



Delphi e Object Pascal

Autor:
Maurício Capobianco Lopes



**UNIVERSIDADE REGIONAL DE BLUMENAU
CENTRO DE CIÊNCIAS EXATAS E NATURAIS
DEPARTAMENTO DE SISTEMAS E COMPUTAÇÃO**

**DISCIPLINA: Linguagem de Programação Visual
PROFESSOR: Maurício Capobianco Lopes**

SUMÁRIO

Banco de Dados

 DataBase Desktop

 Paleta DATA ACCESS

 Paleta DATA CONTROLS

 Data Module

 Exemplo da Manipulação de um Banco de Dados

 Form Expert

 Bancos de Dados Relacionais

 Chaves de Acesso

 Exemplo de um Banco de Dados Relacional

Impressão de Relatórios

 Paleta QReport

 Exemplo de Uso do Quick Report

BANCO DE DADOS

Os **bancos de dados** são utilizados para o armazenamento de dados. O DELPHI pode gerenciar vários tipos de banco de dados, entre eles: **DBase**, **Paradox** e **Interbase**, e outros como **Oracle**, **Sybase**, **Microsoft SQL Server** e **Informix**, através de **ODBC** (*Open Database Connectivity*).

As diferentes edições do DELPHI (**Desktop**, **Developer** e **Client Server Suite**) diferem principalmente nos recursos de gerenciamento de banco de dados oferecidos.

As principais ferramentas de gerenciamento de banco do DELPHI são:

- **DataBase Desktop (DBD)**: ferramenta para criar, alterar e editar bancos de dados;
- **Componentes** (visuais e não visuais): permitem conexões e interface com banco de dados. Estão disponibilizados nas paletas **Data Access** e **Data Controls**;
- **Data Module**: centraliza componentes de acesso a dados dos formulários e aplicações;
- **Form Expert**: permite a geração automática de formulários-padrão para acesso a banco de dados.
- **Object Repository**: armazena formulários e módulos de dados para serem compartilhados entre diferentes aplicações;
- **Borland Database Engine (BDE)**: faz o relacionamento entre o DELPHI e o banco de dados;
- **SQL Explorer**: ferramenta para gerenciar BDE e criar dicionário de dados;
- **Quick Report**: permite a geração de relatórios a partir de bancos de dados;

DATABASE DESKTOP

O **DataBase Desktop** é uma ferramenta do DELPHI para manipulação de Banco de Dados. Ela é útil principalmente para:

- criação de banco de dados;
- alteração das características dos campos.

EXEMPLO: criar uma tabela contendo os funcionários de uma empresa.

Para ilustrar o funcionamento do *DataBase Desktop* será criada uma tabela de produtos com os seguintes campos: CÓDIGO, NOME DA PESSOA, IDADE, SEXO, ENDERECO, CIDADE, UF e SALÁRIO. Para isto devem ser seguidos os passos abaixo:

PASSO 1

Clique na opção **Tools** no menu principal do DELPHI, e, em seguida, na opção **Database Desktop**. Neste momento será aberto o programa *Database Desktop*.

PASSO 2

No menu principal do *Database Desktop* selecione a opção **File**, seguida de **New** e, por fim, a opção **Table** (Figura **Erro! Argumento de opção desconhecido.**).

PASSO 3

O próximo diálogo permite que se selecione o tipo da tabela a ser criada. Selecione a opção **Paradox 7** e clique em **OK**. As tabelas *Paradox* tem extensão (DB) e oferecem mais recursos que as tabelas *DBase* (extensão DBF).

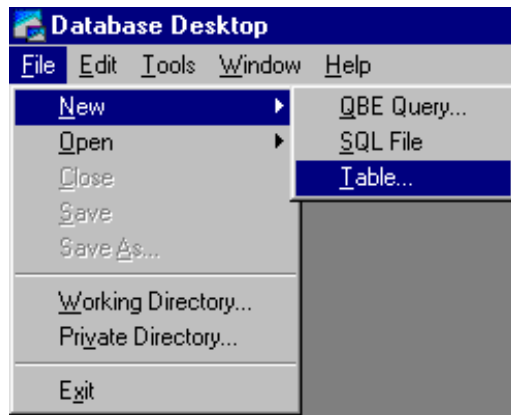


Figura Erro! Argumento de opção desconhecido.: *Database Desktop*: criando uma nova tabela.

PASSO 4

Neste instante devem ser definidos os atributos dos campos, como segue (Figura Erro! Argumento de opção desconhecido.):

- **Field Name**: define o nome do campo;
- **Type**: define o tipo do campo. Clicando com o botão direito do mouse pode-se selecionar o tipo do campo;
- **Size**: define o tamanho do campo. Só está disponível para alguns tipos;
- **Key**: define campos-chave para bancos de dados relacionais.

Ainda nesta tela pode-se definir:

- **Table properties**: define características gerais dos campos da tabela;
- **Required Field**: define se o campo tem um valor requerido;
- **Minimum Value**: valor mínimo para o campo;
- **Maximum Value**: valor máximo para o campo;
- **Default Value**: valor default para o campo;
- **Picture**: define o tipo de campos figura.

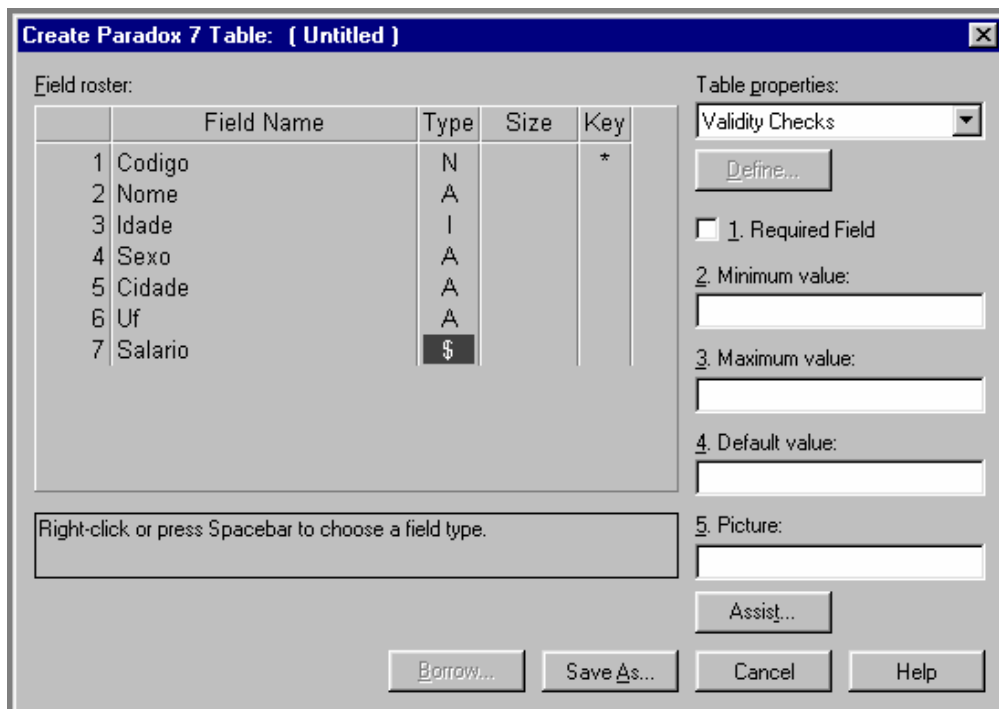


Figura Erro! Argumento de opção desconhecido.: *Database Desktop*: definindo os atributos dos campos

PASSO 5

Clicando no botão **Save As**, deve aparecer uma caixa de diálogo para que se possa salvar a tabela em disco. Dê um nome a tabela e clique no botão **Save**.

☞ Verifique com atenção o diretório onde está gravando a tabela.

Neste instante a tabela foi criada. Para verificar se ela foi criada corretamente pode-se selecionar **File | Open | Table** no menu principal do *Database Desktop*, para poder abrir o arquivo gravado.

Para manipular este banco de dados deve-se retornar ao DELPHI. Inicialmente deve ser criado um formulário que conterá os campos do banco de dados para edição. A criação de um formulário pode ser feita de duas formas: automática (através do **Form Expert**) ou manual (através das paletas de componentes **Data Access** e **Data Controls**).








PALETA DATA ACCESS

A paleta Data Access (Figura **Erro! Argumento de opção desconhecido.**) contém 9 componentes que permitem a interface entre o DELPHI e os bancos de dados.



Figura Erro! Argumento de opção desconhecido.: Paleta Data Access: componentes de acesso a banco de dados

Os componentes desta paleta são:

 DataSource	Permite a interface entre os componentes de acesso e de visualização de dados.
 Table	Permite inserir uma tabela na aplicação.
 Query	Permite realizar consultas especiais sobre as tabelas, através da linguagem SQL.
 StoredProc	Permite executar <i>procedures</i> armazenadas em um servidor de banco de dados.
 Database	Permite agrupar tabelas para ser utilizada em uma aplicação.
 Session	Permite controlar as conexões da aplicação com o banco de dados.
 BatchMove	Permite trazer dados de um banco de dados remoto para serem tratados localmente.



Permite executar operações de modificação, inserção e deleção de registros em tabelas através da linguagem SQL.



Permite a interface da aplicação com o Report Smith, que é um gerador de relatórios do DELPHI.

Os componentes são manipulados através de suas propriedades, métodos e eventos de banco de dados.

PALETA DATA CONTROLS

A paleta Data Controls (Figura **Erro! Argumento de opção desconhecido.**) contém 13 componentes que permitem a manipulação e visualização dos dados de um banco de dados.



Figura Erro! Argumento de opção desconhecido.: Paleta Data Controls: componentes de controles de banco de dados

Os componentes desta paleta são:



DBGrid

Permite manipular os dados em forma de uma matriz.



DBNavigator

Permite controlar a posição dos registros em um banco de dados, permitindo fazer diversas operações sobre o registro, como inserir, apagar, alterar, gravar, etc.



DBText

Permite a visualização do conteúdo de um campo, sem a possibilidade de alterá-lo.



DBEdit

Permite visualizar/editar um campo do banco de dados.



DBMemo

Permite visualizar/editar dados de um campo Memo.



DBImage

Permite visualizar/editar dados de um campo de imagem.



DBListBox

Permite visualizar/editar dados através de uma *List Box*.



DBComboBox

Permite visualizar/editar dados através de uma *Combo Box*.



DBCheckBox

Permite visualizar/editar dados através de uma *Check Box*.



DBRadioGroup

Permite visualizar/editar dados através de uma *Radio Buttons*.



DBLookupListBox

Permite visualizar/editar dados através de uma *List Box*, utilizando, como opções, um outro conjunto de dados.



DBLookupComboBox

Permite visualizar/editar dados através de uma *Combo Box*, utilizando, como opções, um outro conjunto de dados.



DBCtrlGrid

Semelhante ao TDBGrid, mas permite que se defina uma forma própria de apresentação dos dados.

DATA MODULE

O **Data Module** (Figura **Erro! Argumento de opção desconhecido.**) é uma ferramenta que permite centralizar os componentes de controle de banco de dados. Suas principais vantagens são:

- permitem que toda a aplicação utilize os mesmo componentes de controle de dados, evitando duplicação de componentes;
- garantem o uso correto das tabelas, pois elas não precisam ser recriadas em cada formulário;
- permitem definir novos eventos ou métodos sobre as tabelas que sejam comuns em toda a aplicação, através do **Code Editor** (editor de códigos).

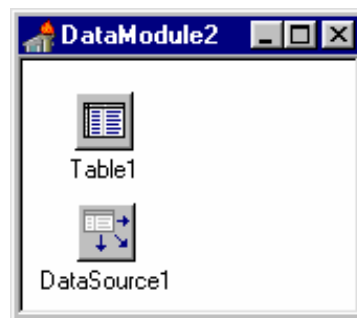


Figura **Erro! Argumento de opção desconhecido.:** *Data Module*: contém componentes de acesso a banco de dados.

EXEMPLO DA MANIPULAÇÃO DE UM BANCO DE DADOS

A criação de um formulário para a manipulação do banco de dados criado no exemplo demonstrado no **Database Desktop** pode ser feita de forma “manual” (adicionando-se cada componente individualmente) ou de forma automatizada (através do **Form Expert**). Neste exemplo mostra-se o modo “manual” de criação do formulário.

PASSO 1

Inicie um novo projeto, clicando na opção **File** no menu principal do DELPHI, e, em seguida, na opção **New Application**. Neste momento será aberto um novo projeto.

PASSO 2

Selecione a paleta de componentes **Data Access**. Clique no componente **Table** e o insira no formulário. Clique também no componente **DataSource** e o insira no formulário.

PASSO 3

Selecione o componente **Table**, alterando as seguintes **propriedades**:

- **Name**: coloque o nome **TabelaFuncionario**.
- **DatabaseName**: nesta propriedade deve ser colocado a localização do banco de dados. Para esta aplicação, coloque o diretório ou o *alias* onde foi criada a tabela do exemplo apresentado no *Database Desktop*.
- **TableName**: nesta propriedade deve ser colocado o nome da tabela. Para esta aplicação, coloque o nome da tabela criada no exemplo apresentado no *Database Desktop*.

PASSO 4

Selecione o componente **DataSource**, alterando as seguintes **propriedades**:

- **Name**: coloque o nome **FonteFuncionario**.
- **DataSet**: nesta propriedade deve ser colocado a tabela que contém os dados.

PASSO 5

Neste momento devem ser colocados os componentes de acesso ao banco de dados, no formulário. Os componentes estão situados na paleta **Data Controls**. Para colocar os componentes siga os passos abaixo:

1. Coloque o componente **DBNavigator** alterando as seguintes **propriedades**:
 - **Name**: coloque o nome **NavegaFuncionario**.
 - **DataSource**: esta propriedade deve conter o componente que tem a fonte dos dados. No caso selecione **FonteFuncionario**.
 - **VisibleButtons**: esta propriedade permite selecionar os botões que ficarão visíveis. Para o exemplo mantenha todos selecionados.
2. Coloque cinco componentes **DBEdit**, com seus respectivos componentes **Label**, para os campos Código, Nome, Idade, Cidade e Salário, alterando as propriedades:
 - **Name**: coloque um nome significativo para cada **DBEdit**.
 - **DataSource**: esta propriedade deve conter o componente que tem a fonte dos dados. No caso selecione **FonteFuncionario**.
 - **DataField**: esta propriedade deve conter o nome do campo ao qual o componente está ligado na tabela.
3. Coloque um componente **DBRadioGroup** para o campo Sexo, alterando as seguintes propriedades:
 - **Name**: coloque um nome significativo para o componente.
 - **DataSource**: esta propriedade deve conter o componente que tem a fonte dos dados. No caso selecione **FonteFuncionario**.
 - **DataField**: esta propriedade deve conter o nome do campo ao qual o componente está ligado na tabela.
 - **Items**: esta propriedade deve conter as opções de itens para o **RadioGroup**. Coloque as opções *Masculino* e *Feminino*.
4. Coloque um componente **DBComboBox** para o campo UF, alterando as seguintes propriedades:

- **Name**: coloque um nome significativo para o componente.
- **DataSource**: esta propriedade deve conter o componente que tem a fonte dos dados. No caso selecione **FonteFuncionario**.
- **DataField**: esta propriedade deve conter o nome do campo ao qual o componente está ligado na tabela.
- **Items**: esta propriedade deve conter as opções de itens para o **ComboBox**. Coloque o nome de estados.

PASSO 6

Neste momento está criado o formulário. Para que a aplicação funcione, no entanto, é necessário **abrir a tabela**. Para isto faça o seguinte:

1. Selecione o componente **TabelaFuncionário**.
2. Mude sua propriedade **Active** para **True**.

Seguidos os passos acima, a aplicação está pronta para ser executada.

FORM EXPERT

O **Form Expert** é uma ferramenta do DELPHI que permite criar um formulário padrão para a manipulação de banco de dados. Sua principal **vantagem** está na automatização do processo de criação de formulário, o que permite ganhar tempo no desenvolvimento da aplicação.

EXEMPLO: neste exemplo será criado um formulário para manipular os dados da tabela criada no exemplo apresentado no [Database Desktop](#).

PASSO 1

Inicie um novo projeto, clicando na opção **File** no menu principal do DELPHI, e, em seguida, na opção **New Application**. Neste momento será aberto um novo projeto.

PASSO 2

Clique na opção **Database** no menu principal do DELPHI, e, em seguida, na opção **Form Expert**. Neste momento será aberto uma tela para iniciar o processo de criação do formulário.

PASSO 3

Na tela inicial do **Form Expert** deve-se selecionar o tipo do banco de dados e o tipo do conjunto de dados. Os tipos de banco de dados são:

- **Simple**: banco de dados simples, onde todos os dados estão armazenados em uma mesma tabela.
- **Master/Detail**: banco de dados relacional

Já o tipo de conjunto de dados pode ser:

- **TTable**: quando os dados forem manipuladas diretamente na tabela.
- **TQuery**: quando os dados forem manipulados a partir de uma consulta.

Para o **exemplo** apresentado selecione **Simple Form**, **TTable** e em seguida clique no botão **Next**. Neste instante deverá aparecer a tela para seleção do banco de dados.

PASSO 4

Esta tela permite que se selecione a tabela onde estão armazenados os dados. Selecione a tabela criada no exemplo do **Database Desktop** e pressione o botão **Next**. Neste instante deverá aparecer a tela de seleção dos campos a serem editados.

PASSO 5

Esta tela permite selecionar os campos que poderão ser editados. Todos os campos que estiverem na coluna **Ordered Selected Fields** poderão ser manipulados no

formulário. Selecione todos os campos (botão >>) e clique em **Next**. Neste instante deverá aparecer a tela de layout do formulário.

PASSO 6

Esta tela permite selecionar a forma de apresentação dos dados, que poderá ser na Vertical, Horizontal, ou em linhas de grade. Selecione a opção **Horizontal** e clique em **Next**. Neste instante deverá aparecer a tela final de criação do formulário.

PASSO 7

Na última tela da criação do formulário deve-se definir se o formulário é o formulário principal da aplicação e se na geração do formulário deve ser criado ou não um **Data Module**. Ative a opção **Generate main form** e selecione a opção **Form and Data Module** e clique no botão **Finish**. Neste instante deve aparecer na tela o formulário criado pelo *Form Expert*, de acordo com as definições selecionadas (Figura **Erro! Argumento de opção desconhecido.**).

Figura Erro! Argumento de opção desconhecido.: Formulário criado com o *Form Expert*.

BANCOS DE DADOS RELACIONAIS

Existem situações no projeto de um banco de dados, onde é necessário criar relações entre tabelas, visando principalmente **evitar inconsistência e duplicação** de informações. O modo mais comum de se realizar esta tarefa é através dos **bancos de dados relacionais**.

Os **bancos de dados relacionais** permitem o relacionamento entre as tabelas através da definição de **chaves de acesso**.

Os bancos de dados também devem conter **índices** que são utilizados principalmente para agilizar operações de pesquisa e classificação. Os bancos de dados que contém índices para acesso aos dados nele armazenados são denominados de **bancos de dados indexados**.

Os **índices** em um banco de dados estabelecem uma ordem pré-definida de registros em determinados campos de uma tabela. No *Paradox* e no *DBase*, os índices são gerados automaticamente, a partir do momento que se define um campo como **chave (key)**.

VANTAGEM DOS BANCOS DE DADOS INDEXADOS:

- os bancos de dados devem ser indexados para agilizar operações de pesquisa e classificação sobre os mesmos, a partir dos campos definidos como chave.

DESVANTAGEM DOS BANCOS DE DADOS INDEXADOS:

- a indexação em um banco de dados torna as operações de atualização e inserção mais lentas.

CHAVES DE ACESSO

As **chaves** são campos do banco de dados para agilizar sua manipulação.

As chaves podem ser classificadas como:

- **chaves primárias**: estabelecem as chaves principais de acesso aos campos de uma tabela;
- **chaves secundárias**: estabelecem chaves auxiliares de acesso aos campos de uma tabela;
- **chaves estrangeiras**: estabelecem chaves de acesso aos campos de outras tabelas.

CHAVES PRIMÁRIAS

As chaves primárias são as principais chaves de acesso a um banco de dados. As principais **características** das chaves primárias são:

- definindo-se uma chave como primária garante-se que ela não estará duplicada dentro de um banco de dados;
- a definição de chaves primárias, em bancos de dados *DBase* e *Paradox*, cria automaticamente um **índice** para acesso aos dados das chaves;
- a definição de chaves primárias faz com que o banco de dados seja automaticamente ordenado pelas chaves;
- para definir uma chave como primária em bancos de dados *DBase* e *Paradox*, basta colocar um * no campo **Key**, utilizando o **Database Desktop**;

É importante ressaltar que um banco de dados pode ter vários campos definidos como chaves primárias.

CHAVES SECUNDÁRIAS

As chaves secundárias estabelecem chaves auxiliares de acesso aos campos de uma tabela. Elas devem ser utilizadas em campos onde se fazem pesquisas ou classificações eventuais sobre uma tabela.

Para se definir uma chave secundária, através do **Database Desktop**, deve-se seguir os passos abaixo:

1. Abra o **Database Desktop** e a tabela onde se quer definir uma chave secundária.
2. Entre na opção **Restructure Table**.
3. Selecione, na caixa **Table properties**, a opção **Secondary Indexes**.
4. Selecione o botão **Define**.
5. Selecione o campo que quer utilizar como chave secundária, de modo que ele apareça na lista **Indexed Fields**.
6. Clique em **OK**, dê um nome a nova chave secundária e clique novamente em **OK**.
7. Salve a tabela utilizando o botão **Save**.

Os campos definidos como chaves secundárias também possuem índices relacionados a eles.

CHAVES ESTRANGEIRAS

As chaves estrangeiras permitem o acesso e validação a outros banco de dados. Suas principais características são:

- as chaves estrangeiras permitem que se faça o relacionamento entre os bancos de dados;
- as chaves estrangeiras, em um banco de dados, armazenam chaves de outros bancos de dados;
- as chaves estrangeiras devem ter o mesmo tipo de sua correspondente chave em um outro banco de dados.

Para se definir uma chave estrangeira, através do Database Desktop, deve-se seguir os passos abaixo:

1. Abra o **Database Desktop** e a tabela onde se quer definir uma chave estrangeira.
2. Entre na opção **Restructure Table**.
3. Selecione, na caixa **Table properties**, a opção **Referential Integrity**.
4. Selecione o botão **Define**.
5. Selecione o campo que quer utilizar como chave estrangeira na coluna **Fields** e clique na seta para a direita para definir o campo **Child Fields**.
6. Selecione a tabela com a qual quer estabelecer o relacionamento e clique na seta para a esquerda para definir o campo **Parent's Key**.
7. O relacionamento está estabelecido entre as chaves definidas em **Child Fields** e **Parent's Key**, que devem ter o mesmo tipo.
8. Clique em **OK**, dê um nome a nova chave secundária e clique novamente em **OK**.
9. Salve a tabela utilizando o botão **Save**.

A restrição desta definição é que o relacionamento pode ser definido apenas com o primeiro campo da outra tabela.

Em tabelas *Paradox* para ser definida como chave estrangeira uma chave tem que ser definida como primária ou secundária.

EXEMPLO DE UM BANCO DE DADOS RELACIONAL

Como exemplo de banco de dados relacional será desenvolvida uma nota fiscal. Para isto devem ser definidas as seguintes tabelas:

- **Nota Fiscal**: Número (chave primária), Data (chave secundária) e Código do Vendedor (chave estrangeira).
- **Itens da Nota Fiscal**: Número do Item (chave primária), Número Nota Fiscal (chave estrangeira), Código do Produto, Quantidade Vendida e Preço de Venda.
- **Vendedor**: Código (chave primária), Nome e Data de Admissão.
- **Produto**: Código (chave primária), Nome, Quantidade em Estoque e Preço.

Depois de definidas as tabelas siga os passos abaixo:

1. Insira dados nas tabelas **Vendedor** e **Produto** no **Database Desktop**, ou defina formulários para realizar este cadastramento.
2. Retorne ao DELPHI e abra um formulário onde possa ser lançada uma nota fiscal.
3. Coloque componentes **Table** e **DataSource** para todas as tabelas.
4. Insira um componente **DBNavigator** ligado a tabela **Nota Fiscal**.
5. Insira dois componentes **DBEdit** ligados a tabela **Nota Fiscal** e os associe aos campos **Número** e **Data**.
6. Insira um componente **DBLookupCombo** ligado a tabela **Nota Fiscal** e o associe ao campo **Código do Vendedor**. Defina que a lista de campos possíveis venha da tabela **Produto**, através das propriedades **ListSource** e **ListField**.
7. Insira um componente **DBGrid** associado a tabela **Itens da Nota Fiscal**.

8. Selecione a tabela de **Itens da Nota Fiscal**. Na propriedade **MasterSource**, defina a fonte de dados da tabela **Nota Fiscal**.
9. Na propriedade **MasterFields** defina o relacionamento entre o campo **Número da Nota Fiscal** da tabela **Itens da Nota Fiscal** e o campo **Número** da tabela **Nota Fiscal**, e pressione em **OK**.
10. No evento **Create** do formulário abra as tabelas e no evento **Destroy** feche-as.

A partir deste momento execute a aplicação. O preenchimento do campo **Número da Nota Fiscal** da tabela **Itens da Nota Fiscal** deverá ser automático, recebendo o valor do campo **Número** da tabela **Nota Fiscal**.

IMPRESSÃO DE RELATÓRIOS

A impressão de relatórios no DELPHI 2.0 pode ser feita de diversas maneiras, entre elas destacam-se:

- uso do programa **Report Smith**;
- uso da classe **TPrinter**, através do objeto **Printer**, utilizando-se os métodos *BeginDoc*, *EndDoc*, *TextOut*, entre outros;
- tratamento da impressora como **arquivo texto** através dos métodos *AssignPrn*, *CloseFile*, *Write*, entre outros;
- uso da paleta de componentes **QReport**.

PALETA QREPORT

A paleta QReport (**Erro! Argumento de opção desconhecido.**) contém 11 componentes que permitem a impressão de relatórios a partir de um banco de dados. A construção de um relatório será feita através de **um exemplo de uso do Quick Report**.

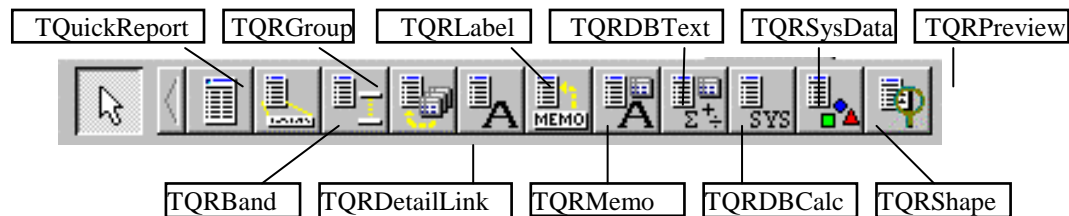


Figura Erro! Argumento de opção desconhecido.: Paleta QReport: componentes de impressão de relatórios

Os componentes desta paleta são:

- **TQuickReport***: estabelece que o formulário corrente é um formulário de impressão.
- **TQRBand#**: permite definir as faixas de impressão.
- **TQRGroup***: permite agrupar dados de mais de um banco de dados.
- **TQRDetailLink***: permite criar relatórios do tipo *Master/Detail*, ou seja, com relacionamentos entre bancos de dados.
- **TQRLabel#**: permite colocar um texto no relatório.
- **TQRMemo#**: permite colocar um Memo no relatório.
- **TQRDBText#**: permite inserir um campo de uma tabela no relatório.

* Componentes não visuais.

Componentes visuais.

- **TQRDBCalc#**: permite fazer operações matemáticas sobre um campo de uma tabela no relatório.
- **TQRSysData#**: permite colocar dados do sistema operacional no relatório.
- **TQRShape#**: permite desenhar figuras geométricas no relatório.
- **TQRPreview#**: permite construir o próprio visualizador de impressão.

EXEMPLO DE USO DO QUICK REPORT

O Quick Report permite que se construa relatórios a partir de um Form. Para construir os relatórios é necessário dividir o Form em faixas, através do componente QRBand. Para ilustrar um exemplo de relatório construído com o Quick Report serão utilizadas as tabelas de Nota Fiscal, apresentadas no item [exemplo de um Banco de Dados Relacional](#).

1. Abra uma nova apresentação.
2. Coloque no formulário três **botões**: um para Imprimir, um para Verificar a Impressão e outro para Sair do aplicativo.
3. Abra um novo formulário.
4. Insira neste novo formulário dois componentes **Table** e dois componentes **DataSource** relacionados aos componentes Table.
5. Relacione os componentes **Table** às tabelas de Nota Fiscal e de Itens da Nota Fiscal, deixando-as com a propriedade **Active** como **TRUE**.
6. Crie um campo calculado na tabela de itens da Nota Fiscal, denominado de Total, e defina no evento **OnCalcFields** da tabela, a multiplicação do campo **Quantidade** pelo campo **Preço**.
7. Faça um relacionamento entre as tabelas definindo os campos **MasterFields** e **MasterSource** da tabela de itens da Nota Fiscal.
8. Insira um componente **QuickReport**. Este componente faz com que o formulário possa ser impresso como um relatório.
9. Insira um componente **QRBand** no **Form**. Deverá aparecer uma faixa branca que pode ser redimensionada de acordo com a necessidade do usuário. Com este componente selecionado defina a propriedade **Name** para "Cabecalho" e a propriedade **BandType** como **rbPageHeader**.
10. Dentro do componente "Cabecalho" insira um componente **QRSysData** e mude a propriedade **Data** para **PageNumber**.
11. Insira um novo componente **QRBand** no **Form**. Defina sua propriedade **Name** para "Titulo" e **BandType** como **rbTitle**.
12. No componente "Titulo" insira um componente **QRLabel** e mude a propriedade **Caption** para "NOTA FISCAL". Mude também a propriedade **Font** para qualquer fonte desejada.
13. Insira um novo componente **QRBand**, definido seu **Name** para "NotaFiscal" e sua propriedade **BandType** como **rbDetail**. Neste componente serão colocados os dados gerais da Nota Fiscal.
14. Neste novo componente **QRBand** insira três componentes **QRLabel**, posicionados um embaixo do outro, e na frente de cada um coloque um componente **QRDBText**.
15. Nomeie os componentes **QRLabel** para "Número", "Data" e "Funcionário" respectivamente.
16. Defina as propriedades **DataSource** e **DataField** dos componentes **QRDBText**. A propriedade **DataSource** deve ser a fonte de dados da Nota Fiscal e a propriedade **DataField** o campo correspondente.
17. Insira um novo componente **QRBand** e defina sua propriedade **Name** para "Cabecalholtens" e **BandType** para **rbGroupHeader** ou **rbColumnHeader**. Neste componente serão colocados os títulos dos campos dos itens da Nota Fiscal.
18. Insira neste novo componente **QRBand** quatro componentes **QRLabel**, posicionados um ao lado do outro. Nomeie estes componentes para "Código", "Quantidade", "Preço" e "Total".

19. Insira um novo componente **QRBand** e defina sua propriedade **Name** para "ItensNotaFiscal" e **BandType** para **rbSubDetail**. Neste componente serão colocados os campos dos itens da Nota Fiscal.
20. Insira neste novo componente **QRBand** quatro componentes **QRDBText**, alinhando-os abaixo de cada item **QRLabel** da faixa anterior.
21. Defina as propriedades **DataSource** e **DataField** dos componentes **QRDBText**. A propriedade **DataSource** deve ser a fonte de dados dos itens da Nota Fiscal e a propriedade **DataField** o campo correspondente.
22. Insira um novo componente **QRBand** e defina sua propriedade **Name** para "Rodapeltens" e **BandType** para **rbGroupFooter**. Neste componente será colocado o total da Nota Fiscal.
23. Neste novo componente **QRBand** coloque um componente **QRDBCalc**, alinhado abaixo da coluna "Total" da faixa anterior.
24. Defina as propriedades **DataSource** para a tabela de itens da Nota Fiscal e a propriedade **DataField** para o campo calculado "Total". Defina também a propriedade **ResetBand** para a faixa "Nota Fiscal".
25. Insira um componente **QRDetailLink**. Defina as seguintes propriedades com os respectivos valores: **DataSource**: DataSource2 - **DetailBand**: ItensNotaFiscal - **FooterBand**: Rodapeltens - **HeaderBand**: Cabecalholtens - **Master**: QuickReport1.

Neste momento está definido o relatório. Para verificar seu resultado associe o comando **Form2.QuickReport1.Preview** ao botão Verificar a Impressão no form principal e **Form2.QuickReport1.Print** ao botão Imprimir. Note que o relatório aparece apenas para o registro corrente. Para que ele apareça para todos os registros da nota fiscal selecione o componente **QuickReport1** e defina sua propriedade **DataSource** como sendo o DataSource1.