Updating Static Queries and Using Cached Updates in Delphi C++Builder Kylix

It is sometimes necessary to provide an updateable dataset that is based on a static query (which is by definition a read-only dataset). Borland Delphi, C++Builder, and Kylix can provide this functionality through the use of a TClientDataSet component.

This same functionality can also be used as a replacement for the BDE’s cached update feature.

The result set from a TAdsQuery component is first loaded into an in-memory table (TClientDataSet). Any and all data-aware components can then be configured to get their data from the TClientDataSet, instead of from the original TAdsQuery component.

After your data-aware controls are pointing to the TClientDataSet, all data modifications are internally stored by the TClientDataSet, and can be posted at any time using the TClientDataSet.ApplyUpdates method. If you’d prefer each row was posted immediately, the TClientDataSet.AfterPost method can be used to automatically call TClientDataSet.ApplyUpdates.

Next, simply define a BeforeUpdateRecord event for your TDataSetProvider. The BeforeUpdateRecord event is passed a delta dataset, which contains both the new field values and the old field values for the in-memory dataset. These values can be used in individual update operations to each of the base tables involved in the query. The BeforeUpdateRecord event then sets the return parameter Applied to true, indicating that it has applied the update, and no other actions are necessary.

Example

The following example is a BeforeUpdateRecord event that posts changes made to an original static query of:

SELECT a.empid, a.lastname, b.firstname FROM table1 a, table2 b WHERE a.empid = b.empid

A full example project (for Delphi 6 and newer) can be downloaded from Code Central on the Devzone.advantagedatabase.com zone.
const UpdateStmt1 = 'UPDATE table1 SET lastname = "%s" WHERE empid = %s';
DeleteStmt1 = 'DELETE FROM table1 WHERE empid = %s';
InsertStmt1 = 'INSERT INTO table1 VALUES ( %s, "%s" )';
UpdateStmt2 = 'UPDATE table2 SET firstname = "%s" WHERE empid = %s';
DeleteStmt2 = 'DELETE FROM table2 WHERE empid = %s';
InsertStmt2 = 'INSERT INTO table2 VALUES ( %s, "%s" )';

procedure TForm1.DataSetProvider1BeforeUpdateRecord(Sender: TObject; SourceDS: TDataSet; DeltaDS: TCustomClientDataSet; UpdateKind: TUpdateKind; var Applied: Boolean);
begin
  var SQL: string;
  begin
    // NOTE: If converting an app that used to use CachedUpdates, this is a good
    // place to use the TUpdateSQL components you already have (and more importantly,
    // the sql inside them). See the Join Updates section of the following Borland
    // article for details:
    // http://bdn.borland.com/article/0,1410,20567,00.html
    //
    // For this example, we'll use the global statements declared at the top of
    // this unit (UpdateStmt1, DeleteStmt1, etc.).
    case UpdateKind of
      ukModify:
        begin
          SQL := Format(UpdateStmt1, [DeltaDS.Fields[1].NewValue, DeltaDS.Fields[0].OldValue]);
          AdsConnection1.Execute(SQL);
        end;
      ukDelete:
        begin
          SQL := Format(DeleteStmt1, [DeltaDS.Fields[0].OldValue]);
          AdsConnection1.Execute(SQL);
        end;
      ukInsert:
        begin
          SQL := Format(InsertStmt1, [DeltaDS.Fields[0].NewValue, DeltaDS.Fields[1].NewValue]);
          AdsConnection1.Execute(SQL);
          SQL := Format(InsertStmt2, [DeltaDS.Fields[0].NewValue, DeltaDS.Fields[2].NewValue]);
          AdsConnection1.Execute(SQL);
        end;
    end;
    Applied := True;
  end;
end;

References

All content in this tech tip was derived from Dan Miser's Borland Community article titled ClientDataSet as a Replacement for Cached Updates and the Delphi help files.

The ClientDataSet as a Replacement for Cached Updates article can be found online at http://bdn.borland.com/article/0,1410,20567,00.html

Cary Jensen has also written a very comprehensive series of articles describing the TClientDataSet component and its features. The first article in this series can be found on the Borland Community web site at http://bdn.borland.com/article/0,1410,28876,00.html