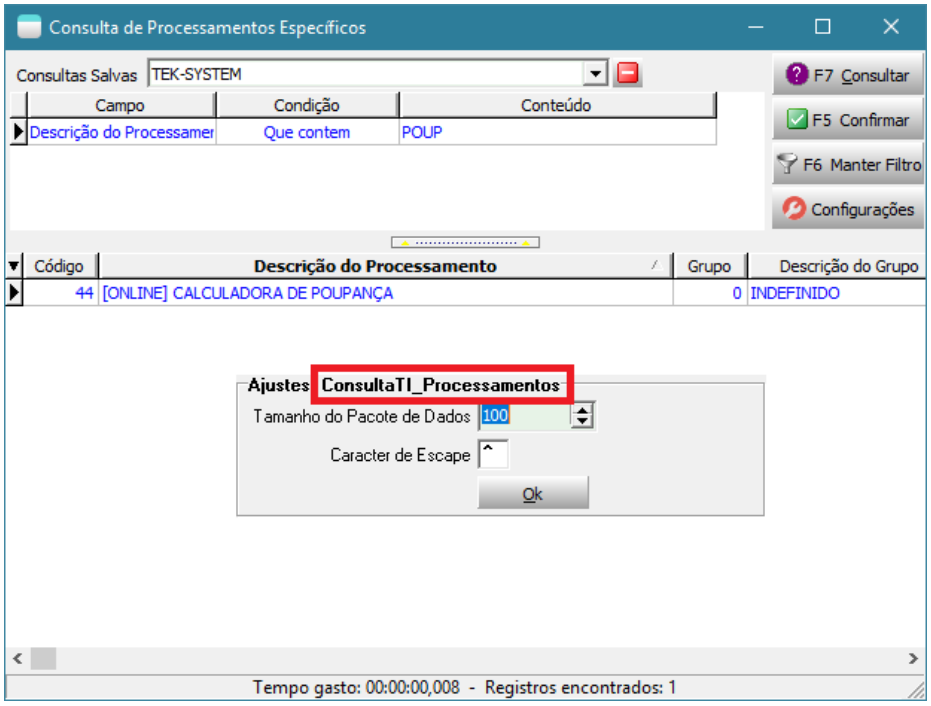
Novidades no módulo BI (Inteligência de Negócios e Gerador de Relatórios) da Tek-System

Inovações

- Criada a possibilidade de acionar, em processamentos específicos, as consultas internas do sistema ou mesmo consultas dinâmicas usando as características das consultas do sistema. Ver exemplos de codificação ao final deste documento. Para saber o nome de uma consulta, pressione F12 com a consulta em execução.



- Criados e disponibilizados dois métodos para ClientDataSet, que permitirão que a exposição da coleção de dados de forma ordenada.

CDS.DadosOrdenados => Devolverá a coleção de dados levando em consideração o índice atribuido no CDS.

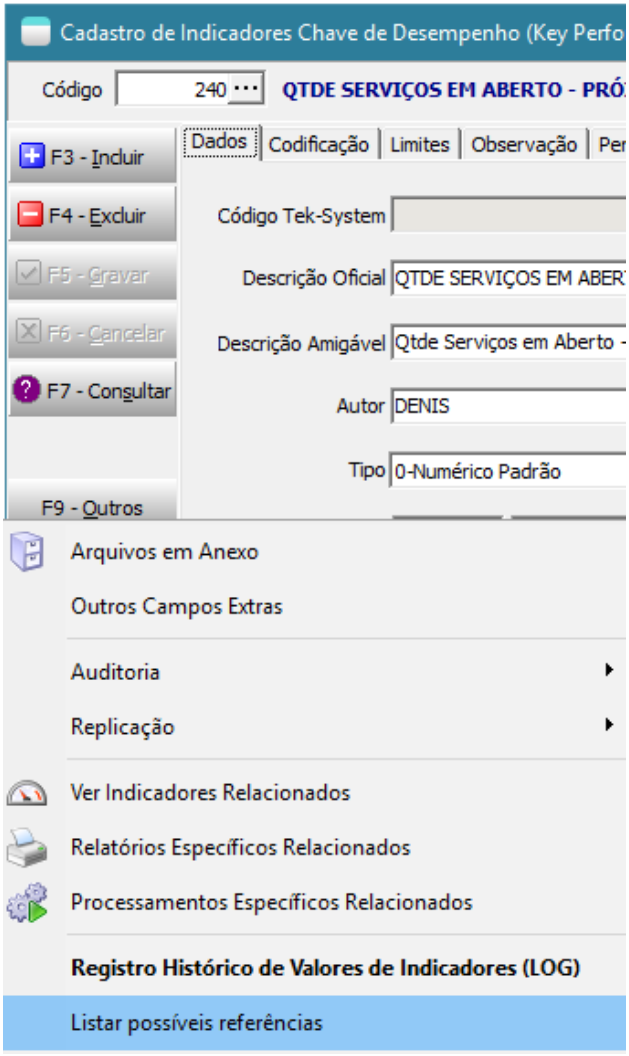
CDS.DadosOrdenados_RemoverCampos([CampoAIgnorar1, CampoAIgnorar2]) => Além de devolver a coleção de dados ordenada ainda permite remover campos que foram usados apenas temporariamente para a montagem dos dados, mas que não são interessantes na visualização final.

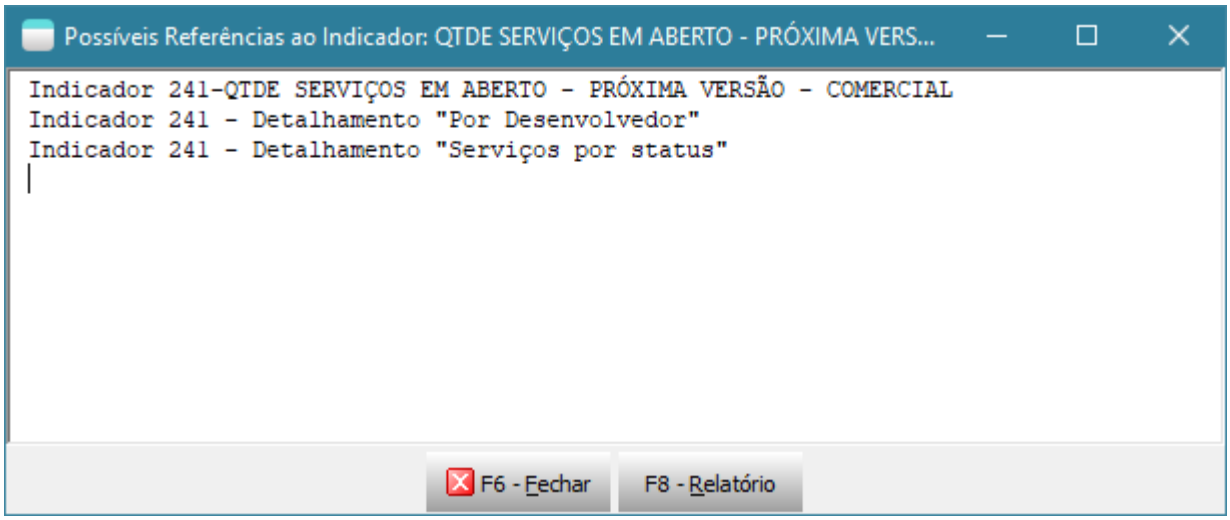
A criação destes métodos facilita a visualização de dados através de tabelas ou gráficos. Não necessitando criar outro artifício externo para ordenação inicial dos indicadores.

Ex:

```
var CDS: TClientDataSet;
begin
  CDS := TClientDataSet.Create;
  try
    CDS.Data := ExecuteReader(
      'select CODIGO_BANCO, DESCRICAO_BANCO, CODCAMARACOMPENSACAO_BANCO from BANCO');
    CDS.IndexFieldNames := 'DESCRICAO_BANCO';
    Result := CDS.DadosOrdenados;
    Result := CDS.DadosOrdenados_RemoverCampos(['CODCAMARACOMPENSACAO_BANCO']);
  finally
    CDS.Free;
  end;
end;
```

- Visto que o cálculo de um indicador pode ser chamado em qualquer lugar do sistema onde se permita interpretação de codificações como regras de processamento, processamentos específicos ou mesmo em outros indicadores, criando um indicador composto ou que utilize outros parâmetros, às vezes era difícil saber se um indicador que estava sendo alterado ou excluído já havia sido utilizado em alguma parte do sistema, então criamos um local específico para realizar esta busca. Disponibilizada a função especial "Listar possíveis referências" no botão outros do cadastro de indicadores.





- Liberada a possibilidade de usar a função MostrarCDS passando um XMLData como parâmetro. As funções Indicador_Valor e IndicadorTek_Valor retornam XMLData, então estava sendo um pouco complicado visualizar os seus dados em meio a processamentos específicos.

Ex:

```
{//Esta era a forma antiga para funcionar:
procedure Main;
var CDS: TClientDataSet;
begin
  CDS := TClientDataSet.Create;
  try
    CDS.XMLData := Indicador_Valor(1084);
    MostrarCDS(CDS);
  finally
    CDS.Free;
end;
end;}

procedure Main;
begin
  MostrarCDS(Indicador_Valor(1084)); // Ajuste para um indicador do tipo tabela de dados
no seu banco de dados.
  MostrarCDS(IndicadorTEK_Valor('StatusPedidosVendaUlt30dias'));
end;
```

- Disponibilizada a possibilidade de chamadas das funções de visualização de títulos em processamentos executados no módulo financeiro.

Ex:

```
  if (StrToInt(GetValueJson(SecaoAtualJson, 'Sistema')) = 3) then
    ExecutarMetodoDeClasse('FuncoesVisualizacaoTitulos', 'TFuncoesVisualizacaoTitulos',
'VerDuplicataReceber', [Application.MainForm, nil, 316])
  else
    ShowMessage('Processamento só pode ser executado no módulo financeiro');
```

- A função SelecionarRegistros quando recebe um ClientDataSet como primeiro parâmetro e o parâmetro ConsiderarReadOnlyDosFields for True, passa a considerar os manipuladores de evento de campos "setText" e "validate" e também o manipulador de evento de registro BeforePost

Ex:

```
procedure Main;
var CDS: TClientDataSet;
begin
  CDS := TClientDataSet.Create;
  try
    CDS.FieldDefs.Add('Marque', ftInteger, 0, False);
    CDS.FieldDefs.Add('CampoS', ftString, 20, False);
    CDS.FieldDefs.Add('CampoD', ftDate, 0, False);
    CDS.CreateDataSet;
    CDS.FieldByName('CampoD').EditMask := '99/99/9999;1; ';
    CDS.AppendRecord([0, 'registro 1', null]);
    CDS.AppendRecord([1, 'registro 2', today]);
    CDS.FieldByName('CampoD').OnSetText := 'SetText_CampoData';
    CDS.BeforePost := 'AntesDeDarPost';
    SelecionarRegistros(CDS, 2, 'Teste de SetText, Validate, BeforePost', True);
  finally
    CDS.Free;
  end;
end;

procedure AntesDeDarPost(DataSet: TDataSet);
begin
  if DataSet.FieldByName('CampoD').IsNull then
    DataSet.FieldByName('Marque').AsInteger := 0
  else
    DataSet.FieldByName('Marque').AsInteger := 1;
end;

procedure SetText_CampoData(Sender: TField; const Text: string);
begin
{  if (Trim(Text) = '/ /') or (Trim(Text) = '') then
    Sender.AsString := ''
  else
    begin
      StrToDate(Text);
      Sender.AsString := Text;
    end;  }

  try
    ExecutarMetodoDeClasse('GetTexts', 'TGetTexts', 'SetText_CampoData', [Sender, Text]);
  except
    on E: Exception do
      begin
        ShowMessage('Data Inválida');
        Abort;
      end;
  end;
end;
```

- Criada e registrada classe que permite realizar cópia de itens de PopupMenus.
- Disponibilizada a possibilidade de preencher um PopupMenu de grade com opções de marcação, em processamentos interpretados.
- Disponibilizadas novas opções de manipulação de grades em processamentos específicos. CentralizarTitulos, Configurar, ConfigurarColunaMarcacao, OcultarColuna,ReexibirColunas, GridParaClipBoard

Ex:

```
procedure Main;
var
  F: TForm;
  G: TJvDBGrid;
  DS: TDataSource;
  CDS: TClientDataSet;
  PM: TPopupMenu;
begin
  F   := TForm.Create(nil);
  G   := TJvDBGrid.Create(F);
  PM  := TPopupMenu.Create(F);
  DS  := TDataSource.Create(F);
  CDS := TClientDataSet.Create;
  try
    F.Caption     := 'Selecione os títulos que estão sendo cobrados';
    F.Width       := 800;
    F.Height      := 600;
    F.Position    := poScreenCenter;

    G.Parent      := F;
    G.Align       := alClient;
    G.DataSource  := DS;
    G.PopupMenu   := PM;
//    G.PopupMenu   := DMCriadoPeloNome('DMConexao').FindComponent('PopupMenuGrid');
    ExecutarMetodoDeClasse('ClassFuncoesMenus', 'TClassFuncoesMenus', 'CopiarMenuPopup',
[
      Application,
      DMCriadoPeloNome('DMConexao').FindComponent('PopupMenuGrid'),
      PM,
      False]);
    ExecutarMetodoDeClasse('ClassFuncoesMenus', 'TClassFuncoesMenus',
'CriarOpcoesMarcacaoEmMenuPopup', [PM, True]);

    DS.DataSet    := CDS;

    CDS.Data := ExecuteReader('select 0 MARQUE, CODIGO_STATUS, DESCRICAO_STATUS from
status');

    G.CentralizarTitulos;
    G.Configurar;
    G.ConfigurarColunaMarcacao(nil, True);
    G.OcultarColuna('CODIGO_STATUS');
    //G.ReexibirColunas;

    G.GridParaClipBoard(
      {ComCabecalho}True,
      {ApenasLinhaAtual}False,
      {ApenasColuna}'');

    F.ShowModal;
  finally
    F.Free;
  end;
end;
```

- Ajustes no sistema para permitir maiores possibilidades de interação com a tela de seleção de registros, em processamentos interpretados.

```
unit ProcessamentoEspecifico;

var CDS: TClientDataSet;

procedure main;
var
  F: TFSelecionaRegistros;
  MI: TMenuItem;
  PM: TPopupMenu;
begin
  F := CriarFormPeloNome('FSelecionaRegistros');
  MI := TMenuItem.Create(F);
  CDS := TClientDataSet.Create;
  try
    F.Width := 800;
    F.Height := 600;
    CDS.Data := ExecuteReader('select 0 marque, duplicata.* from duplicata where
duplicata.tipo_dup = 1 and duplicata.valoraberto_dup > 0');
    if (StrToInt(GetValueJson(SecaoAtualJson, 'Sistema')) = 3) then
      begin
        MI.Caption := 'Ver Documento';
        MI.OnClick := 'VerDocumento';
        MI.Default := True;
        MI.Visible := True;
        PM := TPopupMenu(F.FindComponent('PMSelecao'));
        PM.Items.Insert(0, MI);
      end;
    ExecutarMetodoDeClasse('USelecionaRegistros', 'TFSelecionaRegistros',
'SelecionarRegistrosFC', [F, CDS, 2, 'Selecione os títulos que estão sendo cobrados',
True]);
  finally
    CDS.Free;
    F.Free;
  end;
end;

procedure VerDocumento;
begin
  ExecutarMetodoDeClasse(
       'FuncoesVisualizacaoTitulos',
       'TFuncoesVisualizacaoTitulos',
       'VerDuplicataReceber',
       [Application.MainForm, nil, CDS.FieldByName('AUTOINC_DUP').AsInteger]);
end;

end.
```

Novas funções disponibilizadas

- AplicarMacroSubstituicao: Aplica as mesmas macrosubstituições disponíveis em filtros dinâmicos de relatórios.
  Ex: `DateToStr(AplicarMacroSubstituicao(GetValueJson(RequestBodyJSON, 'Data')));`

- ExecutarMetodoDeClasse: permitirá executar métodos de classes do sistema que estejam registradas e que não dependam de uma instância criada.
  Ex: `ShowMessage(ExecutarMetodoDeClasse('ClassPessoa', 'TClassPessoa', 'DescricaoTipoConta', [1, True]));`

Codificações para testes das classes/funções

- **Exemplo de chamadas de consultas em meio a processamentos específicos:**

```
unit ProcessamentoEspecifico;

procedure Main;
var Retorno: Variant;
begin
//  RetornarConsulta(); // Deve dar erro exibindo quais são os parâmetros necessários

//  RetornarConsulta('ConsultaBorderoDescontoX'); // Deve dar erro acusando que a
TClassBorderoDescontoX não está disponível neste módulo

{
  // Deve consultar clientes e retornar o código selecionado
  Retorno := RetornarConsulta('ConsultaCliente');
  if Assigned(Retorno) then
    ShowMessage('O código do cliente selecionado é ' + VarToStr(Retorno));
}
{
  // Deve consultar contas a receber e retornar o vencimento do título selecionado
  Retorno := RetornarConsulta('ConsultaReceber', nil, 3);
  if Assigned(Retorno) then
    ShowMessage('O vencimento do título selecionado é ' + VarToStr(Retorno));
}

  // Deve executar a função configurada para executar consultas dinâmicas
  ConsultarPessoa;
end;

procedure ConsultarPessoa;
var TelaConsulta: TFPaiConsulta;
begin
  TelaConsulta := CriarFormPeloNome('FPaiConsulta');
  TelaConsulta.ConsultaAtual.Titulo := 'Consulta Dinâmica de Pessoas';

  ExecutarMetodoDeObjeto(TelaConsulta, 'ConfigureConsulta',
    ['select' +
     '  PESSOA.CODIGO_PESSOA      "Código",'      + #13 +
     '  PESSOA.RAZAOSOCIAL_PESSOA "Razão Social"' + #13 +
     ' from PESSOA']);

  TelaConsulta.ConsultaAtual.CampoEmpresa := '';
  TelaConsulta.ConsultaAtual.FiltroFixo   := 'PESSOA.CLIENTE_PESSOA = ''S''';
  TelaConsulta.ConsultaAtual.GroupBy      := '';
  TelaConsulta.ConsultaAtual.MapeamentoDeCampos.Values['Código']       :=
'PESSOA.CODIGO_PESSOA';
  TelaConsulta.ConsultaAtual.MapeamentoDeCampos.Values['Razão Social'] :=
'PESSOA.RAZAOSOCIAL_PESSOA';
  //ExecutarMetodoDeObjeto(TelaConsulta.ConsultaAtual, 'MapearCampo', ['Código',
'PESSOA.CODIGO_PESSOA']);
  //ExecutarMetodoDeObjeto(TelaConsulta.ConsultaAtual, 'MapearCampo', ['Razão Social',
'PESSOA.RAZAOSOCIAL_PESSOA']);
  ExecutarMetodoDeObjeto(TelaConsulta.ConsultaAtual, 'ConfigureRetorno', [1, 'Código']);

  if (TelaConsulta.ShowModal = mrOk) then
    begin
      ShowMessage('Código retornado pela consulta foi ' + VarToStr(RetornoDeConsulta(1))
+ '. A seguir visualize todos os registros que estavam sendo exibidos pelo filtro no ato
da confirmação da consulta.');
      MostrarCDS(RetornoDeConsulta);
    end;
end;

end.
```

- **Teste de uso de consultas e validações de campos**

```
unit ProcessamentoEspecifico;

const PosY = 50;

var E1, E2: TEdit;

procedure Main;
var
  F: TForm;
  L: TLabel;
  B: TButton;
  SB: TSpeedButton;
begin
  F  := TForm.Create(nil);
  L  := TLabel.Create(F);
  E1 := TEdit.Create(F);
  E2 := TEdit.Create(F);
  B  := TButton.Create(F);
  SB := TSpeedButton.Create(F);
  try
    F.Caption      := 'Teste de Consulta e Validação';
    F.Height       := 200;
    F.Width        := 430;
    F.Position     := poScreenCenter;
    F.KeyPreview   := True;
    F.OnKeyDown    := 'TratarPressTeclaForm';

    L.Parent       := F;
    L.Top          := PosY;
    L.Left         := 20;
    L.Caption      := 'Pessoa';
    L.Height       := 21;
    L.LayOut       := tlCenter;

    E1.Parent      := F;
    E1.Top         := PosY;
    E1.Left        := 60;
    E1.Width       := 50;
    E1.NumbersOnly := True;
    E1.Text        := '0';
    E1.OnKeyDown   := 'TratarPressTeclaEdit';
    E1.OnExit      := 'TratarSaidaEdit';

    SB.Parent      := F;
    SB.Top         := PosY;
    SB.Left        := 111;
    SB.Width       := 20;
    SB.Height      := 21;
    SB.Caption     := '...';
    SB.OnClick     := 'TrataClickBotao';

    E2.Parent      := F;
    E2.Top         := PosY;
    E2.Left        := 132;
    E2.Width       := 250;
    E2.Text        := 'INDEFINIDA';
    E2.ReadOnly    := True;
    E2.TabStop     := False;
    E2.Color       := clBtnFace;

    B.Parent       := F;
    B.Text         := 'Confirmar';
    B.Top          := 150;
    B.Anchors      := [];
```

```
    B.Width        := 100;
    B.Left         := ((F.Width - B.Width) div 2);
    B.ModalResult  := mrOk;

    if (F.ShowModal = mrOk) then
      ShowMessage('Confirmado');
  finally
    F.Free;
  end;
end;

procedure TratarPressTeclaForm(Sender: TObject; var Key: Word; Shift: TShiftState);
begin
  if (Key = vk_Escape) then Screen.ActiveForm.Close;
end;

procedure TratarPressTeclaEdit(Sender: TObject; var Key: Word; Shift: TShiftState);
begin
  if (Key = vk_F1) then
    RetornarConsulta('ConsultaPessoa', Sender);
end;

procedure TrataClickBotao(Sender: TObject);
begin
  RetornarConsulta('ConsultaPessoa', E1);
end;

procedure TratarSaidaEdit(Sender: TObject);
var
  CodPessoa: Integer;
  NomePessoa: String;
begin
  CodPessoa  := StrToIntDef(TEdit(Sender).Text, 0);
  NomePessoa := LocalizarPessoa(CodPessoa);
  if (Trim(NomePessoa) = '') then
    begin
      ShowMessage('Pessoa não localizada');
      Abort;
    end;

  E2.Text := NomePessoa;
end;

function LocalizarPessoa(CodPessoa: Integer): string;
begin
  Result := ExecuteScalar('select PESSOA.RAZAOSOCIAL_PESSOA from PESSOA where
PESSOA.CODIGO_PESSOA = ' + IntToStr(CodPessoa));
end;

end.
```

Novos modelos de indicadores padrões disponibilizados

MTCONTROL: JSON DE PEÇAS DO LOTE POR MÁQUINA

Unidades de Codificação Novas ou Ajustadas

TEK_DISTRIBUICAO_DFE -> Permitirá agendar a busca de NFe e CTe destinados à empresa.

Abraço a todos!

Denis Pereira Raymundo
Certified Delphi Developer
Professional Coach of Life Coaching
Especialista em Gestão e Manutenção de Tecnologia da Informação
Bacharel em Ciência da Computação
Licenciado em Matemática
Técnico em Processamento de Dados

Gerente de Sistemas
www.teksystem.com.br
Prêmios: Top Móbile - Segmento: Fornecedores de Softwares p/Setor Moveleiro
- 1ª lugar (2013)
- 2ª lugar (2012, 2014, 2015, 2016 e 2018)
- 3ª lugar (2009)

"Em tudo dai graças, porque esta é a vontade de Deus em Cristo Jesus para convosco." 1 Ts 5.18