Borland Delphi

Curso Intermediário

Módulo 2

Instrutor

Jackson Pires de O. S. Júnior

Jackson_pires@yahoo.com.br

Desenvolvendo Aplicações com Banco de Dados

Módulo 2

Instrutor

Jackson Pires de O. S. Júnior

Jackson_pires@yahoo.com.br

Capítulo

1

Introdução

O *Módulo Desenvolvendo Aplicações com Banco de Dados* apresentará conceitos e habilidades necessárias para criar Aplicações com Banco de Dados.

NOTA: Você precisará da versão Enterprise ou Professional para poder desenvolver aplicações com Banco de Dados. Para Implementar Banco de Dados Cliente\Servidor mais avançados você precisará de algumas características encontrada apenas na versão Enterprise.

Criando Aplicações com Banco de Dados (Definição)

Aplicações com Banco de Dados permitem que os usuários interajam com informações que estão guardadas em Banco de dados. Os Banco de Dados são providos de informações estruturadas, e são compartilhados entre várias aplicações.

O Delphi provê suporte a Aplicações com Banco de Dados Relacionais. Banco de Dados Relacionais organizam informações dentro de tabelas que contém linhas (Rows - Records) e colunas (Columns – Fields). Estas tabelas podem ser manipuladas com simples operações conhecidas como cálculos relacionais.

Quando fazemos o design de uma aplicação de Banco de Dados, você tem por obrigação de como os dados estão estruturados. Baseado na estrutura, você pode criar uma interface para mostrar os dados para um usuário reservado que poderá inserir ou modificar um dado existente.

Este Capítulo faz uma introdução e algumas considerações para se criar uma Aplicação de Banco de Dados bem como decisões envolvidas na criação da Interface com o Usuário.

Usando Banco de Dados

O Delphi inclui muitos componentes para acesso a Banco de Dados e para representar os dados contidos neles. Eles estão agrupados concordando com um mecanismo de acesso a dados.

- Paleta BDE
- Paleta ADO
- Paleta dbExpress
- Paleta Data Access
- Paleta Iterbase

NOTA: Diferentes versões do Delphi incluem diferentes Drivers para acessar Banco de Dados de Servidores usando BDE, ADO ou dbExpress.

Quando estamos criando uma aplicação que envolve Banco de Dados temos a obrigação de escolher os componentes que vamos usar. Cada um dos mecanismos de acesso a dados são diferenciados pela sua escala de capacidade de suportar funcionalidades de acesso a dados. Uma delas é a facilidade de distribuição e a viabilidade de suportar diferentes tipos de Servidores de Banco de Dados.

Ao escolher um mecanismo de acesso a dados, você tem a obrigação de escolher um tipo de Servidor de Banco de Dados. Existem diversos tipos diferentes de Servidores de Banco de Dados, e você deve considerar as vantagens e desvantagens de cada um antes de escolher o seu Servidor de Banco de Dados Pessoal.

Todos os tipos de Banco de Dados contém tabelas que guardam informações. Em alguns casos (mas não em todos) servidores suportam:

- Segurança em Banco de Dados.
- Transações
- Integridade Referencial, Triggers e Store Procedures.

Tipos de Bancos de Dados

Servidores de Banco de Dados Relacionais variam na maneira em que as informações são armazenadas e na maneira em que elas são acessadas por vários usuários. O Delphi provê acesso a dois tipos de acesso à Servidores de Banco de Dados Relacionais.

- **Servidores de banco de Dados Remoto** residem um uma máquina separada. As vezes os dados provem de servidores de banco de dados remotos que nem sempre é única máquina, na maioria das vezes está distribuída em alguns servidores. Embora os servidores de banco de dados variem a forma de armazenar as informações, todos eles fornecem uma relação lógica comum aos clientes. È comum interface baseda em (Structured Query Language – SQL) . Por causa desse tipo de acesso usando SQL estes servidores também são chamados de SQL Servers. Exemplos de Banco de Dados Remoto que usam servidores SQL são InterBase, Oracle, Sybase, Informix, Microsoft SQL server, e DB2.

- **Banco de Dados Locais** residem em um drive local ou em uma LAN (Local Area Network). Frequentemente é usada API's para acessar os dados. Quando são compartilhados por diversos usuários, usam mecanismos travamento baseados em arquivos. Por causa disso, às vezes são chamados de banco de dados baseados em arquivos. Exemplos de banco de dados locais são Paradox, dBASE, FoxPro, and Access.

Aplicações que usam banco de dados são chamados de "Aplicações de camada simples" porquê a aplicação compartilha apenas um sistema de arquivos. Aplicações que usam Banco de Dados Remoto são chamados de "Aplicações de duas camadas" ou "Aplicações de múltiplas camadas" porque a aplicação e o banco de dados operam independente do sistema (ou camada).

Para escolher um tipo de Banco de Dados para usar dependerá de alguns fatores. Por exemplo, seus dados já podem estar armazenados em um banco de dados já existente. Se você tiver que criar o banco de dados e suas tabelas você terá que considerar as seguintes questões.

- **Quantos usuários estão compartilhando as tabelas?** Servidores de Banco de Dados Remoto são designados para o acesso por diversos usuários ao mesmo tempo. Fornecem a sustentação para múltiplos usuários através de um mecanismo chamado transações. Algumas bases de dados locais (tais como interBase local) fornecem também a sustentação

da transação, mas muitas fornecem somente mecanismos de travamento de arquivos, e algumas não fornecem nenhuma sustentação multiusuário.

- **Qual a quantidade de dados que as tabelas armazenarão?** Os usuários remotos da base de dados podem aramazenar mais dados do que bases de dados locais. Alguns usuários remotos da base de dados estão projetados para quantidades de dados de grandes armazenando quando outros estão otimizados para outros critérios (tais como updates rápidos).

- **Que tipo de desempenho (velocidade) você requer da base de dados?** As bases de dados locais são geralmente mais rápidas do que usuários remotos da base de dados porque residem no mesmo sistema que a aplicação da base de dados. Os usuários remotos diferentes da base de dados otimizadas podem suportar tipos diferentes de operações, assim se você pode quiser considerar o desempenho ao escolher um usuário remoto da base de dados.

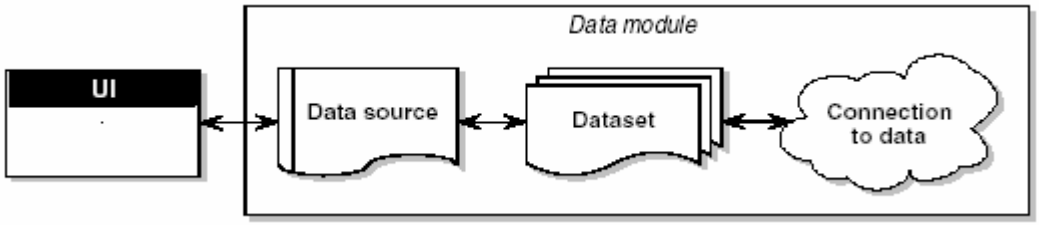
- **Que tipo de suporte estará disponível para o adminstrador da base de dados?** As bases de dados locais requerem menos suporte do que usuários remotos da base de dados. Tipicamente, são menos cara operar-se porque não requerem usuários separados instalados ou licenças caras.

Arquitetura de Banco de Dados

As aplicações da base de dados são construídas com elementos de interface de usuários, os componentes que representam a informação da base de dados (Datasets), e componentes que conectam estes a à fonte da informação da base de dados. Como você organiza estas partes é a arquitetura de sua aplicação da base de dados.

Estrutura Geral

Quando houver muitas maneiras distintas organizar os componentes em uma aplicação da base de dados, siga mais o esquema geral ilustrado na figura:



O Formulário de Interface com o Usuário

É uma boa idéia isolar a relação de usuário em um formulário que seja completamente separado do resto da aplicação. Isto tem diversas vantagens. Isolando a relação de usuário dos componentes que representam a informação própria da base de dados, você introduz uma maior flexibilidade em seu projeto: Mudanças na maneira que você controla a informação da base de dados não o requer reescrever sua interface de usuário, e as mudanças na interface de usuário não o requerem mudar a parte de sua aplicação que trabalha com a base de dados. Além disso a isolação deixa-o desenvolver os formulários comuns de que você pode compartilhar entre aplicações múltiplas, fornecendo desse modo uma interface de usuário

consistente. Para armazenar links aos formulários exemplo no repositório do objeto, você e outros colaboradores podem construir em fundações existentes, melhor do que começar sobre o risco para cada projeto novo. Compartilhar forms também faz ele possível para que você desenvolva padrões incorporados para relações da aplicação.

O Data Module

Se você isolar sua interface de usuário em seu próprio formulário, você pode usar um módulo de dados para abrigar os componentes que representam a informação da base de dados (Datasets), e os componentes que conectam estas Datasets de dados às outras partes de sua aplicação. Como os formulários da interface de usuário, você pode compartilhar dos módulos de dados no repositório do objeto de modo que possam reúso ou compartilhado entre aplicações.

O Data Source

O primeiro artigo no módulo de dados é uma origem dos dados (DataSource). O DataSource age como uma canalização entre a relação de usuário e uma série de dados que represente a informação de uma base de dados. Diversos controles (data-ware) em um formulário podem compartilhar de uma única origem dos dados (DataSource), que no caso a exposição em cada controle é sincronizada de modo que com os scrolls (Rolagens) do usuário através dos registros, o valor correspondente nos campos para o registro atual são indicados em cada controle.

O Dataset

O coração de sua aplicação da base de dados é a Dataset. Este componente representa um jogo dos registros da base de dados subjacente. Estes registros podem ser os dados de uma única tabela da base de dados, de um subconjunto dos campos ou dos registros em uma tabela, ou da informação de mais de uma tabela juntada em uma única vista. Usando DataSets, sua lógica da aplicação é protegida de reestruturamento das tabelas físicas em suas bases de dados. Quando a base de dados subjacente muda, você pôde necessitar alterar a maneira que o componente Dataset especifica os dados contidos, mas o resto de sua aplicação pode continuar a trabalhar sem alteração.

A Conexão com os Dados

Os tipos diferentes de DataSets usam mecanismos diferentes para conectar à informação subjacente da base de dados. Estes mecanismos diferentes, por sua vez, fazem acima as diferenças principais na arquitetura das aplicações que da base de dados você pode construir. Há quatro mecanismos básicos para conectar aos dados:

- Conectar diretamente a um usuário da base de dados. A maioria de séries de dados usam um descendente de *TcustomConnection* para representar a conexão com um Servidor de Banco de Dados.

- Usando um arquivo dedicado no disco. As DataSets do cliente suportam a abilidade ao trabalho com um arquivo dedicado no disco. Nenhum componente separado da conexão é necessário ao trabalhar com um arquivo dedicado porque a Dataset do própria cliente sabe ler de e escrever o arquivo.

- Conectar a uma outra DataSet. As DataSets do cliente podem trabalhar com os dados fornecidos por uma outra DataSets. Um componente de TDataSetProvider serve como um intermediador entre a DataSet do cliente e sua DataSet da fonte. Este DatasetProvider pode residir no mesmo módulo de dados que a Dataset do cliente, ou pode ser peça de um usuário da aplicação que funciona em uma outra máquina. Se o Provider for parte de um usuário da aplicação, você necessitará também um descendente especial de TCustomConnection para representar a conexão ao usuário da aplicação.

- Obtendo dados de um objeto de RDS DataSpace. As Datasets do ADO podem usar um componente de TRDSConnection nas aplicações multi-tier (multicamadas) da base de dados que são construídas usando usuários baseados em aplicação ADO.

Usando características comuns de controle de dados

As seguintes tarefas são comuns para muitos contoles de dados.

- Associação de controles com DataSets
- Edição e atualização de dados
- Desabilitar e habilitar a visualização dos dados
- Atualização do dados mostrados
- Ativar mouse, teclado e eventos temporizados

Os controles de dados deixaram-no indicar e editar campos dos dados associados com o registro atual em uma série de dados. A tabela abaixo mostra os controles de dados que aparecem na paleta Data Controls.

Componente	Descrição
TDBGrid	Exibe informações de um DataSource em um formato tabular. As colunas na grade correspondem às colunas na série de dados subjacente da tabela ou dos querys. As fileiras na grade correspondem aos registros
TDBNavigator	Navegue através dos registros de dados ou em uma série de dados atualizando registros, afixando registros, suprimindo registros, cancelando, editando os registros, e atualizando a exposição dos dados.
TDBText	Exibe os dados de um campo em um Label.
TDBEdit	Exibe os dados de um campo em um Edit Box.
TDBMemo	Exibe os dados de um campo Memo ou BLOB em uma tabela com barra de rolagem (Multiplas linhas).
TDBImage	Exibe gráficos de um campo em um graphic box.
TDBListBox	Exibe uma lista de items o qual pode atualizar um dado em um determindao registro.
TDBComboBox	Exibe uma lista de items no qual podemos atualizar um campo, e também entrada direta de texto como num edit box.
TDBCheckBox	Exibe um CheckBox para indicar um valor booleano.
TDBRadioGroup	Exibe opções de escolha exclusiva para um campo.
TDBLookUpListBox	Exibe uma lista de items vindos de um outro dataset baseados no valor do campo.
TDBLookUpComboBox	Exibe uma lista de items vindos de um outro dataset baseados no valor do campo e também entrada direta de texto como num edit box.
TDBCtrlGrid	Exibe um configurável e repetido controle data-aware como em uma grade.
TDBRichEdit	Exibe dados formatados de um campo em um edit box.