------------------------------------------------------------

*-------------------------------------------- // definiši error codeblock:
bErrorHandler :=ErrorBlock( {|e| Break(e)} )  // prekini kod dođe do greške
BEGIN SEQUENCE                                // Break() radi sa BEGIN SEQUENCE
*-------------------------------------------- // definiši error codeblock:

USE (cDBFpath) NEW SHARED

      aField := Array( FCount() )
      aType  := Array( FCount() )
      aLen   := Array( FCount() )
      aDec   := Array( FCount() )

      nFieldCount := AFields( aField, aType, aLen, aDec )
USE
*--------------------------------------------

RECOVER

*--------------------------------------------
   ch:=chr(13)
   ec_k() // bazne.dll
   confirmbox(,;
   "DBF FAJL NIJE MOGUĆE OTVORITI"            +ch+;
   "- Ili je DBF fajl oštećen"                +ch+;
   "- Ili DBF fajl zahteva i DBT/FPT fajl"    +ch+;
   "- Ili to nije DBF fajl, bez obzira na"    +ch+;
   "  to što ima extenziju DBF"               +ch+;
   " "                                        +ch+;
   "DBF FILE COULD NOT BE OPENED"             +ch+;
   "- Or DBF file is corrupted"               +ch+;
   "- Or DBF file also requires a DBT/FPT file" +ch+;
   "- Or it's not a DBF file, regardless"       +ch+;
   "  that has the extension DBF"               +ch+;
   "","STOP",;
   XBPMB_OK,XBPMB_CRITICAL,;
   XBPMB_SYSMODAL+XBPMB_MOVEABLE)
   RETURN NIL
*--------------------------------------------
END SEQUENCE
ErrorBlock( bErrorHandler ) // reset old codeblock


//------------------------------------------------------------
// READING STRUCTURE OF DBF FILE FOR TRANSLATION INTO SQL TABLE
//------------------------------------------------------------ end
```

```
// FIXED ON :
work_database := "POSTGRESQL"
// u ovom programu pravi se samo POSTGRESQL database
// only POSTGRESQL database is created in this program


IF work_database = "ORACLE"

//------------------------------------------------------------- start
// S Q L  S C R I P T  F O R  O R A C L E
//-------------------------------------------------------------
   __ORACLE__()  // ovde
//-------------------------------------------------------------
// S Q L  S C R I P T  F O R  O R A C L E
//------------------------------------------------------------- end

ELSE // IF work_database = "POSTGRESQL"

//------------------------------------------------------------- start
// S Q L  S C R I P T  F O R  P O S T G R E S Q L
//-------------------------------------------------------------

* GENERIŠI SQL SCRIPT - KAO ASCII TEXT FAJL SA NAZIVOM: cSQLpath
* GENERATE SQL SCRIPT - AS ASCII TEXT FILE WITH NAME: cSQLpath
* START

xx:=space(2) // left margin
ctn := "-- " + _tsql_+ ".TXT"  // 'KUP.TXT'


SET PRINTER TO (cTXTpath)
SET PRINT ON
SET CONSOLE OFF

xx  := space(2)

?  "--"
?  "-- START -- "+ctn
?  "--"
?  "-----------------------------------------------------------------"
?  "  "
?  "-- COBA Systems "+ dtoc(date())+" --- "+time()
?  "-- Open Source Projct: Alaska Xbase++ DBF To " + work_database
?  "--"
?  "-- Translate"
?  padr("-- ISAM Xbase DBF File ",23) +   " --> "+_tdbf_
?  "-- To"
?  padr("-- SQL database Table  ",23) +   " --> "+_tsql_
?  "--"
?  "-- SQL Script: "+cTXTpath
?  "-- Database  : "+work_db
?  "-- Schema    : "+work_sema
?  "-- User      : "+work_user
?  "-- Pass      : "+work_pass
?  "  "
?  "-----------------------------------------------------------------"
```

```
 *  LOCAL cDBFpath   // "C:\PRG\KUP.DBF"
 *  LOCAL _tdbf_     // "KUP.DBF"
 *  LOCAL _tsql_     // "KUP"


 ******************************************// ORACLE and PGSQL
   DBF_polja_Xbase(cDBFpath,_tdbf_,_tsql_) // ispis liste dbf polja - print list of dbf fields
 ******************************************// here

?  "-----------------------------------------------------------------"
? xx + " "
? xx + "-- "+"P O S T G R E S Q L"+" --> "+_tsql_
? xx + "--"
? xx + "--    Ovde je poseban kod za svaku od DBF to SQL tabli | Here is a separate code for
each DBF to SQL table"
? xx + "--    Koriste se varijable umesto naziva šeme i table  | Variables are used instead of
schema and table names"
? xx + "--    PRIVATE ctable := "+work_sema+"."+_tsql_
? xx + "--    PRIVATE ckey   := "+_tsql_ + "_pkey"
? xx + " "
?  "-----------------------------------------------------------------"
? xx + "-- Brisanje postojeće table | Deleting an existing table -- "
*** ? xx + "DROP TABLE IF EXISTS "+_tsql_+";"  // varijanta a --- show literal table name
    ? xx + "DROP TABLE IF EXISTS &ctable;"     // varijanta b --- shov variable for table name
? xx + " "


?  xx+"-- Kreiranje nove table | create new table ----------------------- START"


*** ?  xx+"CREATE TABLE "+_tsql_      // varijanta a
    ?  xx+"CREATE TABLE &ctable"      // varijanta b
?  xx+"(                                     "


nCharLen := 1
// broj kojim se množi dužina stringa u karakterima
// zbog povećanja dužine polja u Oracle tabeli u
// odnosu na dužinu u DBF tabeli
//---
// number by which to multiply the length of the string in characters
// due to increasing field length in Oracle table to relative to the
// length in the DBF table


//------------------------------------------------------------------------ start
//--- KREIRANJE STRUKTURE REDA TABELE - POLJA U SLOGU
//--- CREATION OF TABLE ROW STRUCTURE - FIELDS IN RECORD
//------------------------------------------------------------------------

/*
* PRIMER PL/SQL KOMANDE ZA CREATE TABLE
* EXAMPLE PL/SQL COMMAND FOR CREATE TABLE
CREATE TABLE DEMO -- napravi novu praznu tabelu DEMO
(      ID_     integer,   ili NOT NULL DEFAULT 0,     -- integer
      TEKST_ character(10),                          -- max 10 char
       BROJ_  numeric(20,2),                          -- decimal number
       DATUM_ date,                                   -- 2018-12-31
       FLAG_  boolean,   ili NOT NULL DEFAULT false, -- yes or no, true or false
      MEMO_  text,                                    -- max 4000 char
      BMP_   bytea                                    -- bitmap
      COUNT_ serial NOT NULL,                         -- autoincremental
);
-- COUNT_ ima unikatnu vrednost (ne mogu postojati dva ista COUNT_ broja)
-- COUNT_ has a unique value (no two COUNT_ numbers can be the same)
*/


FOR i = 1 TO nFieldCount // broj polja u slogu
                         // the number of fields in the record
                         // number of columns in a row

   cField := aField[i]
   cType  := "xxx"

   // string
   IF aType[i]$"C"
      nLen := (aLen[i] * nCharLen)
      nLen := INT(nLen)
      // za NVARCHAR2 ne treba povećavati polje u karakterima
      // for NVARCHAR2 the field should not be increased in characters
      // nCharLen = 1
      cType := "character("+var2char(nLen)+")"
      // NVARCHAR2 je dužina polja u unicode karakterima
      // NVARCHAR2 is the length of the field in unicode characters
   ENDIF

   // date
   IF aType[i]$"D"
      cType := "date"
   ENDIF

   // numeric
   IF aType[i]$"N"
      *---
        cType := "numeric("+var2char(aLen[i])+","+var2char(aDec[i])+")"
      * uzmi da svaki numerik bude NUMBER(20,2)  |  take each number to be NUMBER(20,2)
      * bez obzira koliko je aLen[i] u DBF polju |  no matter how many aLen[i] are in the DBF
field
      * cType := "numeric("+var2char(20)+","+var2char(aDec[i])+")"
      *---
      * samo ako je numerik bio integer | only if the numeric was an integer
      * (koristi se za brojače i za šifre) | (used for counters and for codes)
        IF aDec[i]=0
           cType := "integer"
        ENDIF
      *---
   ENDIF


   // boolean
   IF aType[i]$"L"
   // for DBF boolean .T. or .F. in postgresql table
   // is used
      cType := "boolean"
   ENDIF


   // memo
   IF aType[i]$"M"
   // for memo field:
        cType := "text"
   ENDIF



UPSIZE := .T.

IF UPSIZE == .F.

 *-------------------- ako nema UPSIZE dodatka za postgreSQL tablu -------------
 *                      if there is no UPSIZE plugin for the postgreSQL table

   IF i < nFieldCount

       // sva polja osim zadnjeg polja završavaju sa ","
       // all fields except last field end with ","
    ? xx+PADR(cField,20 ) + PADR(cType,20 ) + "," // zarez

   ELSE

       // ako je ovo zadnje polje u slogu završava sa ");"
       // ili sa ")" ako posle njega ide 'WITH'.
       // 'WITH' završava sa ");"

       // if this is the last field in the record ends with ");"
       // or with ")" if it is followed by 'WITH'.
       // 'WITH' ends with ");"

    ? xx+PADR(cField,20 ) + PADR(cType,20 )
    *** ?  xx+");"
    ? xx+")"
    ? xx+"WITH (OIDS=FALSE,autovacuum_enabled=true);"


   ENDIF
 *                      if there is no UPSIZE plugin for the postgreSQL table
 *-------------------- ako nema UPSIZE dodatka za postgreSQL tablu -------------

ELSE  // UPSIZE == .T.

 *-------------------- ako ima UPSIZE dodatka za postgreSQL tablu --------------
 *                      if there is a UPSIZE plugin for the postgreSQL table

    ? xx+PADR(cField,20 ) + PADR(cType,20 ) + "," // zarez

 *                      if there is a UPSIZE plugin for the postgreSQL table
 *-------------------- ako ima UPSIZE dodatka za postgreSQL tablu --------------

ENDIF // IF UPSIZE== .T.

NEXT i

IF UPSIZE == .T.
*-------------------- ako ima UPSIZE dodatka za postgreSQL tablu --------------
*                      if there is a UPSIZE plugin for the postgreSQL table

? xx+"   __deleted boolean NOT NULL DEFAULT false,"
? xx+"   __record serial NOT NULL,                "
? xx+"   __rowversion integer NOT NULL DEFAULT 1, "
? xx+"   __keyversion integer NOT NULL DEFAULT 0, "
? xx+"   __lock_owner integer NOT NULL DEFAULT 0, "
? xx+"   CONSTRAINT &ckey PRIMARY KEY (__record)  "
? xx+")"
? xx+"WITH (OIDS=FALSE,autovacuum_enabled=true);"

*                      if there is a UPSIZE plugin for the postgreSQL table
*-------------------- ako ima UPSIZE dodatka za postgreSQL tablu --------------
ENDIF

//----------------------------------------------------------------------
//--- KREIRANJE STRUKTURE REDA TABELE - POLJA U SLOGU
//--- CREATION OF TABLE ROW STRUCTURE - FIELDS IN RECORD
//---------------------------------------------------------------------- end

?  xx+" "
?  xx+"-- Kreiranje nove table | create new table ----------------------- END"
?  xx+" "
?  xx+"-- Kod između crta: Kreiranje nove table START i END     |  Code between lines: Creating
a new table START and END "
?  xx+"-- Upisuje se (HARD CODED) u izvorni kod EXE aplikacije  |  It is written (HARD CODED)
into the source code of the EXE application"
?  xx+"-- in prg:        __tab__create__app.prg "
?  xx+"-- in function:   __tab__create__app()   "
?  xx+"  "
?  "-----------------------------------------------------------------"
?  "--"
?  "-- END -- "+ctn
?  "--"

SET PRINTER TO
SET PRINTER OFF
SET CONSOLE ON


* END
* GENERIŠI SQL SCRIPT - KAO ASCII TEXT FAJL SA NAZIVOM: cTXTpath
* GENERATE SQL SCRIPT - AS ASCII TEXT FILE WITH NAME: cSQLpath


ENDIF // IF work_database = "POSTGRESQL"


//-------------------------------------------------------------
// S Q L   S C R I P T   F O R   P O S T G R E S Q L
//------------------------------------------------------------- end

 // vidi SQL text string   | see SQL text string
 RUNSHELL( cTXTpath, "NOTEPAD.EXE")


 // napuni SQL tablu podacima iz DBF  |  fill the SQL table with data from the DBF
 ************************************************** // prenos slogova iz DBF u POSTGRESQL
 PODACI_FROM_DBF_TO_SQL(cDBFpath,_tsql_,aField,xx)  // here
 ************************************************** // transfer of records from DBF to ORACLE

RETURN(NIL)
```

```
***************************************************************************
FUNCTION PODACI_FROM_DBF_TO_SQL(cDBFpath,_tsql_,aField,xx)
***************************************************************************
* Punjenje SQL fajla podacima preuzetim iz DBF fajla
* Filling the SQL file with data taken from the DBF file


LOCAL radno:=SELECT()
LOCAL x := 0, i := 0, nField_Count
LOCAL cField, cData, cField_Name, cField_Type


//-------------------------------------------------------
//--- PREPIS PODATAKA IZ DBF FAJLA U SQL FAJL
//--- WRITE DATA FROM DBF FILE INTO SQL FILE
//-------------------------------------------------------
x := 0

USE (cDBFpath) NEW SHARED
IF NETERR()
   msgbox(_tdbf_,"No file available:")
   RETURN NIL
ENDIF

set date german
set century on
brojac:=reccount()
IF brojac==0
   USE
   SELECT(radno)
   RETURN NIL
ENDIF


nField_Count := FCount()

* For i=1 TO nField_Count
* cField_Name  := FieldName(i)
* NEXT i


*********************************************************************   POSTGRESQL - START
* START
* GENERIŠI SQL SCRIPT - KAO ASCII TEXT FAJL SA NAZIVOM: cSQLpath
* GENERATE SQL SCRIPT - AS ASCII TEXT FILE WITH NAME: cSQLpath
*********************************************************************

xx:=space(2) // left margin
ctn := "-- " + _tsql_+ ".SQL"

SET PRINTER TO (cSQLpath)
SET PRINT ON
SET CONSOLE OFF


?  "--"
?  "-- START -- "+ctn
?  "--"
?  "-----------------------------------------------------------------------"
?  "  "
?  "-- COBA Systems "+ dtoc(date())+" --- "+time()
?  "-- Open Source Projct: Alaska Xbase++ DBF To " + work_database
?  "--"
?  "-- DATA TRANSFER FROM:"
?  padr("-- ISAM Xbase DBF File ",23) +   " --> "+_tdbf_
?  "-- TO:"
?  padr("-- SQL database Table  ",23) +   " --> "+_tsql_
?  "--"
?  "-- SQL Script: "+cSQLpath
?  "-- Database   : "+work_db
?  "-- Schema     : "+work_sema
?  "-- User       : "+work_user
?  "-- Pass       : "+work_pass
?  "  "
?  "-----------------------------------------------------------------------"
? xx + "  "
? xx + "-- "+"P O S T G R E S Q L"+" --> "+_tsql_
? xx + "--"
? xx + "--    Ovde je poseban kod za svaku od DBF to SQL tabli | Here is a separate code for
each DBF to SQL table"
? xx + "--    Važno u ovoj verziji programa | Important in this version of the program: -------
------------------"
? xx + "--    Ne Koriste se varijable umesto naziva šeme i table | No Variables are used
instead of schema and board names:"
? xx + "--    PRIVATE ctable := "+work_sema+"."+_tsql_
? xx + "--    PRIVATE ckey   := "+_tsql_ + "_pkey"
? xx + "--    Moraju se koristiti literal (stvarni, pravi) nazivi | Literal (real, proper)
names must be used"
? xx + "  "
?  "-----------------------------------------------------------------------"



/*

* insert into table: varijanta A
*----------------------------------
* može se prvo navesti lista naziva svih kolona odnosno svih polja,
* a zatim po istom redosledu vrednosti koje pripadaju tim kolonama,
* ali to se radi samo kada se ne insertuju sve kolone iz tabele:
* ---
* a list of the names of all columns, i.e. all fields, can be specified first,
* and then in the same order of values belonging to those columns,
* but this is done only when not all columns from the table are inserted:
*
* INSERT INTO tablename
* (field1, field2, field3)
* VALUES
* (field1_values11, field2_values21, field3_values31),  // prvi slog-record
* (field1_values12, field2_values22, field3_values32),  // drugi slog-record
* (field1_values12, field2_values22, field3_values32);  // treći slog-record

* formiranje SQL stringa
*
? xx+"INSERT INTO "+_tsql_
? xx+"("
FOR i=1 TO nFieldCount
```

```
  IF i < nFieldCount
     ? xx+aField[i] + ","
  ELSE
     ? xx+aField[i] + " "
  ENDIF

NEXT i
? xx+")"

*/





/*
* insert into table: varijanta B
* ---------------------------------
* ako je insert svih kolona iz tabele nema potrebe navoditi
* listu naziva kolona (polja) već se navode samo vrednosti koje
* pripadaju tim poljima, po istom redosledu po kome su polja
*
* if it is an insert of all columns from the table, there is no need to specify
* the list of column names (fields) only lists the values that
* belong to those fields, in the same order as the fields

    INSERT INTO tablename
    VALUES
    (field1_values11, field2_values21, field3_values31),  // prvi slog-record
    (field1_values12, field2_values22, field3_values32),  // drugi slog-record
    (field1_values12, field2_values22, field3_values32);  // treći slog-record


-- primer za način upisa podataka |  an example of how to enter data
-- Dodaj jedan red u DEMO tabelu  |  Add one row to the DEMO table

INSERT INTO DEMO VALUES
( 999999      , -- integer
 'Text=ŠĐČĆŽ' , -- string se šalje sa navodnicima
 123456.78    , -- numerik se šalje bez navodnika
 '31-DEC-2016', -- datum se ovde šalje u polje DATE kao string
 'T'          , -- Logical Boolean se šalje kao string T=TRUE, F=FALSE
 -- dugački tekst | long text
 'Register here |long text| which may contain
  Serbian Latin and Serbian Cyrilic fonts.
  Serbian Latin:..... ŠĐČĆŽ šđčćž
  Serbian Cyrilic:... QWŠĐČĆŽX qwšđčćžx
  Fonts should be Unicode UTF-8.
  Convert single quotes in the text to a character "|"'
  -- dugački tekst | long text
);
*/



// navesti listu kolona-polja u koja se insertuju podaci
// specify the list of column-fields into which data is inserted
PRIVATE ctable := work_sema+"."+_tsql_

    ? "INSERT INTO "+ctable    // varijanta a
*** ? "INSERT INTO &cTable"    // varijanta b
```

```
// lista kolona-polja | list of column-fields
? "("
FOR i = 1 TO LEN(aField)
    IF i < nField_Count
       ? aField[i] + ","
    ELSE
       ? aField[i] + " "
    ENDIF
NEXT i
? ")"
? xx+"VALUES "



PROGRES1(RECCOUNT())  // bazne.dll

GO TOP
DO WHILE .NOT. EOF()
x := x + 1


*=============
* VARIJANTA A
*=============
* vidi kod na kraju pod naslovom VARIJANTA A  |  see the code at the end under the heading
VARIANT A


*=============
* VARIJANTA B
*=============
* ovde se koristi VARIJANTA B  |  VARIANT B is used here



? xx+"("

FOR i = 1 TO nField_Count



    cField_Name  := FieldName(i)  // dobijanje naziv polja | getting the field name
    cField       := &cField_Name  // dobijanje sadržaja polja | getting the contents of field


    cField_Type  := TYPE(cField_Name)


  //-------------- DATE

    IF cField_Type$"D"     // ako je datum |  if date
       IF EMPTY(cField)    // ako je prazan datum | if empty date
          cField := date() // datum ne sme biti prazan | date must not be empty
       ENDIF
       // pretvori ga u string radi uštampavanja u SQL fajl (TXT fajl)
       // convert it to string for printing to SQL file (TXT file)
       cField := DTOC(cField)
       cData := "'" + cField +"'"
       // u SQL datum je pod navodnicima kao string '31.12.2016'
```

```
      // in SQL the date is quoted as the string '12/31/2016'
   ENDIF

 //-------------- NUMERIC

   IF cField_Type$"N" // ako je numerik | if numeric
      // pretvori ga u string radi uštampavanja u SQL fajl (TXT fajl)
      // convert it to string for printing to SQL file (TXT file)
      cField := Var2char(cField)
      cData := cField  // u SQL numerik-string je bez navodnika
   ENDIF

 //-------------- LOGIC (BOOLEAN)

   IF cField_Type$"L" // ako je boolean | if boolean
      // pretvori ga u string radi uštampavanja u SQL fajl (TXT fajl)
      // convert it to string for printing to SQL file (TXT file)
      cField := Var2char(cField)
      // String je ".F." ili ".T." ako je polje prazno string je ".F."
      // PostgreSQL prima u tabelu samo "False","True"
      // ---
      // String is ".F." or ".T." if the field is empty the string is ".F."
      // PostgreSQL accepts only "False","True" or "F","T" in the table
      IF cField$".T."
         cField := "T"   // TRUE
      ELSE
         cField := "F"   // FALSE
      ENDIF
      cData := "'" + cField +"'"
      // u SQL BOOLEAN je string pod navodnicima 'T' ili 'F'
      // in SQL BOOLEAN is a quoted string 'T' or 'F'
   ENDIF

 //-------------- MEMO

   IF cField_Type$"M" // ako je memo | if memo
      IF EMPTY(cField)
         cField := ""
      ELSE
      // pretvori jednostruke navodnike ' u dvostruke "
      // jer se ' koristi kod uštampavanja u SQL fajl (TXT fajl)
      // ---
      // convert single quotes ' to double "
      // because ' is used when printing to a SQL file (TXT file)
         cField := StrTran(cField,"'","|")
      ENDIF
      cData := "'" + cField +"'"
      // u SQL fajlu MEMO je string pod navodnicima 'tekst...'
      // in the SQL file MEMO is the quoted string 'text...'
   ENDIF


   IF i < nField_Count

      IF cField_Type$"N" // numerik nema navodnike | if it is numeric, there are no quotes
         ? xx+cField+"," // ima zarez | has a comma
      ELSE
         ? xx+"'" + cField +"'"+"," // ima zarez | has a comma
      ENDIF
```

```
    ELSE

       ? xx+"'" + cField +"'"+" " // nema zareza | no comma

    ENDIF

NEXT i

if brojac=x
   ? xx+");"
else
   ? xx+"),"
endif

?  "-----------------------------------------------------------------"

* //----------- TEST exit on record 100
* if x=100
*   exit
* endif
* //----------- TEST


PROGRES2()  // bazne.dll

SKIP
ENDDO
USE
SELECT(radno)


? xx+"-- DATA TRANSFER END -- "
? xx+"-- Records = "+VAR2CHAR(x)+" --"
? "--"
? "-- END -- "+ctn
? "--"

SET PRINTER TO
SET PRINTER OFF
SET CONSOLE ON

//-----------------------------------------------------
//--- PREPIS PODATAKA IZ DBF FAJLA U SQL FAJL
//--- WRITE DATA FROM DBF FILE INTO SQL FILE
//-----------------------------------------------------

PROGRES3() // bazne.dll

*********************************************************************
* END
* GENERIŠI SQL SCRIPT - KAO ASCII TEXT FAJL SA NAZIVOM: cSQLpath
* GENERATE SQL SCRIPT - AS ASCII TEXT FILE WITH NAME: cSQLpath
*********************************************************************   POSTGRESQL - END

 ************************************
  RUNSHELL( cSQLpath, "NOTEPAD.EXE")
 ************************************

RETURN NIL
```

```
/*
************************************************************************ VARIJANTA A START
// VARIJANTA A

// za svaki slog-record navesti listu kolona-polja u koja se insertuju podaci
// for each slog-record specify a list of column-fields in which data is inserted

? "INSERT INTO "+_tsql_

// lista kolona-polja
? "("
FOR i = 1 TO LEN(aField)
    IF i < nField_Count
       ? aField[i] + ","
    ELSE
       ? aField[i] + " "
    ENDIF
NEXT i
? ")"

// za svaki slog-record navesti podatke koji idu u gornja polja po istom redosledu
// for each slog-record, specify the data that goes into the above fields in the
// same order

? xx+"VALUES "
? xx+"("

FOR i = 1 TO nField_Count

    cField_Name  := FieldName(i)    // dobijanje naziv polja
    cField       := &cField_Name    // dobijanje sadržaja polja

    cField_Type  := TYPE(cField_Name)

  //-------------- DATE

    IF cField_Type$"D"      // ako je datum
       IF EMPTY(cField)     // ako je prazan datum
          cField := date() // datum ne sme biti prazan
       ENDIF
       // pretvori ga u string radi uštampavanja u SQL fajl (TXT fajl)
       cField := DTOC(cField)
       cData := "'" + cField +"'"  // u SQL datum je pod navodnicima kao string '31.12.2016'
    ENDIF

  //-------------- NUMERIC

    IF cField_Type$"N" // ako je numerik
       // pretvori ga u string radi uštampavanja u SQL fajl (TXT fajl)
       cField := Var2char(cField)
       cData := cField  // u SQL numerik-string je bez navodnika
    ENDIF

  //-------------- LOGIC (BOOLEAN)

    IF cField_Type$"L" // ako je boolean
       // pretvori ga u string radi uštampavanja u SQL fajl (TXT fajl)
       cField := Var2char(cField)
       // String je ".F." ili ".T." ako je polje prazno string je ".F."
       // Oracle prima u tabelu samo 0,1 ili "N","Y" ili "False","True"
       IF cField$".T."
          cField := "T"   // TRUE
       ELSE
          cField := "F"   // FALSE
       ENDIF
       cData := "'" + cField +"'"  // u SQL BOOLEAN je string pod navodnicima 'T' ili 'F'
    ENDIF

  //-------------- MEMO

    IF cField_Type$"M" // ako je memo
       IF EMPTY(cField)
          cField := ""
       ELSE
       // pretvori jednostruke navodnike ' u dvostruke "
       // jer se ' koristi kod uštampavanja u SQL fajl (TXT fajl)
          cField := StrTran(cField,"'","|")
       // pretvori chr(13)+chr(10) u znak "|"
        * cField := StrTran(cField,chr(13)+chr(10),"|")
       ENDIF
       cData := "'" + cField +"'"  // u SQL MEMO je string pod navodnicima 'tekst...'
    ENDIF

    IF i < nField_Count

       IF cField_Type$"N" // ako je numerik nema navodnike
          ? xx+cField+"," // ima zareza
       ELSE
          ? xx+"'" + cField +"'"+"," // ima zareza
       ENDIF

    ELSE

       ? xx+"'" + cField +"'"+" " // nema zareza

    ENDIF

NEXT i

? xx+");"
**************************************************************************  VARIJANTA A END
*/
```

```
// ORACLE and PGSQL
***************************************************************************
FUNCTION DBF_polja_Xbase(cDBFpath,_tdbf_,_tsql_)
***************************************************************************
* štampani prikaz (lista) polja u DBF fajlu
* printed display (list) of fields in the DBF file

LOCAL aField, aType, aLen, aDec

 *  LOCAL cDBFpath     // "C:\PRG\KUP.DBF"
 *  LOCAL _tdbf_       // "KUP.DBF"
 *  LOCAL _tsql_       // "KUP"

   LOCAL i,x,B,DBalias
   LOCAL naziv,duzina,ostatak
   LOCAL brojac,duz,crta1,crta2


   USE (cDBFpath) NEW SHARED                 // use dbf
     B := FCOUNT()                           // number fields in record
     DECLARE fime[B],ftip[B],fduz[B],fdec[B] // create array
     AFIELDS(fime,ftip,fduz,fdec)            // fill array
   USE


  FOR i = 1 to B
    fime[i] := PADR(ALLTRIM(fime[i]),10)
  NEXT i

// print list DBF fields
xx := space(2)
? "--"
? "-- X B A S E  F I L E -- "+_tdbf_ // cDBFpath
? "--"

  crta1 = space(5)
  brojac := 0
  duz    := 0
  FOR i = 1 to B
      brojac := brojac+1
      duz    := duz+ fduz[i]
       ?  "-- "+ crta1 + str(brojac,3) + " "
       ?? fime[i] + " " + ftip[i] + " " + str(fduz[i],4,0) +" "+ str(fdec[i],1,0)+;
       " "+"--"
  NEXT i
? " "
RETURN NIL
```

```
                  *################################
                  *            ORACLE
                  *################################

*************************************************************************
FUNCTION __ORACLE__()
*************************************************************************
//-------------------------------------------------------------- start
// S Q L  S C R I P T  Z A  O R A C L E
//--------------------------------------------------------------

*********************************************************************  ORACLE - START
* GENERIŠI SQL SCRIPT - KAO ASCII TEXT FAJL SA NAZIVOM: cSQLpath
* START
*********************************************************************
xx:=space(2) // leva margina
ctn := "-- " + work_database +" "+ _tsql_+ ".SQL"
lCoba:=.T.


SET PRINTER TO (cSQLpath) // (cTXTpath)
SET PRINT ON
SET CONSOLE OFF

xx  := space(2)

?  "--"
?  "-- START"
?  ctn // "-- "+_tsql_+".SQL"
?  "--"
?  "-------------------------------------------------------------"
?  "-- COBA Systems "+ dtoc(date())+" --- "+time()
?  "-- Open Source Projct: Alaska Xbase++ DBF To "+work_database
?  "--"
?  "-- Translate"
?  padr("-- ISAM Xbase DBF File ",33)              +   " --> "+_tdbf_
?  "-- To"
?  padr("-- SQL database Table "+work_database,33) +   " --> "+_tsql_
?  "--"
?  "-- SQL Script: "+cSQLpath
?  "-- Database  : "+work_db
?  "-- Schema    : "+work_sema
?  "-- User      : "+work_user
?  "-- Pass      : "+work_pass
?  "-------------------------------------------------------------"

 ********************      // ORACLE i PGSQL
   DBF_polja_Xbase()       // ispis liste dbf polja // ovde
 ********************      // isti program se koristi za ORACLE i PGSQL

*********************************************************************
* KONEKTUJ SE NA DATABAZE SHEMU SA USER/PASSWORD ZA SHEMU (ZA USERA)
*********************************************************************
? xx + "--"
? xx + "-- "+work_database+" --> "+_tsql_
? xx + "--"
? xx + "-- Konekcija:"
? xx + "-- CONNECT user/password iz fajla:"
? xx + "-- DBFDOC.INI"
? xx + "--"
? xx + "CONNECT "+work_user+"/"+work_pass       // konekcija na databazu
? xx + "SPOOL "+_tsql_+".LST"               // log fajl
? xx + "SET LINESIZE 100"
? xx + "SET PAGESIZE 50"
? xx + " "
*********************************************************************
* OBRIŠI POSTOJEĆU TABELU SA ISTIM NAZIVOM _tsql_
*********************************************************************
? xx + "-- lokalizacija: format datuma i kodna strana = UTF --"
? xx + "ALTER SESSION SET NLS_TERRITORY = 'SERBIA AND MONTENEGRO';"
? xx + " "

? xx + "-- Brisanje postojeće table start -- "
? xx + "DROP TABLE "+_tsql_+";"
? xx + " "

* ?  xx + "-- Brisanje svih slogova iz table start --  "
* ?  xx + "DELETE FROM "+_tsql_+";"
* ?  xx + " "
*********************************************************************
* KREIRAJ NOVU PRAZNU TABELU _tsql_
*********************************************************************
?  xx+"-- Kreiranje nove table start --      "
?  xx+"CREATE TABLE "+_tsql_
?  xx+"(                                     "


nCharLen := 1 // broj kojim se množi dužina stringa u karakterima
              // zbog povećanja dužine polja u Oracle tabeli u
              // odnosu na dužinu u DBF tabeli

//-------------------------------------------------------
//--- KREIRANJE STRUKTURE REDA TABELE - POLJA U SLOGU
//-------------------------------------------------------


*  PRIMER PL/SQL KOMANDE ZA CREATE TABLE
*  CREATE TABLE DEMO -- napravi novu praznu tabelu DEMO
*  (  "ID_"    NUMBER(6) NOT NULL, -- ovo je integer, a može i PLS_INTEGER
*         "TEKST_" NVARCHAR2(10),      -- podrazumeva da je CHAR a ne BYTE
*     "BROJ_"  NUMBER(20,2),       -- decimalni broj do 38 cifri
*     "DATUM_" DATE,               -- 31-DEC-2016
*     "BUL_"   NCHAR(1),           -- podrazumeva da je CHAR (do 2000 char)
*     "MEMO_"  NVARCHAR2(500)      -- podrazumeva da je CHAR (do 4000 char)
*  );
*  -- Razlika između CHAR() i NCHAR():
*  -- CHAR(1) ili CHAR(1 BYTE) preciznost je 1 BYTE
*  -- CHAR(1 CHAR) preciznost je 1 CHAR
*  -- VARCHAR(10) ili VARCHAR2(10 BYTE) preciznost je u BYTE bajtovima
*  -- VARCHAR2(10 CHAR) preciznost je u CHAR characterima
*  -- NCHAR(1) preciznost je 1 CHAR unicode characterima
*  -- NVARCHAR2(10) preciznost je 10 CHAR unicode characterima
*  --
*  -- ID_ mora imati unikatnu vrednost (ne mogu postojati dva ista ID_ broja)


FOR i = 1 TO nFieldCount // broj polja u slogu
```

```
   cField := aField[i]
   cType  := "xxx"

   IF aType[i]$"C"
      nLen := (aLen[i] * nCharLen)
      nLen := INT(nLen)
      // za NVARCHAR2 ne treba povećavati polje u karakterima
      // pa je nCharLen = 1
      cType := "NVARCHAR2("+var2char(nLen)+")"
      // NVARCHAR2 je dužina polja u unicode karakterima
   ENDIF
   IF aType[i]$"D"
      cType := "DATE"
   ENDIF
   IF aType[i]$"N"
      *---
      * cType := "NUMBER("+var2char(aLen[i])+","+var2char(aDec[i])+")"
      * uzmi da svaki numerik bude NUMBER(20,2)
      * bez obzira koliko je aLen[i] u DBF polju
        cType := "NUMBER("+var2char(20)+","+var2char(aDec[i])+")"
      *---
      * samo ako je numerik bio integer
      * uzmi da svaki numerik bude NUMBER(10)
      * (koristi se za brojače i za šifre)
        IF aDec[i]=0
           cType := "NUMBER(10)"
        ENDIF
      *---
   ENDIF
   IF aType[i]$"L"
   // za DBF boolean .T. ili .F. u oracle tabeli
   // koristi se string i to samo jedan CHAR znak 'T' ili 'F'
      cType := "NCHAR(1)"
   ENDIF
   IF aType[i]$"M"
   // za memo polje dozvoli max 1000 unicode karaktera, bez obzira koliko
   // karaktera ima u DBF memo polju
      * cType := "NVARCHAR2("+var2char(aLen[i])+")"
        cType := "NVARCHAR2(600)"
   ENDIF

   IF i < nFieldCount  // svi osim zadnjeg polja
    ? xx+PADR(cField,20 ) + PADR(cType,20 ) + "," // zarez
   ELSE
    ? xx+PADR(cField,20 ) + PADR(cType,20 ) // nema zarez
   ENDIF

NEXT i

?  xx+");"
?  xx+" "
?  xx+"-- Broj karaktera u CHAR DBF polju pomnožen je sa "+var2char(nCharLen)
?  xx+"-- i dobijen je broj karaktera za NVARCHAR2 polje"
?  xx+"-- Kreiranje nove table kraj --       "
?  xx+" "
//-------------------------------------------------------
//--- KREIRANJE STRUKTURE REDA TABELE - POLJA U SLOGU
//-------------------------------------------------------
```

```
 ***********************************************************************
 * izmeni_tabelu_prema_coba_dbf_fajlu(cDBFpath,_tsql_,aField,cSQLpath,xx)
 // DBF_to_oracle_and_postgres_coba.prg
 ***********************************************************************

 ***************************************************
 // PUNJENJE TABLE
 * podaci_iz_dbf_u_oracle(cDBFpath,_tsql_,aField,xx)
 // DBF_to_oracle_and_postgres_coba.prg
 ***************************************************



?  xx+"SPOOL OFF"
?  xx+"EXIT;"

? xx+" "
?  "-- "+_tsql_+".SQL"
?  "-- END --"
? xx+" "

SET PRINTER TO
SET PRINTER OFF
SET CONSOLE ON

*********************************************************************
* END
* GENERIŠI SQL SCRIPT - KAO ASCII TEXT FAJL SA NAZIVOM: cSQLpath
*********************************************************************    ORACLE - END

//------------------------------------------------------------
// S Q L  S C R I P T  Z A  O R A C L E
//------------------------------------------------------------ end

*************************************************************************
RETURN NIL // FUNCTION __ORACLE__()
*************************************************************************



                  *#################################
                  *         T E S T I N G
                  *#################################

/*

V A Ž N O
---------
Iz ove aplikacije odnosno iz funkcije:

     FUNCTION DBF_TO_POSTGRESQL_SQLSCRIPT(_cDBFpath)
     FUNCTION DBF_TO_ORACLE_SQLSCRIPT(_cDBFpath)

ne može se u ORACLE i PGSQL bazi podataka napraviti TABELA
i napuniti podacima preuzetim iz DBF tabele.

Ova funkcija samo pravi SQL fajl (običan ASCII tekst fajl)
koji se može naknadno startovati iz proceduralnih jezika:
ORACLE database (sqlplus) i PGSQL database (pl/pgsql)
```

```
čime će biti napravljene ORACLE i PGSQL tabele i napunjene
podacima iz DBF tabele

Za test rada funkcije:

     FUNCTION DBF_TO_ORACLE_SQLSCRIPT(_cDBFpath)

treba koristiti DBF tabelu "XBASE2ORACLE.DBF" na osnovu koje
će funkcija generisati SQL fajl: "XBASE2ORACLE.SQL" a koji
posle toga treba startovati u ORACLE database i PGSQL database

Ovde je dat prikaz pravljenja i upotrebe DBF tabele
"XBASE2ORACLE.DBF" i njenog SQL fajla "XBASE2ORACLE.SQL"
kao i prikaz upotrebe tog SQL fajla u ORACLE i PGSQL database.



IMPORTANT
---------
From this application or function:

     FUNCTION DBF_TO_POSTGRESQL_SQLSCRIPT(_cDBFpath)
     FUNCTION DBF_TO_ORACLE_SQLSCRIPT(_cDBFpath)

it is not possible to create a TABLE in ORACLE and PGSQL database
and populate with data taken from the DBF table.

This function only creates an SQL file (plain ASCII text file).
which can subsequently be started from procedural languages:
ORACLE database (sqlplus) and PGSQL database (pl/pgsql)
which will create the ORACLE and PGSQL tables and populate them
data from the DBF table

To test the functionality:

      FUNCTION DBF_TO_ORACLE_SQLSCRIPT(_cDBFpath)

should use the DBF table "XBASE2ORACLE.DBF" based on which
the function will generate an SQL file: "XBASE2ORACLE.SQL" which
after that you should start in ORACLE database and PGSQL database

Here is an overview of creating and using a DBF table
"XBASE2ORACLE.DBF" and its SQL file "XBASE2ORACLE.SQL"
as well as a display of the use of that SQL file in ORACLE and PGSQL database.

*/


**************************************************************************
FUNCTION test__oracle__dbf_file_to_sql_table(lokacija)
**************************************************************************
*
* Ovaj XBASE2ORACLE.DBF fajl služi kao primer za kreiranje programa
*
*        FUNCTION DBF_TO_ORACLE_SQLSCRIPT(_cDBFpath)
*
* kojim se iz DBF fajla creira SQL fajl za kreiranje ORACLE database i
* ORACLE SQL table XBASE2ORACLE
*
```

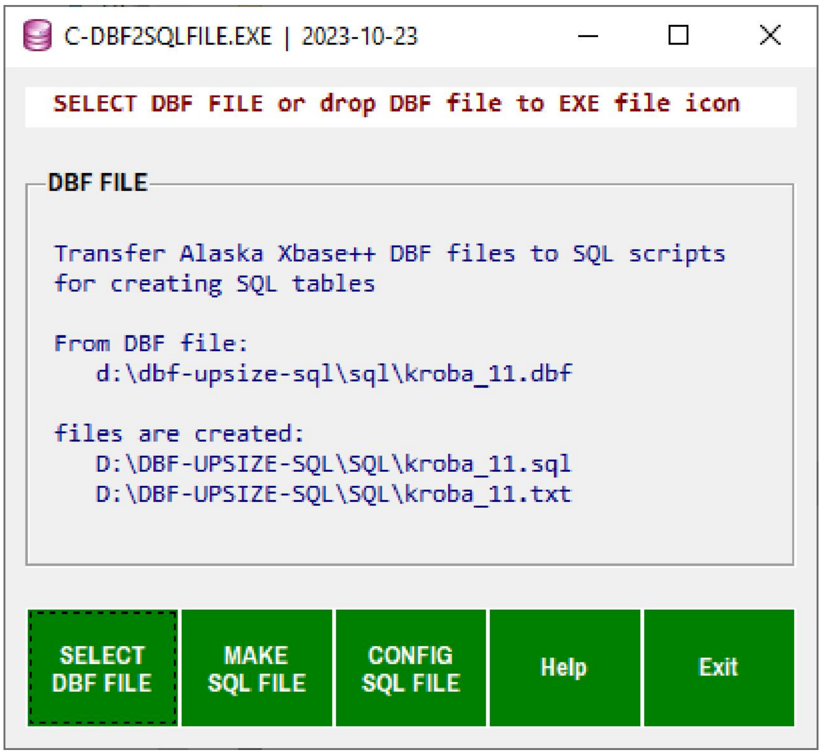
```
//-----------------------------------------------------------------
// (1) TESTIRANJE - PRVO SE NAPRAVI DBF fajl XBASEORACLE.DBF
// (1) TESTING - FIRST CREATE DBF file XBASEORACLE.DBF
//-----------------------------------------------------------------
      LOCAL radno := SELECT(), astructure := {}
      SELECT 0 // prvo slobodno područje
      db:=lokacija+"\"+"XBASE2ORACLE.DBF"
      DELETE FILE (db)

      aStructure := { ;
        { "ID_"         ,"N",   6, 0 } ,;  // N=NUMERIC-INTEGER FIELD
        { "TEKST_"      ,"C",  10, 0 } ,;  // C=CHARACTER FIELD
        { "BROJ_"       ,"N",  10, 2 } ,;  // N=NUMERIC-DECIMAL FIELD
        { "DATUM_"      ,"D",   8, 0 } ,;  // D=DATE FIELD
        { "BUL_"        ,"L",   1, 0 } ,;  // L=LOGIC FIELD
        { "MEMO_"       ,"M",  10, 0 }  ;  // M=MEMO FIELD
      }
      DbCreate( db, aStructure, "DBFNTX" )

//----------------------------------------------------------------------
// (2) TESTIRANJE - DODA SE JEDAN SLOG U TU DBF TABELU XBASEORACLE.DBF
// (2) TESTING - ADDING ONE RECORD TO THAT DBF TABLE XBASEORACLE.DBF
//----------------------------------------------------------------------
      USE (db) NEW SHARED
      APPEND BLANK
      //---DATE-FILL iz VARIJABLI:
           REPLACE DATUM_          WITH DATE()
      //---CARACTER-FILL iz VARIJABLI:
           REPLACE TEKST_          WITH "Text=ŠĐČĆŽ"
      //---NUMERIC-FILL iz  VARIJABLI:
           REPLACE BROJ_           WITH 123456.78
      //---LOGIC-FILL iz VARIJABLI:
           REPLACE BUL_            WITH .T.

dugi_text := ;
      "Ovde se upisuje 'dugi tekst' koji može da sadrži"+chr(13)+chr(10)+;
      "Serbian Latin i Serbian Cyrilic fontove.        "+chr(13)+chr(10)+;
      "Serbian Latin:..... ŠĐČĆŽ šđčćž                 "+chr(13)+chr(10)+;
      "Serbian Cyrilic:... QWŠĐČĆŽX qwšđčćžx           "+chr(13)+chr(10)+;
      "treba da budu Unicode UTF-8.                    "+chr(13)+chr(10)+;
      "Jednostruke navodnike u tekstu prevesti u znak |"+chr(13)+chr(10)+;
      "---------- ENG --------"                         +chr(13)+chr(10)+;
      "Here you enter 'long text' which can contain"    +chr(13)+chr(10)+;
      "Serbian Latin and Serbian Cyrillic fonts. "      +chr(13)+chr(10)+;
      "Serbian Latin:..... ŠĐČĆŽ šđčćž "                +chr(13)+chr(10)+;
      "Serbian Cyrillic:... QWŠĐČĆŽX qwšđčćžx "         +chr(13)+chr(10)+;
      "should be Unicode UTF-8. "                       +chr(13)+chr(10)+;
      "Translate single quotes in the text to |"

      //---MEMO-FILL iz VARIJABLI:
           REPLACE MEMO_           WITH dugi_text

      USE
      SELECT(radno)

  RETURN NIL
```

```
//----------------------------------------------------------------------------
// (3) TESTIRANJE - NAPRAVI SE SQL SCRIPT ZA PRENOS PODATAKA IZ DBF U ORACLE
// (3) TESTING - CREATE A SQL SCRIPT TO TRANSFER DATA FROM DBF TO ORACLE
//----------------------------------------------------------------------------

* GENERATING SQL FILE: XBASE2ORACLE.SQL
*
* program za generisanje ORACLE SQL fajla iz DBF fajla
* generisaće SQL fajl
*
*   XBASE2ORACLE.SQL
*
* koji će startom u SQLplus komand promptu ili u SQLdeveloperu,
* formirati Oracle tabelu XBASE2ORACLE za konekciju:
*
*   CONNECT user/password
*
* ---
*
* GENERATING SQL FILE: XBASE2ORACLE.SQL
*
* program for generating an ORACLE SQL file from a DBF file
* will generate a SQL file
*
* XBASE2ORACLE.SQL
*
* which will start in the SQLplus command prompt or in SQLdeveloper,
* create the Oracle table XBASE2ORACLE for the connection:
*
* CONNECT user/password



/*
************************************************************************* START SQL
--
-- START
-- XBASE2ORACLE.SQL
--
-------------------------------------------------------------
-- COBA Systems 29.09.2016
-- Open Source Projct: Alaska Xbase++ DBF To Oracle DBF
-- Automatski generisano iz Xbase++ programa XBASE2ORACLE.EXE
-- SQL Script: Z:\[XW]\{DBFDOC}\DBFDOC\XBASE2ORACLE.SQL
-- Translate Xbase++ DBF file To Oracle Table
-- Xbase: XBASE2ORACLE.DBF To Oracle: XBASE2ORACLE
-------------------------------------------------------------

  --
  -- X B A S E  F I L E -- XBASE2ORACLE.DBF
  --
  --        1 ID_        N    6 0 --
  --        2 TEKST_     C   10 0 --
  --        3 BROJ_      N   10 2 --
  --        4 DATUM_     D    8 0 --
  --        5 BUL_       L    1 0 --
  --        6 MEMO_      M   10 0 --
  --
  -- O R A C L E  T A B L E -- XBASE2ORACLE
  --
  -- Konekcija:
  -- CONNECT user/password iz fajlova:
  -- oracle_user.cfg i oracle_password.cfg
  --
  CONNECT COBASYSTEMS/Coba#1
  SPOOL XBASE2ORACLE.LST
  SET LINESIZE 100
  SET PAGESIZE 50

  -- lokalizacija: format datuma i kodna strana = UTF --
  ALTER SESSION SET NLS_TERRITORY = 'SERBIA AND MONTENEGRO';

  -- Brisanje postojeće table start --
  DROP TABLE XBASE2ORACLE;

  -- Kreiranje nove table start --
  CREATE TABLE XBASE2ORACLE
  (
  ID_                   NUMBER(10)          ,
  TEKST_                NVARCHAR2(10)       ,
  BROJ_                 NUMBER(20,2)        ,
  DATUM_                DATE                ,
  BUL_                  NCHAR(1)            ,
  MEMO_                 NVARCHAR2(600)
  );

  -- Broj karaktera u CHAR DBF polju pomnožen je sa 1
  -- i dobijen je broj karaktera za NVARCHAR2 polje
  -- Kreiranje nove table kraj --


  -- Unos slogova u tabelu start --

  INSERT INTO XBASE2ORACLE
  VALUES
  (
  0,
  'Text=ŠĐČĆŽ',
  123456.78,
  '29.09.2016',
  'T',
  'Ovde se upisuje |dugi tekst| koji može da sadrži
Serbian Latin i Serbian Cyrilic fontove.
Serbian Latin:..... ŠĐČĆŽ šđčćž
Serbian Cyrilic:... QWŠĐČĆŽX qwšđčćžx
treba da budu Unicode UTF-8.
Jednostruke navodnike u tekstu prevesti u znak |'
  );

  -- Unos slogova u tabelu kraj --
  -- Ukupno slogova = 1 --

  SPOOL OFF
  EXIT;

-- XBASE2ORACLE.SQL
-- END --
```

```
************************************************************************* END SQL
*/


* KAKO SE KORISTI DOBIJENI SQL FAJL
*      XBASE2ORACLE.SQL
*
* HOW TO USE THE OBTAINED SQL FILE
*      XBASE2ORACLE.SQL
*

/*
************************************************************************* START

*------------------------------------
* CREATE ORACLE TABLE 'XBASE2ORACLE'
*------------------------------------

Ovaj SQL script fajl XBASE2ORACLE.SQL startuje se iz command line:
---
This SQL script file XBASE2ORACLE.SQL is started from the command line:

   C:\> sqlplus /nolog @XBASE2ORACLE.SQL

ili iz XBASE2ORACLE.BAT fajla čiji je sadržaj:
---
or from the XBASE2ORACLE.BAT file whose content is:

   ECHO OFF
   sqlplus /nolog @XBASE2ORACLE.SQL
   PAUSE


Ako BAT fajl nije u folderu u kome je XBASE2ORACLE.SQL
tada se mora navesti pun PATH do XBASE2ORACLE.SQL na primer:
---
If the BAT file is not in the folder where XBASE2ORACLE.SQL is
then the full PATH to XBASE2ORACLE.SQL must be specified, for example:

   ECHO OFF
   sqlplus /nolog @C:\TEST\XBASE2ORACLE.SQL
   PAUSE


Izvršavanje komande sqlplus /nolog @XBASE2ORACLE.SQL
izdaje poruke na ekranu i iste poruke upisuje u
log faj pod nazivom XBASE2ORACLE.LST
---
Executing the command sqlplus /nolog @XBASE2ORACLE.SQL
outputs messages on the screen and writes the same messages to
log file named XBASE2ORACLE.LST

U fajlu XBASE2ORACLE.LST koji će biti generisan
u istom folderu gde i XBASE2ORACLE biće sadržaj:
---
In the XBASE2ORACLE.LST file that will be generated
in the same folder as XBASE2ORACLE will be the content:
```

```
Session altered.
Table dropped.
Table created.
1 row created.

Ako operacija nije izvršena ovde će biti poruke
o greškama zbog kojih nije izvršena.
---
If the operation was not performed, there will be messages here
about the errors due to which it was not executed.

-------------------------------------------------------------------------

*-----------------------------------
* CREATE ORACLE TABLE 'XBASE2ORACLE'
*-----------------------------------

   PRIMER ORACLE PL/SQL KOMANDE ZA CREATE TABLE
   ---
   EXAMPLE OF ORACLE PL/SQL COMMAND FOR CREATE TABLE


   CREATE TABLE XBASEORACLE          -- create new blanko table
   (   "ID_"    NUMBER(6) NOT NULL, -- means integer, or PLS_INTEGER
        "TEKST_" NVARCHAR2(10),      -- means CHAR - not BYTE
       "BROJ_"  NUMBER(20,2),       -- means decimal number up to 38 digits
       "DATUM_" DATE,               -- means 31-DEC-2016
       "BUL_"   NCHAR(1),           -- means CHAR (up 2000 char)
       "MEMO_"  NVARCHAR2(500)      -- means CHAR (up 4000 char)
   );
   -- Razlika između CHAR() i NCHAR():
   -- Difference between CHAR() and NCHAR():

   -- CHAR(1) or CHAR(1 BYTE) precision is 1 BYTE
   -- CHAR(1 CHAR) precision is 1 CHAR
   -- VARCHAR(10) ili VARCHAR2(10 BYTE) precision in BYTE
   -- VARCHAR2(10 CHAR) precision in CHAR
   -- NCHAR(1) precision is 1 CHAR unicode characters
   -- NVARCHAR2(10) precision is 10 CHAR unicode characters
   --
   -- ID_ mora imati unikatnu vrednost (ne mogu postojati dva ista ID_ broja)
   -- ID_ must have a unique value (there cannot be two identical ID_ numbers)

************************************************************************* END
*/
```

C-DBF2SQLFILE.EXE

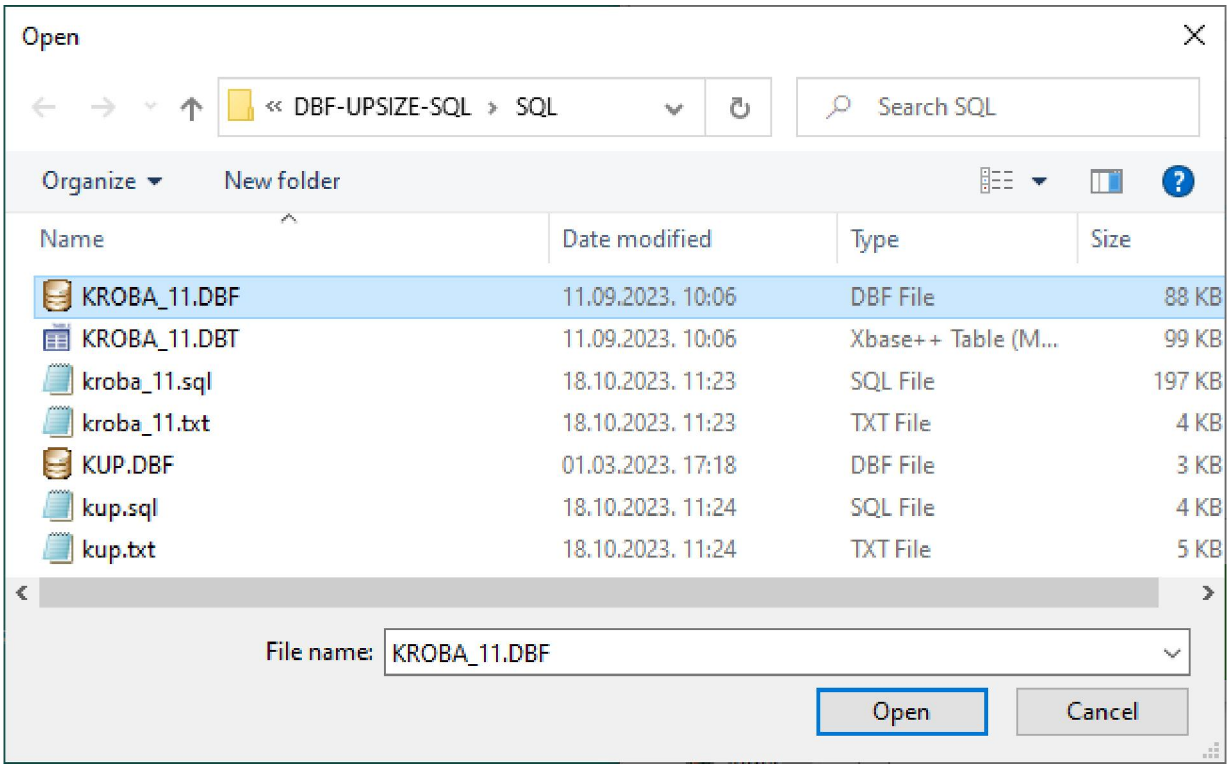

C-DBF2SQLFILE.EXE | 2023-10-23

SELECT DBF FILE or drop DBF file to EXE file icon

DBF FILE

```
Transfer Alaska Xbase++ DBF files to SQL scripts
for creating SQL tables

From DBF file:
   d:\dbf-upsize-sql\sql\kroba_11.dbf

files are created:
   D:\DBF-UPSIZE-SQL\SQL\kroba_11.sql
   D:\DBF-UPSIZE-SQL\SQL\kroba_11.txt
```

SELECT DBF FILE


Open

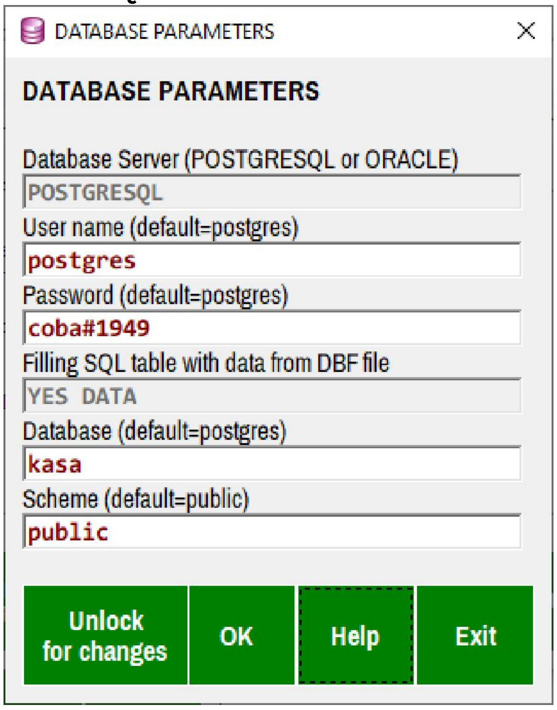
« DBF-UPSIZE-SQL › SQL

Search SQL

Name	Date modified	Type	Size
KROBA_11.DBF	11.09.2023. 10:06	DBF File	88 KB
KROBA_11.DBT	11.09.2023. 10:06	Xbase++ Table (M...	99 KB
kroba_11.sql	18.10.2023. 11:23	SQL File	197 KB
kroba_11.txt	18.10.2023. 11:23	TXT File	4 KB
KUP.DBF	01.03.2023. 17:18	DBF File	3 KB
kup.sql	18.10.2023. 11:24	SQL File	4 KB
kup.txt	18.10.2023. 11:24	TXT File	5 KB

File name: KROBA_11.DBF

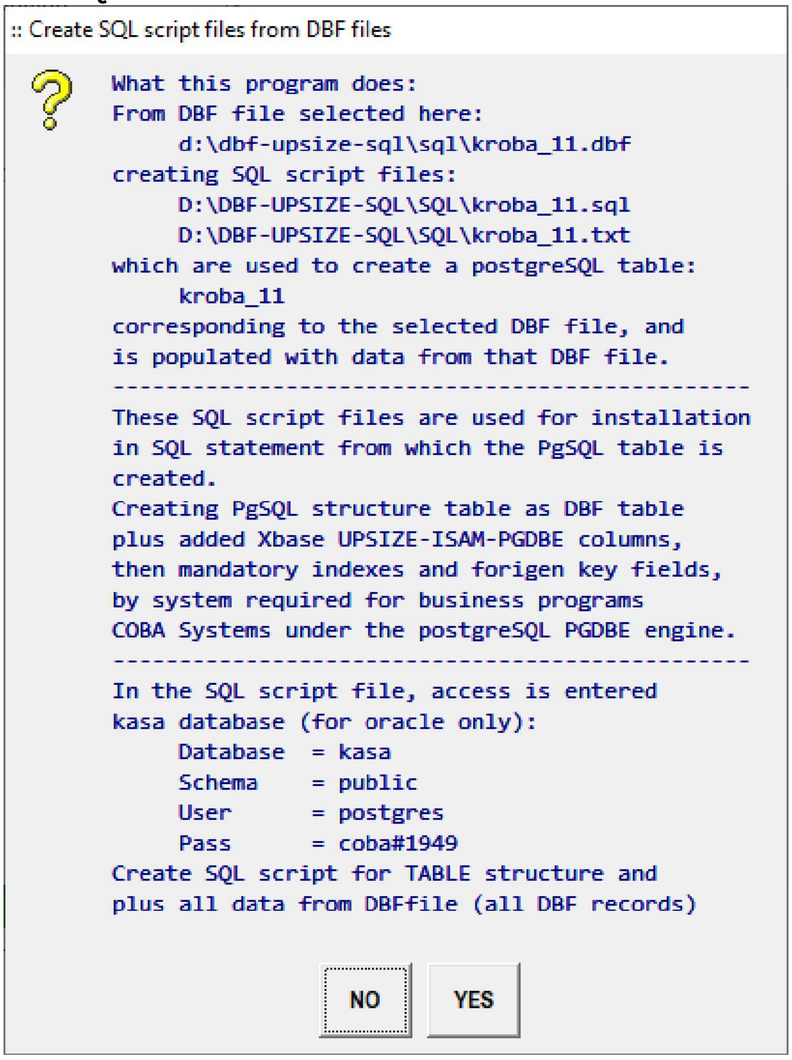
CONFIG SQL FILE



DATABASE PARAMETERS

Database Server (POSTGRESQL or ORACLE): POSTGRESQL
User name (default=postgres): postgres
Password (default=postgres): coba#1949
Filling SQL table with data from DBF file: YES DATA
Database (default=postgres): kasa
Scheme (default=public): public

MAKE SQL FILE



```
What this program does:
From DBF file selected here:
     d:\dbf-upsize-sql\sql\kroba_11.dbf
creating SQL script files:
     D:\DBF-UPSIZE-SQL\SQL\kroba_11.sql
     D:\DBF-UPSIZE-SQL\SQL\kroba_11.txt
which are used to create a postgreSQL table:
     kroba_11
corresponding to the selected DBF file, and
is populated with data from that DBF file.
------------------------------------------------
These SQL script files are used for installation
in SQL statement from which the PgSQL table is
created.
Creating PgSQL structure table as DBF table
plus added Xbase UPSIZE-ISAM-PGDBE columns,
then mandatory indexes and forigen key fields,
by system required for business programs
COBA Systems under the postgreSQL PGDBE engine.
------------------------------------------------
In the SQL script file, access is entered
kasa database (for oracle only):
     Database  = kasa
     Schema    = public
     User      = postgres
     Pass      = coba#1949
Create SQL script for TABLE structure and
plus all data from DBFfile (all DBF records)
```

Note:

U delu 3 ove knjige biće opisan i dokumentovan program
Part 3 of this book will describe and document the program
C-POSTRGRESQL-DATABASE.EXE (supplementary program with each EXE application)

U delu 4 ove knjige biće opisan i dokumentovan program
Part 4 of this book will describe and document the program
C-PGDB.DLL (common procedures and functions for applications)