Componentes de acceso a Interbase/Firebird

Después de que alguno de los colegas del foro me largase una pequeña bronquita por no participar últimamente en ellos, lo cual es totalmente cierto, lo reconozco, me he decidido a publicar lo siguiente:

Existen muchas preguntas en los foros acerca de cual es la manera más eficaz, o la menos mala, de acceder a FB/IB en Delphi, mediante qué componentes, forma, etc. Muchas han sido las respuestas, pero pese a ello, le he metido algo de tiempo al asunto, creando una pequeñita aplicación sobre testeo de los más conocidos o empleados (por lo menos hasta donde yo sé) componentes de acceso a estos sistemas gestores de bases de datos.

Los implicados han sido (todo bajo WINDOWS):

- **S.O.**: Windows 2000 Profesional SP4 (ojo, tengo alguna actualización de Windows Update con SP5), corriendo FB 1.5 (ojo, no está actualizado a la 1.51) como servicio de arranque automático, siendo la misma máquina como cliente y servidor. Se hace notar que he intentado que las diversas pruebas se realizases en las mismas condiciones de máquina, para que los resultados obtenidos en las diversas pruebas se puedan cotejar con más credibilidad.
- **Lenguaje**: DELPHI 6, con SP2.
- **DBEXPRESS**: Exactamente el que acompaña a la versión de Delphi citada.
- **BDE**: Exactamente la que acompaña a la versión de Delphi citada, es decir, la 5.01.
- **ZEOS**: son la versión zeosdbo-6.1.5-stable, aunque si no estoy mal informado ya ha salido la zeosdbo-6.5.0-alpha (http://www.zeoslib.net, http://www.sourceforge.net/projects/zeoslib)
- **IBX**: Exactamente la versión 6.08 (trae soporte para IB 7.01) (http://codecentral.borland.com/codecentral/ccweb.exe/listing?id=20258). Habrá que tener en cuenta el ya NO soporte para FireBird, OJOOOOO.
- **FIBPlus**: exactamente la versión 5.3.0 Release; es una pena que se tenga que pagar licencia por ella, pues como veremos va cañón. De su página se puede descargar la trial/demo, así que para su prueba vale (http://www.devrace.com).
- **UIB**: Unified Interbase v1.2h (2004-05-06) (http://prdownloads.sourceforge.net/synedit/synedit-cvs-2004-04-27.zip)
- **ODBC mediante BDE**: versión Firebird/InterBase(r) driver 1.02.00.64 de fecha 01JUL2004 (http://www.ibphoenix.com).
- **ODBC mediante ADO**: mismo controlador ODBC anterior y ADO en la versión del Delphi y WINDOWS instalada (no sé en este momento la numeración).
- **IBO**: es la versión 11/24/2003 Sub-Release 4.3 Aa, es la trial que se puede descargar de la Web de www.ibobjects.com

Desconozco hasta el momento si existirán otros componentes y similares, es por eso que sólo emplearé estos para el testeo.

Para la prueba he creado una Base da Datos consistente en:

- Dos (2) Tablas
- Dos (2) Disparadores (Trigger).
- Un (1) Procedimiento Almacenado (Store Procedures).
- Un (1) Generador.

La estructura quedaría de la siguiente manera (los metadatos han sido obtenidos mediante la utilidad IBManager de EMS (www.ibmanager.com)):

```
CREATE DATABASE 'C:\testeo\testeo.gdb'
USER 'sysdba' PASSWORD 'la de siempre'
PAGE_SIZE 4096
DEFAULT CHARACTER SET WIN1251;

CREATE GENERATOR SACARID ;

SET GENERATOR SACARID TO 1;

CREATE TABLE TABLA1 (
  ID INTEGER NOT NULL,
  CAMPO1 VARCHAR (50) CHARACTER SET WIN1251 NOT NULL COLLATE WIN1251,
  CAMPO2 VARCHAR (50) CHARACTER SET WIN1251 NOT NULL COLLATE WIN1251
);

CREATE TABLE TABLA2 (
  ID INTEGER NOT NULL,
  VALOR1 VARCHAR (50) CHARACTER SET WIN1251 NOT NULL COLLATE WIN1251
);

ALTER TABLE TABLA1 ADD CONSTRAINT PK_TABLA1 PRIMARY KEY (ID);
ALTER TABLE TABLA2 ADD CONSTRAINT PK_TABLA2 PRIMARY KEY (ID);
ALTER TABLE TABLA2 ADD CONSTRAINT FK_TABLA2 FOREIGN KEY (ID)
    REFERENCES TABLA1 (ID) ON DELETE CASCADE ON UPDATE CASCADE;

CREATE INDEX FK_TABLA2 ON TABLA2 (ID);
CREATE UNIQUE INDEX IDX_TABLA1 ON TABLA1 (CAMPO1);
CREATE UNIQUE INDEX IDX_TABLA11 ON TABLA1 (CAMPO2);
CREATE UNIQUE INDEX IDX_TABLA2 ON TABLA2 (VALOR1);
CREATE UNIQUE INDEX PK_TABLA1 ON TABLA1 (ID);
CREATE UNIQUE INDEX PK_TABLA2 ON TABLA2 (ID);

SET TERM ^ ;
CREATE PROCEDURE METEENTABLA2 (
  VALORDEID INTEGER)
AS
BEGIN
  EXIT;
END
^

ALTER PROCEDURE METEENTABLA2 (
  VALORDEID INTEGER)
AS
  DECLARE VARIABLE IDENSTRING VARCHAR(20);
begin
  IDENSTRING = 'Valor de Id: ' || CAST(:VALORDEID AS varchar(20));
  INSERT INTO tabla2 ( tabla2 . id , tabla2 . valor1 ) VALUES(:VALORDEID, :IDENSTRING )
;

  suspend;
```

```
end
^

CREATE TRIGGER INSERTAENTABLA2 FOR TABLA1 ACTIVE
  AFTER INSERT POSITION 0
AS
begin
  execute procedure meteentabla2(new. id );
end
^

CREATE TRIGGER TABLA1OPTENERID FOR TABLA1 ACTIVE
  BEFORE INSERT POSITION 0
AS
BEGIN
  IF (new. id IS NULL) then

  new. id = gen_id( sacarid , 1);
END
^
SET TERM ; ^
```

Si observamos ambas tablas están relacionados entre sí por la *foreign key* creada, de tal manera, que el proceso es el siguiente:

Cuando introducimos un registro o fila en la TABLA1, antes de ello salta el disparador de tal manera que su ID, el de la *PrimaryKey*, se obtiene mediante un Generador. Posteriormente al darse el alta, salta otro disparador, el cual ejecuta el procedimiento almacenado que desencadena otra alta en la TABLA2 relacionada con esta TABLA1.

Con ello lo que he pretendido ha sido que el Server (en este caso de FB 1.5) ejecute en cierta manera las diferentes posibilidades con las que cuenta, Tablas, Generadores, Trigger y StoreProcedures. Tal vez también debería haber tenido en cuenta alguna UDF, View, Excepcion, pero creo que para valorar un poco el asunto, puede valer.

Una vez creada la BD, procedo a crear una pequeñita aplicación que consiste en lo siguiente:

Provoco un bucle indefinido hasta 250.000, con lo que pretendo que la aplicación genere estas 250.000 altas en la TABLA1, provocando los consiguientes procesos comentados más arriba. Se hace notar que cada 500 altas, provoco un Commit en la BD.

El código de dicha aplicación para el caso de ODBC mediante ADO es el siguiente:

```
procedure TForm1.ButtonComenzarClick(Sender: TObject);
var
  i: Integer;
  j: Integer;
  tInicio: TDateTime;
  tFinal: TDateTime;
  tSegundosInicio: Integer;
  tSegundosFinal: Integer;
begin
  j := 1;
  ADOConnection1.Close;
  ADOConnection1.Open;
  ADOConnection1.BeginTrans;
  tInicio := Now;
  tSegundosInicio := (StrToInt(FormatDateTime('hh', tInicio)) * 3600) +
    (StrToInt(FormatDateTime('nn', tInicio)) * 60) +
    (StrToInt(FormatDateTime('ss', tInicio)) * 1) ;
  LabelInicio.Caption := 'Hora Inicio: ' + DateTimeToStr(Now);
```

```
  for i := 1 to 250000 do
  begin
    if j >= 500 then
    begin
      ADOConnection1.CommitTrans;
      ADOConnection1.BeginTrans;
      j := 1;
    end;

    with ADOQuery1 do
    begin
      Close;
      Parameters.ParamByName('tCampo1').Value := 'Valor de: ' + IntToStr(i);
      Parameters.ParamByName('tCampo2').Value := 'Valor de: ' + IntToStr(i);
      ExecSQL;
      Close;
    end;
  end;

  ADOConnection1.CommitTrans;
  tFinal := Now;
  tSegundosFinal := (StrToInt(FormatDateTime('hh', tFinal)) * 3600) +
    (StrToInt(FormatDateTime('nn', tFinal)) * 60) +
    (StrToInt(FormatDateTime('ss', tFinal)) * 1) ;
  LabelFinal.Caption := 'Hora Final: ' + DateTimeToStr(Now);
  LabelInvertido.Caption := 'Segundos Invertidos: ' +
  IntToStr(tSegundosFinal - tSegundosInicio);
end;
```

Este código ha sido a su vez adaptado a la forma de trabajar de cada componente. En este caso me he decidido por postear el ADO, porque casi nadie lo menciona (e imagino que tampoco utiliza), que pese a emplear el ODBC correspondiente y similar al BDE, es otra alternativa parecida (y sin BDE, claro, lo cual puede ser una ventaja), aunque como luego veremos, tal vez no sea la más ……, en fin, aquí queda.

Cada aplicación sobre componentes diversos, los he lanzado 3 veces cada uno en condiciones de máquina similar (recién arrancada, para ser lo más parecidas posible, caralladas del WIN aparte). Una vez obtenidos los tiempos de proceso, he hecho una simple media aritmética y los resultados han sido los siguientes (expresados, claro, en segundos):

Componente	Tiempo (sg.)
DBEXpress	80
BDE	148
ZEOS	309
IBX	64
FIBPlus	64
UIB	86
BDE/ODBC	195
ADO/ODBC	211
IBO	75

Cabe citar que en la configuración de los ZEOS, he tenido que emplear la configuración para FB 1.0, pues la que traía para 1.5 me daba problemas. Me imagino que en la versión actualizada que comenté arriba lo habrán solucionada. Lo desconozco.

En base a ello, estos han sido los datos que he podido obtener.

Espero, si tal, que ello sirva para determinar en un momento dado, cual es la mejor opción ha tener en cuenta a la hora de buscar rapidez para este tipo de proceso de altas masivas en automático. Lo ideal hubiese sido poder realizar en una red de varios equipos ejecutándola simultáneamente, pero eso es lo que tengo por casa, y por desgracia ya no estoy currando para disponer de una red a mi gusto.

Así mismo, si un día me decido, probaré los mismos a la hora de realizar unos SELECT selectivos y un poco complicadillos, pues a lo mejor la cosa puede varias con respecto a la actual.

Otra cuestión es qué es mejor como Server FB, en Windows o Linux??? La cosa está clara…… pero cuanto más clara????, Eso lo dejo para otro artículo. Búscalo, puede valer la pena que pierdas un rato leyéndolo.

Julio Nogueira Fandiño
CombatF2D
07 de octubre de 2004