

## APLICACIÓN PRÁCTICA

### REQUERIMIENTOS DE HARDWARE

2 Computadores con procesadores Pentium II en adelante o Celeron con procesador 700 en adelante, Memoria RAM 128

### REQUERIMIENTOS DE SOFTWARE

Los sistemas operativos que deben ser instalados en los computadores dependen de cómo se desea realizar la replicación:

#### **Sistemas operativos**

Windows NT y Linux

#### **Paquetes**

##### **Servidor Linux**

Postgresql Versión 6.x

##### **Servidor Windows NT**

Microsoft Visual FoxPro 6.0

Postgresql ODBC Driver

La creación de las librerías en C++ y su compilación se encuentran descritas en el Capítulo IV

### DESCRIPCIÓN DEL PROGRAMA DE REPLICACIÓN

La aplicación que se encuentra en el CDROM se la realizó en Microsoft Visual FoxPro 6.0 y se encuentra en el directorio *conexión* la cual se deberá copiar al disco duro donde se encuentra la aplicación del mismo nombre, a continuación se detalla el funcionamiento de la misma:

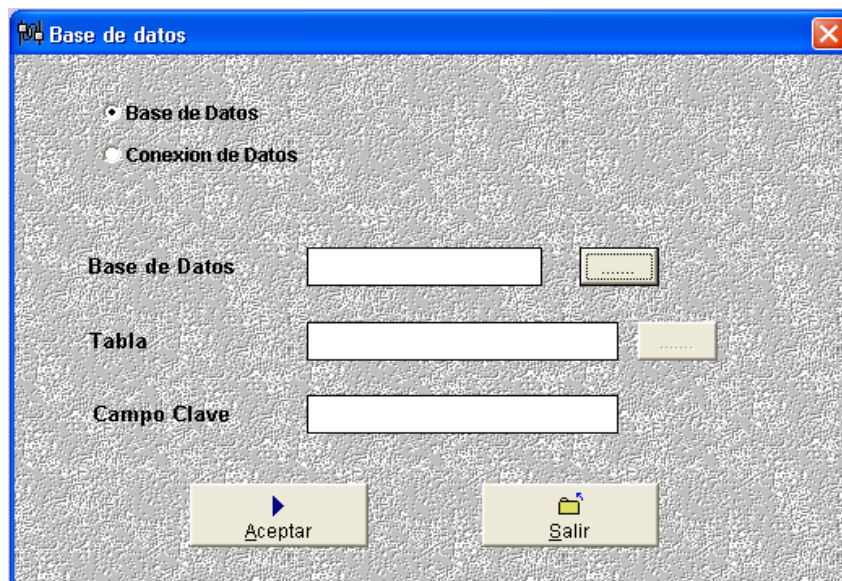
El programa de conexión y replicación de base de datos consiste en hacer una conexión o abrir una base de datos para luego hacer la replicación con las diferentes tablas elegidas en conexión. La conexión de datos debe ser de un ODBC de una base de datos creada en Postgresql.

El programa consta del siguiente menú:



## OPCIÓN CONEXIÓN

Al hacer clic en esta opción del menú se muestra la siguiente pantalla:



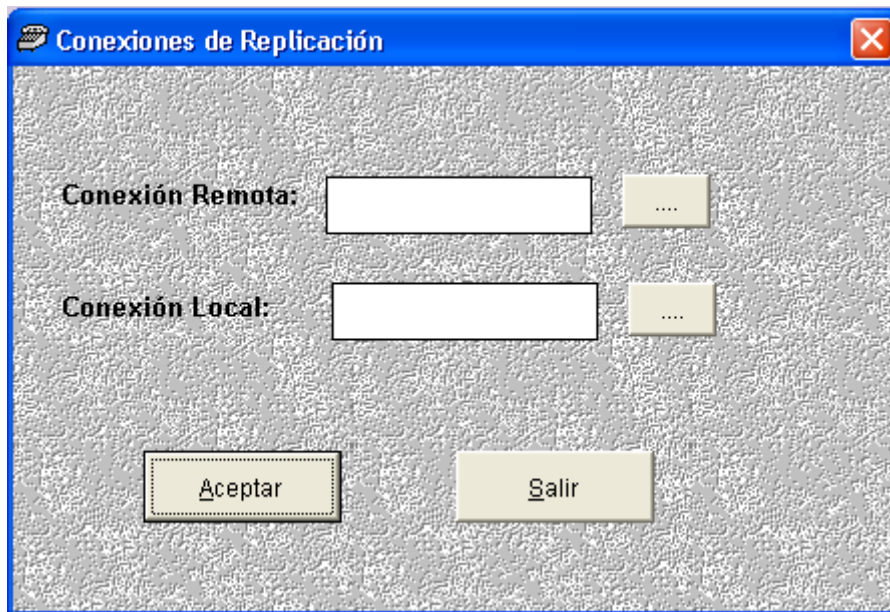
Si elegimos **Base de Datos** en el grupo de opciones se deberá escoger una base de datos creada en Microsoft Visual FoxPro, una tabla y su campo clave. Al hacer clic en **Aceptar** se crearán los procedimientos almacenados y los desencadenantes en la tabla seleccionada.

Si elegimos **Conexión de Datos** debemos de elegir un ODBC de Postgresql o de Visual FoxPro, a continuación en una tabla y su campo clave

Nota: esta información se almacena en la tabla `rp_tablas`, esta información almacenada consta del nombre de la tabla, al campo índice, tipo de dato del índice y la longitud del campo índice.

## Replicación

Esta opción nos permitirá replicar la información de las tablas que se encuentran almacenadas en la tabla `rp_tablas` de la conexión remota a la conexión local



La replicación se la realiza de tres maneras:

- Visual FoxPro a Visual FoxPro
- Visual FoxPro a Postgresql
- Postgresql a Visual FoxPro

## CÓDIGO FUENTE DEL SISTEMA CONEXIÓN

A continuación se describe los programas utilizados en el sistema de replicación:

### Forma base5.scx

#### Command1

#### Evento Click

Leer base de datos o conexión de base de datos

```
private datos1,a
do case
    case thisform.optiongroup1.value=1
        datos1=""
        datos1 = GETFILE('DBC', 'Base de Datos .dbc:', 'Aceptar',0)
        thisform.text1.value=datos1
        thisform.refresh()
    case thisform.optiongroup1.value=2
        a=sqlconnect()
        if a>=1
            =sqltables(a,'TABLE','tablas')
            select tablas
            if len(alltrim((tablas.table_qualifier) ))=0
                =messagebox('conexion realizada',0)
                thisform.text1.value=SQLGETPROP(a,'DataSource')
                thisform.refresh
                =sqldisconnect(a)
            else
                =messagebox("La conexion debe ser de una base de Datos
Postgresql",0)
                thisform.text1.value=""
                thisform.refresh
                =sqldisconnect(0)
            endif
        else
            =messagebox('conexion fallida',0)
        endif
    endif
endcase
```

## Command2

### Evento Click

#### Leer Tabla en caso de

```
private datos1,a
do case
    case thisform.optiongroup1.value=1
        datos1=""
        datos1 = GETFILE('DBF', 'Tabla .dbf:', 'Aceptar',0)
        thisform.text2.value=datos1
        thisform.refresh()
    endcase
```

### Siguiente

### Evento Click

Creas tablas rp\_control y rp\_tablas en la base de datos seleccionada y los desencadenantes inserta(), actualiza() y elimina() en cada una de las tablas ingresadas.

```
private
ban,indice,suma,camino,cadena,tipo,p,m,b,o,camino1,numero1,fin,finuno,uni,cc,cc1,tabla22,tipo1
dimension vector(100,2)
tipo1=""
band=.F.
indice=""
suma=0
camino=""
camino1=""
m=""
b=.F.
o=""
cc=""
cc1=""
numero1=0
camino1=sys(2003)
uni=sys(5)
fin=""
finuno=""
tabla22=""
cc1=uni+camino1
tipo=""
cpcurrent(1)
do case
case thisform.optiongroup1.value=1
indice=thisform.text3.value
on error messagebox(message())
set exclusive off
```

```

open database (thisform.text1.value) validate
if error()#0
messagebox(message(),0)
else
=ADBOBJECTS(gaTables, "TABLE")
tabla22=""
tabla22=upper(alltrim(SUBSTR(thisform.text2.value, rat('\', thisform.text2.value)+1, len(t
hisform.text2.value)-rat('\', thisform.text2.value))))
if rat('.', alltr(tabla22))=0
tabla22=upper(alltrim(tabla22))
else
tabla22=upper(SUBSTR(alltrim(tabla22), 1, rat('.', alltr(tabla22))-1))
endif
tabla22=upper(alltrim(tabla22))
for i=1 to alen(gatables)
if upper(alltrim(gatables(1,i)))==upper(alltrim(tabla22))
i=i+alen(gatables)+1
band=.T.
endif
endifor
if band
close tables all
select 1
use (thisform.text2.value)
select 1
on error messagebox(message(),0)
evaluate(thisform.text3.value)
if error()#0
=messagebox(message(),0)
*=sqldisconnect(a)
thisform.msgerror()
return
else
select 1
m=alltrim(upper(thisform.text3.value))
for i=1 to fcount()
if ratc(upper(field(i)), m, 1) >= 1
suma=suma+fsize(field(i))
endif
next i
cadena=thisform.text3.value
p=thisform.text3.value
tipo=alltrim(p)
*wait window tipo
if rat("+", tipo, 1) > 0
tipo1='C'
else
)
tipo1=type('&tipo')
endif

```

```

*wait window "estamos bien"
*wait window tipo1
camino=substr(thisform.text1.value,1,atc("\",thisform.text1.value,1))
set path to (camino)
set defa to (camino)
*wait window camino
if file(camino+'rp_tablas.dbf')
*wait window "encontrando rp_tablas"
set defa to (camino)
*close tables all
use
open data (thisform.text1.value)
use rp_tablas in 0
select rp_tablas
*delete all
*wait window "usada rp_tablas"
locate for lower(alltrim(tabla))=lower(alltrim(tabla22))
if !found()
    append blank
    replace tabla with lower(tabla22)
    replace clave with thisform.text3.value
    replace tipodato with tipo1
    replace longitud with suma
    replace replica with "replica"
endif
use
*wait window "terminamos con rp_tablas"
if !file(camino+'rp_control.dbf')
    create table rp_control(tabla c(20), valor m, valorant m, operacion c(1), fecha T,
otros c(20), semaforo l)
endif
cc1=alltrim(uni)+alltrim(camino1)+"\procedimiento1.txt"
APPEND PROCEDURES FROM (cc1) overwrite
*wait window "creando procedimiento"
create trigger on (thisform.text2.value) for insert as inserta()
create trigger on (thisform.text2.value) for update as actualiza()
create trigger on (thisform.text2.value) for delete as elimina()
thisform.text1.value=""
thisform.text2.value=""
thisform.text3.value=""
thisform.refresh()
close tables all
else
set defa to (camino)
use
*close tables all
open data (thisform.text1.value)
create table rp_tablas(tabla c(20), clave m, tipodato c(1),longitud n(3), replica m)
*wait window "tabla de base de datos creada"
select rp_tablas

```

```

appen blank
replace tabla with lower(tabla22)
replace clave with lower(thisform.text3.value)
*wait window tipo1
replace tipodato with tipo1
replace longitud with suma
replace replica with "replica"
use
if !file(camino+'rp_control.dbf')
    create table rp_control(tabla c(20), valor m, valorant m, operacion c(1), fecha T,
otros c(20), semaforo l)
endif
cc1=alltrim(uni)+alltrim(camino1)+"\procedimiento1.txt"
APPEND PROCEDURES FROM (cc1) overwrite
*wait window "creando procedimientos almacenados"
create trigger on (thisform.text2.value) for insert as inserta()
create trigger on (thisform.text2.value) for update as actualiza()
create trigger on (thisform.text2.value) for delete as elimina()
thisform.text1.value=""
thisform.text2.value=""
thisform.text3.value=""
thisform.refresh()
close tables all
endif
endif
*wait window "acabamos"
close tables all
*set defa to (camino1)
on error
else
=messagebox("Tabla no existe en la Base de Datos "+thisform.text1.value,0)
thisform.text1.value=""
thisform.text2.value=""
thisform.text3.value=""
thisform.refresh()
return
close tables all
endif
endif
on error
*set defa to (cc1)
case thisform.optiongroup1.value=2
cc=alltrim(thisform.text1.value)
if len(cc)<=0
=messagebox("Ingrese conexión")
thisform.msgerror()
return
endif
a=sqlconnect(cc)
if a>=1

```



```

thisform.borratodo
if sqltables(a,'TABLE','cursor1')>=1
select cursor1
go top
do while !eof()
if upper(alltrim(thisform.text2.value))=upper(alltrim(table_name))
b=.T.
endif
select cursor1
skip
enddo
if b
if sqlexec(a,'select * from '+thisform.text2.value,'cursor2')>=1
select cursor2
m=alltrim(upper(thisform.text3.value))
select cursor2
on error messagebox(message(),0)
evaluate(thisform.text3.value)
if error()#0
messagebox(message(),0)
thisform.msgerror()
return
else
suma=0
for i=1 to fcount()
if ratc(upper(field(i)),m,1)>=1
suma=suma+fsize(field(i))
endif
next i
p=thisform.text3.value
*select cursor3
tipo=p
if rat("+",tipo,1)>0
tipo1='C'
else
tipo1=type('&tipo')
endif
if sqlexec(a,'select * from rp_tablas','cursor3')>=1
*p=thisform.text3.value
*select cursor3
*tipo=p
*if rat("+",tipo,1)>0
* tipo1='C'
*else
* tipo1=type('&tipo')
*endif
if sqlexec(a,'insert into rp_tablas values
('+o+alltrim(thisform.text2.value)+o+', '+o+alltrim(thisform.text3.value)+o+',
'+o+upper(alltrim(tipo1))+o+', '+o+alltrim(str(suma))+o+', '+o+'replica'+o+')')>=1
if sqlexec(a,'select * from rp_control','cursor4')<=0

```

```
if sqlexec(a,'create table rp_control(tabla char(20), valor
text, valorant text, operacion char(1), fecha timestamp, otros char(20), semaforo
bool)')<=0
```

```
    =messagebox('No se puede crear rp_control',0)
    thisform.msgerror()
    return
```

```
else
```

```
    if sqlexec(a,"create function inserta() returns
opaque as '/usr/local/replicap1.0b/lib/replica.so' language 'C'")<=0
```

```
        =messagebox('No se puede crear la funcion
insertar',0)
```

```
        thisform.msgerror()
        return
```

```
    else
```

```
        if sqlexec(a,'create trigger inserta after
insert on '+alltrim(thisform.text2.value)+' for each row execute procedure inserta()')<=0
```

```
            =messagebox('No se pudo crear
Trigger en la tabla '+alltrim(thisform.text2.value),0)
```

```
            thisform.msgerror()
            return
```

```
        endif
```

```
    endif
```

```
    if sqlexec(a,"create function actualiza() returns
opaque as '/usr/local/replicap1.0b/lib/replica.so' language 'C'")<=0
```

```
        =messagebox('No se puede crear la funcion
actualiza',0)
```

```
        thisform.msgerror()
        return
```

```
    else
```

```
        if sqlexec(a,'create trigger actualiza after
update on '+alltrim(thisform.text2.value)+' for each row execute procedure
actualiza()')<=0
```

```
            =messagebox('No se pudo crear
Trigger en la tabla '+alltrim(thisform.text2.value),0)
```

```
            thisform.msgerror()
            return
```

```
        endif
```

```
    endif
```

```
    if sqlexec(a,"create function elimina() returns
opaque as '/usr/local/replicap1.0b/lib/replica.so' language 'C'")<=0
```

```
        =messagebox('No se puede crear la funcion
elimina',0)
```

```
        thisform.msgerror()
        return
```

```
    else
```

```
        if sqlexec(a,'create trigger elimina after
delete on '+alltrim(thisform.text2.value)+' for each row execute procedure
elimina()')<=0
```

```

Trigger en la tabla '+alltrim(thisform.text2.value),0)
=messagebox('No se pudo crear
thisform.msgerror()
return
endif
endif
endif
endif
else
=messagebox('No se puede insertar registro',0)
thisform.msgerror()
return
endif
else
*select cursor3
p=thisform.text3.value
tipo=p
*wait window tipo
if rat("+",tipo,1)>0
tipo1='C'
else
tipo1=type('&tipo')
endif
if sqlxexec(a,'create table rp_tablas(tabla char(20), clave char(50), tipodato
char(1),longitud numeric(3), replica text')>=1
if sqlxexec(a,'select * from rp_tablas','cursor3')>=1
*select cursor3
*p=thisform.text3.value
*tipo=p
*wait window tipo
*if rat("+",tipo,1)>0
* tipo1='C'
*else
* tipo1=type('&tipo')
*endif
if sqlxexec(a,'insert into rp_tablas values
('+o+alltrim(thisform.text2.value)+o+', '+o+alltrim(thisform.text3.value)+o+',
'+o+upper(alltrim(tipo1))+o+', '+alltrim(str(suma))+', '+o+'replica'+o+')')>=1
if sqlxexec(a,'select * from rp_control','cursor4')<=0
if sqlxexec(a,'create table rp_control(tabla
char(20), valor text, valorant text, operacion char(1), fecha timestamp, otros char(20),
semaforo bool)')<=0
=messagebox('No se puede crear
thisform.msgerror()
return
else
rp_control',0)

```

```

if sqlxexec(a,"create function
inserta() returns opaque as '/usr/local/replicap1.0b/lib/replica.so' language 'C'")<=0
    =messagebox('No se puede
crear la funcion insertar',0)
    thisform.msgerror()
    return
else
    if sqlxexec(a,'create trigger
inserta after insert on '+alltrim(thisform.text2.value)+' for each row execute procedure
inserta()')<=0
        =messagebox('No se
pudo crear Trigger en la tabla '+alltrim(thisform.text2.value),0)
        thisform.msgerror()
        return
    endif
endif
if sqlxexec(a,"create function
actualiza() returns opaque as '/usr/local/replicap1.0b/lib/replica.so' language 'C'")<=0
    =messagebox('No se puede
crear la funcion actualiza',0)
    thisform.msgerror()
    return
else
    if sqlxexec(a,'create trigger
actualiza after update on '+alltrim(thisform.text2.value)+' for each row execute
procedure actualiza()')<=0
        =messagebox('No se
pudo crear Trigger en la tabla '+alltrim(thisform.text2.value),0)
        thisform.msgerror()
        return
    endif
endif
if sqlxexec(a,"create function
elimina() returns opaque as '/usr/local/replicap1.0b/lib/replica.so' language 'C'")<=0
    =messagebox('No se puede
crear la funcion elimina',0)
    thisform.msgerror()
    return
else
    if sqlxexec(a,'create trigger
elimina after delete on '+alltrim(thisform.text2.value)+' for each row execute procedure
elimina()')<=0
        =messagebox('No se
pudo crear Trigger en la tabla '+alltrim(thisform.text2.value),0)
        thisform.msgerror()
        return
    endif
endif
endif
else

```

```

if sqlexec(a,"create function inserta()
returns opaque as '/usr/local/replicap1.0b/lib/replica.so' language 'C'")<=0
    =messagebox('No se puede crear la
funcion insertar',0)
    thisform.msgerror()
    return
else
    if sqlexec(a,'create trigger inserta
after insert on '+alltrim(thisform.text2.value)+' for each row execute procedure
inserta()')<=0
        =messagebox('No se pudo
crear Trigger en la tabla '+alltrim(thisform.text2.value),0)
        thisform.msgerror()
        return
    endif
endif
if sqlexec(a,"create function actualiza()
returns opaque as '/usr/local/replicap1.0b/lib/replica.so' language 'C'")<=0
    =messagebox('No se puede crear la
funcion actualiza',0)
    thisform.msgerror()
    return
else
    if sqlexec(a,'create trigger actualiza
after update on '+alltrim(thisform.text2.value)+' for each row execute procedure
actualiza()')<=0
        =messagebox('No se pudo
crear Trigger en la tabla '+alltrim(thisform.text2.value),0)
        thisform.msgerror()
        return
    endif
endif
if sqlexec(a,"create function elimina()
returns opaque as '/usr/local/replicap1.0b/lib/replica.so' language 'C'")<=0
    =messagebox('No se puede crear la
funcion elimina',0)
    thisform.msgerror()
    return
else
    if sqlexec(a,'create trigger elimina
after delete on '+alltrim(thisform.text2.value)+' for each row execute procedure
elimina()')<=0
        =messagebox('No se pudo
crear Trigger en la tabla '+alltrim(thisform.text2.value),0)
        thisform.msgerror()
        return
    endif
endif
endif
endif

```

```

                endif
            endif
        else
            =messagebox('Error al intentar abrir Tabla rp_tablas',0)

            thisform.msgerror()
            return
        endif
    else
        =messagebox('Error al crear Tabla rp_tablas',0)
        thisform.msgerror()
        return
    endif
endif
endif
endif
else
=messagebox('Error en Conexión',0)
thisform.msgerror()
return
endif
on error
else
messagebox("Tabla no existe en la conexión "+thisform.text1.value,0)
=sqldisconnect(a)
return
endif
else
messagebox("Tabla no existe en la conexión "+thisform.text1.value,0)
=sqldisconnect(a)
return
endif
=sqldisconnect(a)
else
=messagebox('conexión fallida',0)
=sqldisconnect(a)
return
endif
close tables all
set defa to
set path to (alltrim(uni)+alltrim(camino1))
set defa to (alltrim(uni)+alltrim(camino1))
thisform.text1.value=""
thisform.text2.value=""
thisform.text3.value=""
thisform.refresh()
otherwise
messagebox('Opción no disponible',0)
return
endcase

```

## Forma Replicap.scx

### Command3

#### Evento Click

Verifica el motor de base de datos en la que fue creada la base de datos de la conexión remota

```
private datos1,a
a=sqlconnect()
if a>=1
    =sqltables(a,'TABLE','tablas')
    select tablas
    if len(alltrim((tablas.table_qualifier)))=0
        =messagebox('conexion realizada',0)
        thisform.text1.value=SQLGETPROP(a,'DataSource')
        thisform.refresh
        =sqldisconnect(a)
        o1="P"
    else
        =messagebox('conexion realizada',0)
        thisform.text1.value=SQLGETPROP(a,'DataSource')
        thisform.refresh
        =sqldisconnect(a)
        o1="V"
    endif
else
    =messagebox('conexion fallida',0)
endif
```

### Command4

#### Evento Click

Verifica el motor de base de datos en la que fue creada la base de datos de la conexión remota

```
private datos1,a
a=sqlconnect()
if a>=1
    =sqltables(a,'TABLE','tablas')
    select tablas
    if len(alltrim((tablas.table_qualifier)))=0
        =messagebox('conexion realizada',0)
        thisform.text1.value=SQLGETPROP(a,'DataSource')
        thisform.refresh
        =sqldisconnect(a)
        o1="P"
    else
```

```

        =messagebox('conexion realizada',0)
        thisform.text1.value=SQLGETPROP(a,'DataSource')
        thisform.refresh
        =sqldisconnect(a)
        o1="V"
    endif
else
    =messagebox('conexion fallida',0)
endif

```

## **Command1**

### **Evento Click**

Realiza la replicación de las conexiones seleccionadas

```

do case
    case o1="V" and o2="V"
        do replica with thisform.text1.value,thisform.text2.value
    case o1="V" and o2="P"
        do replica2 with thisform.text1.value,thisform.text2.value
    case o1="P" and o2="V"
        do replica1 with thisform.text1.value,thisform.text2.value
    otherwise
        =messagebox("Tipo de Replicación no Reconocido",64,"Replicación")
endcase
=messagebox("Proceso de Replicacion Finalizado",0)

```

En este evento se utiliza 3 programas: replica el cual realiza la replicación de Visual FoxPro a Visual Foxpro, replaica2 de Visual Foxpro a Postgresql y replica1 de Postgresql a Visual FoxPro.