LISTING PROGRAM

```
unit AllUnits;

interface
Uses Windows, SysUtils, Controls, Messages, Consts, DB,
     DBGrids, DBTables, classes, forms, Menus, Registry,
     DBCtrls, Dialogs, FileUtil, ShlObj;

Const PublicKey = 'THESIS';
      PublicNum = 2;

procedure DispErr(str : string; oCtrl : TWinControl);
procedure EnterKey(Key : char; Sender : Tobject);

function EnCrypt(Value:String):String;
function DeCrypt(Value:String):String;

function StrRight(VStr : String; VDigit : Integer): String;
function StrMiddle(VStr: String; VdigitAwal, VDigit : Integer): String;
function StrLeft(VStr : String; VDigit : Integer): String;

function Inc1StrNum(VStrNum : String; VDigit : Integer): String;

function  OpenTable(tablename, cIdx : string; parent : TComponent) : TTable;
procedure DBSetPath(parent : TComponent);
procedure DBinitvalue(dataset : TDataset);
procedure DBinitvalue2(dataset : TDataset);

procedure AddColumn(Grid : TDBGrid; FieldName : string);
procedure DBLoadLabel(Dataset : TDataset);

function AppendSlash(Path: string): string;

function  DBGetTotal(field : Tfield) : currency;
function  DBGetCounter(prefix : string; parent : TComponent) : string;

var PathExe, PathData     : string;
    aLabel                : TStringList;
    PathReport, UserLogin : string;

implementation

procedure DispErr;
begin
  try
    if oCtrl <> nil then
```

```
                if oCtrl.Enabled and oCtrl.Visible and oCtrl.Parent.Enabled and
oCtrl.Parent.Visible then oCtrl.SetFocus;
  except
  end;
  raise ERangeError.Create(str);
end;

procedure EnterKey(Key : char; Sender : Tobject);
begin
  if Key = #13 then
    if TForm(Sender).ActiveControl.ClassType <> TDBMemo then
      if TForm(Sender).ActiveControl.ClassType <> TCustomDBGrid then
        TControl(Sender).Perform(WM_NEXTDLGCTL, 0, 0);
end;

function StrLeft(VStr : String; VDigit : Integer): String;
begin
  Result:=Copy(VStr,1,Vdigit);
end;

function StrMiddle(VStr: String; VdigitAwal, VDigit : Integer): String;
begin
  Result:=Copy(VStr,VdigitAwal,Vdigit);
end;

function StrRight(VStr : String; VDigit : Integer): String;
var VLength : Integer;
begin
  VLength:=Length(VStr);
  try
    Result:=Copy(VStr,VLength-VDigit+1,Vdigit);
  except
    Result:='Salah';
    exit;
  end;
end;

function EnCrypt(Value:String):String;
var tmpHasil, Hasil, strPublicKey : String;
    i, panjang : integer;
    Asc : Byte;
begin
  Hasil:='';
  {1. Using Public Keys Methoda, Hasil = Value + PublicKey }
  strPublicKey:=PublicKey;
  While length(Value)>Length(strPublicKey) Do
    strPublicKey:=strPublicKey+PublicKey;
  for i:=1 to length(value) do
    Hasil:=Hasil + Chr(Ord(Value[i]) + Ord(StrPublicKey[i]));
```

```
  {2. Using Transposition Location, Hasil = Hasil[1] <-> Hasil[2], Hasil[3] <->
Hasil[4], etc... if ganjil don't do anything just take it}
 tmpHasil:=Hasil;
 panjang:=length(tmpHasil);
 if (panjang mod 2) <> 0 Then
   panjang:=panjang - (panjang mod 2);
 Hasil:='';
 i:=1;
 While panjang>i do
   begin
     Hasil:=Hasil + tmpHasil[i+1] + tmpHasil[i];
     inc(i,2);
   end;
 if (length(tmpHasil) mod 2) <> 0 Then
   Hasil:=Hasil + tmpHasil[length(tmpHasil)];

 {3. Using Algorithm Change like 1..n/2, hasil = left from 1/2n + right 1/2nNext }
 tmpHasil:=Hasil;
 Hasil:='';
 panjang:=length(tmpHasil);
 for i:=(panjang div 2)  downto 1 do
   Hasil:=Hasil + tmpHasil[i];
 for i:=panjang downto ((panjang div 2)+1) do
   Hasil:=Hasil + tmpHasil[i];

 {4. Using Adding PublicNum to every char, hasil = value + PublicNum }
 tmpHasil:=Hasil;
 Hasil:='';
 panjang:=length(tmpHasil);
 for i:=1 to panjang do
    Hasil:=Hasil + chr(ord(tmpHasil[i]) + PublicNum);

 {5. Using Sorted Algorithm location, hasil = hasil[n]...hasil[1]}
 tmpHasil:=Hasil;
 Hasil:='';
 panjang:=length(tmpHasil);
 for i:=panjang downto 1 do
   Hasil:=Hasil + tmpHasil[i];

 {6. Using Tranposition 1 to n and n to 1 }
 tmpHasil:=Hasil;
 if length(tmpHasil) > 1 Then
        Hasil:=StrRight(tmpHasil,1)  +  StrMiddle(tmpHasil,2,length(tmpHasil)-2)    +
StrLeft(tmphasil,1)
 else
    Hasil:=tmpHasil;

 {7. Using Xor Style }
 tmpHasil:=Hasil;
 Hasil:='';
```

```
  for i:=1 to length(tmpHasil) do
   begin
     Asc:=Ord(tmpHasil[i]);
     Asc:=Asc Xor 8;
     Hasil:=Hasil + Chr(Asc);
   end;
  if length(hasil) <> length(Value) then Hasil:=Copy(hasil,1,length(value));
  Result:=Hasil;
end;

function DeCrypt(Value:String):String;
var tmpHasil, Hasil, strPublicKey : String;
    i, panjang : integer;
    Asc : Byte;
begin
  Hasil:=Value;

  {1. Using Xor Style }
  tmpHasil:=Hasil;
  Hasil:='';
  for i:=1 to length(tmpHasil) do
   begin
     Asc:=Ord(tmpHasil[i]);
     Asc:=Asc Xor 8;
     Hasil:=Hasil + Chr(Asc);
   end;

  {2. Using Tranposition 1 to n and n to 1 }
  tmpHasil:=Hasil;
  if length(tmpHasil) > 1 Then
         Hasil:=StrRight(tmpHasil,1)  +  StrMiddle(tmpHasil,2,length(tmpHasil)-2)     +
StrLeft(tmphasil,1)
  Else
    Hasil:=tmpHasil;

  {3. Using Sorted Algorithm location, hasil = hasil[n]...hasil[1]}
  tmpHasil:=Hasil;
  Hasil:='';
  panjang:=length(tmpHasil);
  for i:=1 to panjang do
   Hasil:=tmpHasil[i]+ Hasil;

  {4. Using Adding PublicNum to every char, hasil = value + PublicNum }
  tmpHasil:=Hasil;
  Hasil:='';
  panjang:=length(tmpHasil);
  for i:=1 to panjang do
   Hasil:=Hasil + chr(ord(tmpHasil[i]) - PublicNum);

  {5. Using Algorithm Change like 1..n/2, hasil = right from 1/2n + left 1/2nNext }
```

```
  tmpHasil:=Hasil;
  Hasil:='';
  panjang:=length(tmpHasil);
  for i:=(panjang div 2)+1 to panjang do
    Hasil:=tmpHasil[i] + Hasil;
  for i:=1 to (panjang div 2) do
    Hasil:=tmpHasil[i] + Hasil;

    {6. Using Transposition Location, Hasil = Hasil[2] <-> Hasil[1], Hasil[4] <->
Hasil[3], etc... if ganjil don't do anything just take it}
  tmpHasil:=Hasil;
  Hasil:='';
  panjang:=length(tmpHasil);
  if (panjang mod 2) <> 0 Then
    panjang:=panjang - (panjang mod 2);
  i:=1;
  While panjang>i do
    begin
      Hasil:=Hasil + tmpHasil[i+1] + tmpHasil[i];
      inc(i,2);
    end;
  if (length(tmpHasil) mod 2) <> 0 Then
    Hasil:=Hasil + tmpHasil[length(tmpHasil)];

  {7. Using Public Keys Methoda, Hasil = Value - PublicKey }
  tmpHasil:=Hasil;
  Hasil:='';
  strPublicKey:=PublicKey;
  While length(TmpHasil)>Length(strPublicKey) Do
    strPublicKey:=strPublicKey+PublicKey;
  for i:=1 to length(tmpHasil) do
    Hasil:=Hasil + Chr(Ord(tmphasil[i]) - Ord(StrPublicKey[i]));

  if length(hasil) <> length(Value) then Hasil:=Copy(hasil,1,length(value));
  Result:=Hasil;
end;

function OpenTable;
var o : TTable;
begin
  o := nil;
  if parent <> nil then o := TTable(parent.FindComponent('tmp' + tablename));
  if o = nil then begin
    o := TTable.Create(parent);
    o.DatabaseName    := pathdata;
    o.TableName       := tablename;
    o.Name            := 'tmp' + tablename;
    o.IndexFieldNames := cIdx;
    o.Open;
    o.First;
```

```
  end else begin
    if not o.Active then o.Open;
    o.Tag := 1;
  end;
  result := o
end;

procedure DBSetPath;
var n : word;
begin
  for n := 0 to parent.ComponentCount - 1 do begin
    if parent.Components[n] is TTable then
    begin
      try
        TTable(parent.Components[n]).Close;
        TTable(parent.Components[n]).DatabaseName := pathdata;
      except
        on E: Exception do
        begin
          raise Exception.Create(TTable(parent.Components[n]).Name+': '+E.Message);
        end;
      end;
    end;
  end;
end;

procedure DBinitvalue(dataset : TDataset);
var n : word;
begin
  for n := 0 to dataset.FieldCount - 1 do
    if dataset.Fields[n].IsNull then
      case dataset.Fields[n].DataType of
        ftSmallint,
        ftInteger,
        ftWord,
        ftFloat,
        ftCurrency  : dataset.Fields[n].AsVariant  := 0;
        ftDate      : dataset.Fields[n].AsDateTime := date;
        ftBoolean   : dataset.Fields[n].AsBoolean  := false;
      end;
end;

procedure DBinitvalue2(dataset : TDataset);
var n : word;
begin
  for n := 0 to dataset.FieldCount - 1 do
    if dataset.Fields[n].IsNull then
      case dataset.Fields[n].DataType of
        ftSmallint,
        ftInteger,
```

```
        ftWord,
        ftFloat,
        ftCurrency  : dataset.Fields[n].AsVariant  := 0;
        ftBoolean   : dataset.Fields[n].AsBoolean  := false;
      end;
  end;

  procedure AddColumn;
  var o : TColumn;
  begin
    o := TDBGrid(Grid).Columns.Add;
    o.FieldName     := FieldName;
  end;

  procedure DBLoadLabel;
  var n, m : integer;
  begin
    if aLabel = nil then begin
      aLabel := TStringlist.Create;
      aLabel.LoadFromFile(pathexe + 'LABEL.DAT');
    end;
    for n := 0 to dataset.FieldCount - 1 do begin
      m := aLabel.IndexOfName(dataset.Fields[n].FieldName);
if m                                  dataset.Fields[n].DisplayLabel
  aLabel.Values[dataset.Fields[n].FieldName];
    end;
  end;

  function AppendSlash(Path: string): string;
  begin
    Result:=Path;
    if (Result<>'') and (AnsiLastChar(Result)<>'\') then Result:=Result+'\';
  end;

  function DBGetTotal;
  var bookmark : TBookmark;
      oldstate : TDatasetState;
  begin
    result := 0;
    if (field.dataset.state <> dsInsert) and not field.dataset.EOF then
    begin
      field.DataSet.DisableControls;
      oldstate := field.DataSet.State;
      bookmark := field.DataSet.GetBookmark;
      field.DataSet.First;
      while not field.DataSet.EOF do begin
        result := result + field.asCurrency;
        field.DataSet.Next
      end;
```

```
      field.DataSet.GotoBookmark(bookmark);
      field.DataSet.FreeBookmark(bookmark);
      field.DataSet.EnableControls;
      if oldstate = dsEdit then field.DataSet.Edit;
    end;
  end;

  function DBGetCounter;
  var tsetup,o : TTable;
  begin
    o := TTable(parent.FindComponent('tmpTCOUNT'));
    if o = nil then
      o := OpenTable('TCOUNT', '', parent)
    else
      o.Open;

    tsetup:= OpenTable('TSETUP', '', nil);

    if not o.FindKey([prefix]) then begin
      o.Append;
      o.fieldbyname('NAMA').asString     := prefix;
      o.fieldbyname('NOMER').asString    := '0000001';
      o.fieldbyname('PANJANG').asInteger := 7;
    end else begin
      o.Edit;
   o.fieldbyname('NOMER').asString:=
    Inc1StrNum(o.fieldbyname('NOMER').asString, 7);
    end;

Result:=prefix+
  tsetup.fieldbyname('KODECAB').asString+strright(o.fieldbyname('NOMER').asStrin
  g, 7);
    o.Post;
    o.Close;
    tsetup.close;
  end;

  function Inc1StrNum(VStrNum : String; VDigit : Integer): String;
  var Vnum : LongInt;
      VTempHasil : String;
      Count : Integer;
  begin
    try
      Vnum:=StrToInt(VStrnum);
      Vnum:=Vnum+1;
    except
      Result:='Salah';
      exit;
    end;
    If Length(IntToStr(Vnum)) > VDigit then
```

```
    begin
      VNum:=0;
    end;
  VTempHasil:=IntToStr(VNum);
  Result:='';
  For Count:=(VDigit - Length(VTempHasil) - 1) downto 0 do
    Result:='0'+Result;
  Result:=Result+VTempHasil;
end;

initialization
  pathexe := extractfilepath(paramstr(0));
end.
```

program Thesis;

```
uses Sysutils,
  Forms,
  Dialogs,
  AllUnits in 'AllUnits.pas',
  uMenuUtama in 'uMenuUtama.pas' {fmMenuUtama},
  UDmInventory in 'UDmInventory.pas' {dmInventory: TDataModule},
  USearch in 'USearch.pas' {fmSearch},
  uDMEntry in 'Udmentry.pas' {dmEntry: TDataModule},
  uBrowse in 'Ubrowse.pas' {fmBrowse},
  uAbout in 'uAbout.pas' {fmAbout},
  UDmPembelian in 'UDMPembelian.pas' {dmPembelian: TDataModule},
  UBeli in 'UBeli.pas' {fmBeli},
  uRtBl in 'uRtBl.pas' {fmReturBeli},
  UDmPenjualan in 'Udmpenjualan.pas' {dmPenjualan: TDataModule},
  UJual in 'UJual.pas' {fmJual},
  uRtJl in 'uRtJl.pas' {fmReturJual},
  UBayarHutang in 'UBayarHutang.pas' {fmBayarHutang},
  uDMHutang in 'Udmhutang.pas' {dmHutang: TDataModule},
  UBayarPiutang in 'uBayarPiutang.pas' {fmBayarPiutang},
  UAdin in 'UAdin.pas' {fmAdin},
  UAdou in 'UAdou.pas' {fmAdou},
  uSrv in 'uSrv.pas' {fmSrv},
  uRepBarang in 'uRepBarang.pas' {fmRepBarang},
  uLap in 'uLap.pas' {dmLap: TDataModule};

{$R *.RES}
begin
  ShortDateFormat := 'dd/mm/yyyy';
  LongDateFormat  := 'dd/mm/yyyy';

  CurrencyString  := 'Rp ';

  Application.CreateForm(TfmMenuUtama, fmMenuUtama);
```

```
  Application.CreateForm(TdmInventory, dmInventory);
  Application.CreateForm(TfmSearch, fmSearch);
  Application.CreateForm(TdmEntry, dmEntry);
  Application.CreateForm(TfmBrowse, fmBrowse);
  Application.CreateForm(TfmAbout, fmAbout);
  Application.CreateForm(TdmPembelian, dmPembelian);
  Application.CreateForm(TfmBeli, fmBeli);
  Application.CreateForm(TfmReturBeli, fmReturBeli);
  Application.CreateForm(TdmPenjualan, dmPenjualan);
  Application.CreateForm(TfmJual, fmJual);
  Application.CreateForm(TfmReturJual, fmReturJual);
  Application.CreateForm(TfmBayarHutang, fmBayarHutang);
  Application.CreateForm(TdmHutang, dmHutang);
  Application.CreateForm(TfmBayarPiutang, fmBayarPiutang);
  Application.CreateForm(TfmAdin, fmAdin);
  Application.CreateForm(TfmAdou, fmAdou);
  Application.CreateForm(TfmSrv, fmSrv);
  Application.CreateForm(TfmRepBarang, fmRepBarang);
  Application.CreateForm(TdmLap, dmLap);
  Application.Run;
end.
```

unit uAbout;

```
interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  ExtCtrls, RXCtrls, StdCtrls;

type
  TfmAbout = class(TForm)
    Image1: TImage;
    SecretPanel1: TSecretPanel;
    CheckBox1: TCheckBox;
    procedure FormShow(Sender: TObject);
    procedure FormClose(Sender: TObject; var Action: TCloseAction);
    procedure CheckBox1Click(Sender: TObject);
  private
  public
    SystemFile : TStringList;
  end;

var
  fmAbout: TfmAbout;

implementation

{$R *.DFM}
```

```
procedure TfmAbout.FormShow(Sender: TObject);
begin
  SecretPanel1.Active := True;

  SystemFile := TStringList.Create;
  SystemFile.LoadFromFile(ExtractFilePath(ParamStr(0)) + 'System.dat');

  if SystemFile.Values['ShowAbout'] = 'Yes' then
    CheckBox1.Checked := False
  else CheckBox1.Checked := True;
end;

procedure TfmAbout.FormClose(Sender: TObject; var Action: TCloseAction);
begin
  SecretPanel1.Active := False;
end;

procedure TfmAbout.CheckBox1Click(Sender: TObject);
begin
  if CheckBox1.Checked then
    SystemFile.Values['ShowAbout'] := 'No'
  else SystemFile.Values['ShowAbout'] := 'Yes';

  SystemFile.SaveToFile(ExtractFilePath(ParamStr(0)) + 'System.dat');
end;

end.
```

unit UAdin;

```
interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  Menus, Grids, DBGrids, ComCtrls, ZKNavigator, ExtCtrls, StdCtrls,
  DBCtrls, Mask, Db, DBTables, ToolEdit, RXDBCtrl;

type
  TfmAdin = class(TForm)
    DBGrid1: TDBGrid;
    Bevel1: TBevel;
    dsWatch: TDataSource;
    Panel3: TPanel;
    Label1: TLabel;
    Label2: TLabel;
    Panel2: TPanel;
    DBText1: TDBText;
    Label3: TLabel;
```

```
    Panel4: TPanel;
    Label4: TLabel;
    KNav: TK2Navigator;
    Panel1: TPanel;
    dsDetail: TDataSource;
    deTanggal: TDBDateEdit;
    PopupMenu1: TPopupMenu;
    Add1: TMenuItem;
    Edit1: TMenuItem;
    Delete1: TMenuItem;
    Simpan1: TMenuItem;
    Batal1: TMenuItem;
    deketerangan: TDBMemo;
    procedure KNavAddClick(Sender: TObject);
    procedure KNavCancelClick(Sender: TObject);
    procedure FormShow(Sender: TObject);
    procedure FormClose(Sender: TObject; var Action: TCloseAction);
    procedure DBGrid1EditButtonClick(Sender: TObject);
    procedure KNavSaveClick(Sender: TObject);
    procedure dsWatchStateChange(Sender: TObject);
    procedure dsDetailDataChange(Sender: TObject; Field: TField);
    procedure Add1Click(Sender: TObject);
    procedure Edit1Click(Sender: TObject);
    procedure Delete1Click(Sender: TObject);
    procedure Simpan1Click(Sender: TObject);
    procedure Batal1Click(Sender: TObject);
    procedure PopupMenu1Popup(Sender: TObject);
    procedure KNavPreviewClick(Sender: TObject);
    procedure KNavSearchClick(Sender: TObject);
    procedure DBGrid1KeyPress(Sender: TObject; var Key: Char);
  private
    taBrg    : TTable;
    subtotal : currency;

    procedure taAdinDtlKODEBRGChange(Sender: TField);
    procedure TaAdinDtlNewRecord(DataSet: TDataSet);
    procedure taAdinDtlBeforeDelete(DataSet: TDataSet);
    procedure taAdinDtlCalcFields(DataSet: TDataSet);
    procedure UpdateStok;
  public
    { Public declarations }
  end;

var
  fmAdin: TfmAdin;

implementation

uses UdmInventory, AllUnits, UBrowse, USearch;
```

```
{$R *.DFM}

procedure TfmAdin.taAdinDtlKODEBRGChange(Sender: TField);
begin
  with dmInventory do
  begin
    taAdinDtlKODEBRG.OnChange := nil;

    if not taBrg.FindKey([Sender.asString]) then DBGrid1EditButtonClick(nil);
    taAdinDtl.Edit;
    taAdinDtlKODEBRG.Value := taBrg.FieldByName('KODE').asString;
    taAdinDtlNAMABRG.Value := taBrg.FieldByName('NAMA').asString;
    taAdinDtlSATUAN.Value  := taBrg.FieldByName('SATUAN').asString;
    taAdinDtlJUMLAH.Value  := 1;
    taAdinDtlHARGA.Value   := 0;
    taAdinDtl.Post;

    taAdinDtlKODEBRG.OnChange := taAdinDtlKODEBRGChange;
  end;
end;

procedure TfmAdin.taAdinDtlCalcFields;
begin
  with dmInventory do
    If taAdinDtl.Active Then
taAdinDtlTOTAL.Value:=taAdinDtlJUMLAH.Value*
taAdinDtlHARGA.Value;
end;

procedure TfmAdin.TaAdinDtlNewRecord;
begin
  DBInitValue(dmInventory.taAdindtl);
  DBGrid1.SelectedIndex := 0;
end;

procedure TfmAdin.taAdinDtlBeforeDelete(DataSet: TDataSet);
begin
  if dsWatch.State <> dsBrowse then
     if MessageDlg('Hapus record ... ?',mtConfirmation,[mbYes, mbNo], 0) <> mrYes
then abort;
end;

procedure TfmAdin.FormShow;
begin
  DBSetPath(dmInventory);                          { set databasename to pathdata
}

  with dmInventory do begin
    taBrg       := OpenTable('TBARANG' , '', fmAdin);
    taGdgBrg.Open;
```

```
    taAdin.Open;
    taAdindtl.Open;

     DBLoadLabel(dmInventory.taAdindtl);                    { field names's short
description used to DBGrid Title }

    taAdinDtlKODEBRG.OnChange := taAdinDtlKODEBRGChange;
    taAdindtl.OnNewRecord     := taAdindtlNewRecord;
    taAdindtl.BeforeDelete    := taAdinDtlBeforeDelete;
    taAdindtl.OnCalcFields    := taAdindtlCalcFields;
  end;
end;

procedure TfmAdin.FormClose;
begin
  dmInventory.taAdindtl.Close;
  dmInventory.taAdin.Close;
  taBrg.Close;
  DMInventory.taGdgBrg.Close;
end;

procedure TfmAdin.KNavAddClick;
begin
  with dmInventory do begin
    taAdin.Append;
    DBInitValue(taAdin);
    taAdinNOMER.Value    := DBGetCounter('AI', fmAdin);
    taAdinNAMAUSER.Value := UserLogin;
    deTanggal.SetFocus;
  end;
end;

procedure TfmAdin.KNavCancelClick(Sender: TObject);
begin
  with dmInventory do begin
    if dsWatch.State = dsInsert then
      taAdin.Post;

    taAdin.Cancel;
    taAdindtl.CancelUpdates;
  end;
  deTanggal.SetFocus;
end;

procedure TfmAdin.KNavSaveClick;
begin
  KNav.SetFocus;

  with dmInventory do
  try
```

```
  taAdindtl.DisableControls;

  { *** checking *** }
   if taAdinTANGGAL.IsNull              then DispErr('Tanggal tidak boleh kosong',
deTanggal);

  taAdinDtl.First;
  while not taAdinDtl.EOF do
  begin
    if taAdindtlKODEBRG.IsNull                  then DispErr('Kode Barang tidak
boleh kosong', DBGrid1);
     if not taBrg.FindKey([taAdindtlKODEBRG.Value]) then DispErr('Kode Barang
tidak ditemukan', DBGrid1);

    if not taGdgBrg.FindKey([taAdinDtlKODEBRG.Value]) then
     DispErr('Barang ' + taAdinDtlKODEBRG.Value + ' tidak ditemukan di gudang ',
DBGrid1);

     if taAdindtlJUMLAH.IsNull                  then DispErr('Jumlah tidak boleh
kosong', DBGrid1);
    if taAdindtlJUMLAH.Value < 1              then DispErr('Jumlah tidak boleh < 0',
DBGrid1);

     if taAdindtlSATUAN.IsNull                  then DispErr('Satuan tidak boleh
kosong', DBGrid1);
     if taAdinDtlHARGA.IsNull                   then DispErr('Harga tidak boleh
kosong', DBGrid1);
    if taAdinDtlHARGA.Value < 0               then DispErr('Harga tidak boleh < 0',
DBGrid1);

    taAdinDtl.Next;
  end;

  taAdinNAMAUSER.Value := UserLogin;
  taAdinTOTAL.Value    := subtotal;

  UpdateStok;

  DBInitValue2(taAdin);
  taAdin.Post;
  taAdindtl.CommitUpdates;
 finally
  taAdindtl.First;
  taAdindtl.EnableControls;
 end;
 deTanggal.SetFocus;
end;

procedure TfmAdin.UpdateStok;
begin
```

```
  with dmInventory do
  begin
    taAdinDtl.DisableControls;
    taAdinDtl.First;
    While Not taAdinDtl.EOF do
    Begin
      // Update GdgBrg
      taGdgBrg.FindKey([taAdinDtlKODEBRG.Value]);
      taGdgBrg.Edit;
    taGdgBrg.FieldByName('SALDO').AsFloat :=
taGdgBrg.FieldByName('SALDO').AsFloat +
                                    taAdinDtlJUMLAH.Value;
      taGdgBrg.Post;

      taAdinDtl.Next;
    end;
    taAdinDtl.EnableControls;
  end;
end;

procedure TfmAdin.DBGrid1EditButtonClick;
begin
  with dmInventory do
  begin
    fmBrowse.brwTBARANG(DBGrid1.SelectedField.AsString, nil);
    taBrg.FindKey([fmBrowse.StringGrid1.Cells[1, 0]]);
    taAdindtl.Edit;
    taAdinDtlKODEBRG.Value := fmBrowse.StringGrid1.Cells[1, 0];
  end;
end;

procedure TfmAdin.dsWatchStateChange;
begin
  DBgrid1.ReadOnly := dsWatch.State = dsBrowse;

  if dsWatch.State = dsBrowse then
    DBgrid1.Options := Dbgrid1.Options - [dgEditing]
  else
    DBgrid1.Options := Dbgrid1.Options + [dgEditing];
end;

procedure TfmAdin.dsDetailDataChange;
var nSatuan : word;
begin
  dsDetail.OnDataChange := nil;

  try
    with dmInventory do
    begin
      { cari posisi kolom untuk field satuan, ini penting
```

```
        soalnya kolom dapat dipindah-pindah }
        for nSatuan := 0 to DBgrid1.FieldCount - 1 do if DBgrid1.Fields[nSatuan] =
taAdindtlSATUAN then break;
     DBGrid1.Columns[nSatuan].PickList.Clear;

DBGrid1.Columns[nSatuan].PickList.Add(taBrg.FieldByName('SATUAN').AsString
);


     subtotal       := DBGetTotal(taAdindtlTOTAL);
     Panel1.Caption := format('%m ', [subtotal]);
   end;
  except
  end;

  dsDetail.OnDataChange := dsDetailDataChange;
end;

procedure TfmAdin.Add1Click(Sender: TObject);
begin
  dmInventory.taAdinDtl.Insert;
end;

procedure TfmAdin.Edit1Click(Sender: TObject);
begin
  dmInventory.taAdinDtl.Edit;
end;

procedure TfmAdin.Delete1Click(Sender: TObject);
begin
  dmInventory.taAdinDtl.Delete;
end;

procedure TfmAdin.Simpan1Click(Sender: TObject);
begin
  if dsDetail.State in [dsEdit, dsInsert] then dmInventory.taAdinDtl.Post;
end;

procedure TfmAdin.Batal1Click(Sender: TObject);
begin
  dmInventory.taAdinDtl.Cancel;
end;

procedure TfmAdin.PopupMenu1Popup(Sender: TObject);
begin
  if dsWatch.State = dsBrowse then abort;
end;

procedure TfmAdin.KNavPreviewClick(Sender: TObject);
begin
```

```
  with dmInventory do
  Begin
    Crpe.Clear;
    Crpe.ReportName := PathExe + 'Report\ntStokIn.rpt';
    Crpe.Tables.Retrieve;
    Crpe.Tables[0].Path := PathData;
    Crpe.Tables.Propagate := True;
    Crpe.WindowStyle.Title := Caption;
    Crpe.WindowSize.Height := Screen.Height;
    Crpe.WindowSize.Width := Screen.Width;
    Crpe.WindowSize.Left := 0;
    Crpe.WindowSize.Top := 0;
    Crpe.ParamFields.Retrieve;
    Crpe.ParamFields[0].Value := UserLogin;
    Crpe.ParamFields[1].Value := taADINNOMER.Value;
    Crpe.Execute;
  end;
end;

procedure TfmAdin.KNavSearchClick(Sender: TObject);
begin
  fmSearch.LoadSearch(dmInventory.taAdin);
end;

procedure TfmAdin.DBGrid1KeyPress(Sender: TObject; var Key: Char);
begin
  if DBGrid1.SelectedField.FieldName='KODEBRG' then
    Key:=UpCase(Key);
end;

end.
```

unit UAdou;

```
interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  Menus, Grids, DBGrids, ComCtrls, ZKNavigator, ExtCtrls, StdCtrls,
  DBCtrls, Mask, Db, DBTables, ToolEdit, RXDBCtrl;

type
  TfmAdou = class(TForm)
    DBGrid1: TDBGrid;
    Bevel1: TBevel;
    dsWatch: TDataSource;
    Panel3: TPanel;
    Label1: TLabel;
    Label2: TLabel;
```

```
    Panel2: TPanel;
    DBText1: TDBText;
    Label3: TLabel;
    Panel4: TPanel;
    Label4: TLabel;
    KNav: TK2Navigator;
    Panel1: TPanel;
    dsDetail: TDataSource;
    deTanggal: TDBDateEdit;
    PopupMenu1: TPopupMenu;
    Add1: TMenuItem;
    Edit1: TMenuItem;
    Delete1: TMenuItem;
    Simpan1: TMenuItem;
    Batal1: TMenuItem;
    deketerangan: TDBMemo;
    procedure KNavAddClick(Sender: TObject);
    procedure KNavCancelClick(Sender: TObject);
    procedure FormShow(Sender: TObject);
    procedure FormClose(Sender: TObject; var Action: TCloseAction);
    procedure DBGrid1EditButtonClick(Sender: TObject);
    procedure KNavSaveClick(Sender: TObject);
    procedure dsWatchStateChange(Sender: TObject);
    procedure dsDetailDataChange(Sender: TObject; Field: TField);
    procedure Add1Click(Sender: TObject);
    procedure Edit1Click(Sender: TObject);
    procedure Delete1Click(Sender: TObject);
    procedure Simpan1Click(Sender: TObject);
    procedure Batal1Click(Sender: TObject);
    procedure PopupMenu1Popup(Sender: TObject);
    procedure KNavPreviewClick(Sender: TObject);
    procedure KNavSearchClick(Sender: TObject);
    procedure DBGrid1KeyPress(Sender: TObject; var Key: Char);
  private
    taBrg    : TTable;
    subtotal : currency;

    procedure taAdoudtlKODEBRGChange(Sender: TField);
    procedure taAdouDtlNewRecord(DataSet: TDataSet);
    procedure taAdouDtlBeforeDelete(DataSet: TDataSet);
    procedure taAdouDtlCalcFields(DataSet: TDataSet);
    procedure UpdateStok;
  public
    { Public declarations }
  end;

var
  fmAdou: TfmAdou;

implementation
```

```
uses UdmInventory, AllUnits, UBrowse, USearch;

{$R *.DFM}

procedure TfmAdou.taAdoudtlKODEBRGChange(Sender: TField);
begin
  with dmInventory do
  begin
    taAdoudtlKODEBRG.OnChange := nil;

    if not taBrg.FindKey([sender.asString]) then DBGrid1EditButtonClick(nil);
    taAdoudtl.Edit;
    taAdouDtlKODEBRG.Value := taBrg.FieldByName('KODE').asString;
    taAdouDtlNAMABRG.Value := taBrg.FieldByName('NAMA').asString;
    taAdouDtlSATUAN.Value  := taBrg.FieldByName('SATUAN').asString;
    taAdouDtlJUMLAH.Value  := 1;
    taAdouDtlHARGA.Value   := 0;
    taAdoudtl.Post;

    taAdoudtlKODEBRG.OnChange := taAdoudtlKODEBRGChange;
  end;
end;

procedure TfmAdou.taAdouDtlCalcFields;
begin
  with dmInventory do
taAdouDtlTOTAL.Value:=taAdouDtlJUMLAH.Value*
taAdouDtlHARGA.Value;
end;

procedure TfmAdou.taAdouDtlNewRecord;
begin
  DBInitValue(dmInventory.taAdoudtl);
  DBGrid1.SelectedIndex := 0;
end;

procedure TfmAdou.taAdouDtlBeforeDelete(DataSet: TDataSet);
begin
  if dsWatch.State <> dsBrowse then
    if MessageDlg('Hapus record ... ?',mtConfirmation,[mbYes, mbNo], 0) <> mrYes
then abort;
end;

procedure TfmAdou.FormShow;
begin
  DBSetPath(dmInventory);

  with dmInventory do begin
    taBrg       := OpenTable('TBARANG' , '', fmAdou);
```

```
    taGdgBrg.Open;
    taAdou.Open;
    taAdoudtl.Open;

    DBLoadLabel(taAdoudtl);

    taAdouDtlKODEBRG.OnChange := taAdoudtlKODEBRGChange;
    taAdoudtl.OnNewRecord     := taAdoudtlNewRecord;
    taAdoudtl.BeforeDelete    := taAdouDtlBeforeDelete;
    taAdoudtl.OnCalcFields    := taAdoudtlCalcFields;
  end;
end;

procedure TfmAdou.FormClose;
begin
  dmInventory.taAdou.Close;
  dmInventory.taAdoudtl.Close;
  DMInventory.taGdgBrg.Close;
  taBrg.Close;
end;

procedure TfmAdou.KNavAddClick;
begin
  with dmInventory do begin
    taAdou.Append;
    DBInitValue(taAdou);
    taAdouNOMER.Value    := DBGetCounter('AO', fmAdou);
    taAdouNAMAUSER.Value := UserLogin;
    deTanggal.SetFocus;
  end;
end;

procedure TfmAdou.KNavCancelClick(Sender: TObject);
begin
  KNav.SetFocus;
  with dmInventory do begin
    if dsWatch.State = dsInsert then
      taAdou.Post;

    taAdou.Cancel;
    taAdoudtl.CancelUpdates;
  end;
end;

procedure TfmAdou.KNavSaveClick;
begin
  KNav.SetFocus;

  with dmInventory do
  try
```

```
  taAdoudtl.DisableControls;

  { *** checking *** }
   if taAdouTANGGAL.IsNull              then DispErr('Tanggal tidak boleh kosong',
deTanggal);

  taAdouDtl.First;
  while not taAdouDtl.EOF do
  begin
    if taAdoudtlKODEBRG.IsNull                  then DispErr('Kode Barang tidak
boleh kosong', DBGrid1);
     if not taBrg.FindKey([taAdoudtlKODEBRG.Value]) then DispErr('Kode Barang
tidak ditemukan', DBGrid1);

    if not taGdgBrg.FindKey([taAdouDtlKODEBRG.Value]) then
     DispErr('Barang ' + taAdouDtlKODEBRG.Value + ' tidak ditemukan di gudang ',
DBGrid1);

    if taGdgBrg.FieldByName('SALDO').AsInteger - taAdouDtlJUMLAH.Value < 0
then
     DispErr('Stok barang ' + taAdoudtlKODEBRG.Value + ' tidak mencukupi (' +
          FloatToStr(taGdgBrg.FieldByName('SALDO').AsInteger -
taAdouDtlJUMLAH.Value) + ')', DBGrid1);

     if taAdoudtlJUMLAH.IsNull                  then DispErr('Jumlah tidak boleh
kosong', DBGrid1);
    if taAdoudtlJUMLAH.Value < 1              then DispErr('Jumlah tidak boleh <
0', DBGrid1);

    if taAdoudtlSATUAN.IsNull                  then DispErr('Satuan tidak boleh
kosong', DBGrid1);
     if taAdouDtlHARGA.IsNull                  then DispErr('Harga tidak boleh
kosong', DBGrid1);
    if taAdouDtlHARGA.Value < 0               then DispErr('Harga tidak boleh < 0',
DBGrid1);

    taAdouDtl.Next;
  end;

  taAdouNAMAUSER.Value := UserLogin;
  taAdouTOTAL.Value    := subtotal;

  UpdateStok;

  DBInitValue2(taAdou);
  taAdou.Post;
  taAdoudtl.CommitUpdates;
 finally
  taAdoudtl.First;
  taAdoudtl.EnableControls;
```

```
  end;
  deTanggal.SetFocus;
end;

procedure TfmAdou.UpdateStok;
begin
  with dmInventory do
  begin
    taAdouDtl.DisableControls;
    taAdouDtl.First;
    While Not taAdouDtl.EOF do
    Begin
      // Update GdgBrg
      taGdgBrg.FindKey([taAdouDtlKODEBRG.Value]);
      taGdgBrg.Edit;
   taGdgBrg.FieldByName('SALDO').AsFloat :=
taGdgBrg.FieldByName('SALDO').AsFloat -
                                      taAdinDtlJUMLAH.Value;
      taGdgBrg.Post;

      taAdouDtl.Next;
    end;
    taAdouDtl.EnableControls;
  end;
end;

procedure TfmAdou.DBGrid1EditButtonClick;
begin
  with dmInventory do
  begin
    fmBrowse.brwTBARANG(DBGrid1.SelectedField.AsString, nil);
    taBrg.FindKey([fmBrowse.StringGrid1.Cells[0, 1]]);
    taAdouDtl.Edit;
    taAdouDtlKODEBRG.Value := fmBrowse.StringGrid1.Cells[1, 0];
  end;
end;

procedure TfmAdou.dsWatchStateChange;
begin
  DBgrid1.ReadOnly := dsWatch.State = dsBrowse;

  if dsWatch.State = dsBrowse then
    DBgrid1.Options := Dbgrid1.Options - [dgEditing]
  else
    DBgrid1.Options := Dbgrid1.Options + [dgEditing];
end;

procedure TfmAdou.dsDetailDataChange;
var nSatuan : word;
begin
```

```
  dsDetail.OnDataChange := nil;

  try
    with dmInventory do
    begin
      { cari posisi kolom untuk field satuan, ini penting
        soalnya kolom dapat dipindah-pindah }
        for nSatuan := 0 to DBgrid1.FieldCount - 1 do if DBgrid1.Fields[nSatuan] =
taAdoudtlSATUAN then break;
      DBGrid1.Columns[nSatuan].PickList.Clear;

DBGrid1.Columns[nSatuan].PickList.Add(taBrg.FieldByName('SATUAN').AsString
);


      subtotal       := DBGetTotal(taAdoudtlTOTAL);
      Panel1.Caption := format('%m ', [subtotal]);
    end;
  except
  end;

  dsDetail.OnDataChange := dsDetailDataChange;
end;

procedure TfmAdou.Add1Click(Sender: TObject);
begin
  dmInventory.taAdoudtl.Insert;
end;

procedure TfmAdou.Edit1Click(Sender: TObject);
begin
  dmInventory.taAdoudtl.Edit;
end;

procedure TfmAdou.Delete1Click(Sender: TObject);
begin
  dmInventory.taAdoudtl.Delete;
end;

procedure TfmAdou.Simpan1Click(Sender: TObject);
begin
  if dsDetail.State in [dsEdit, dsInsert] then dmInventory.taAdoudtl.Post;
end;

procedure TfmAdou.Batal1Click(Sender: TObject);
begin
  dmInventory.taAdoudtl.Cancel;
end;

procedure TfmAdou.PopupMenu1Popup(Sender: TObject);
```

```
begin
  if dsWatch.State = dsBrowse then abort;
end;

procedure TfmAdou.KNavPreviewClick(Sender: TObject);
begin
  with dmInventory do
  Begin
     Crpe.Clear;
     Crpe.ReportName := PathExe + 'Report\ntStokOut.rpt';
     Crpe.Tables.Retrieve;
     Crpe.Tables[0].Path := PathData;
     Crpe.Tables.Propagate := True;
     Crpe.WindowStyle.Title := Caption;
     Crpe.WindowSize.Height := Screen.Height;
     Crpe.WindowSize.Width := Screen.Width;
     Crpe.WindowSize.Left := 0;
     Crpe.WindowSize.Top := 0;
     Crpe.ParamFields.Retrieve;
     Crpe.ParamFields[0].Value := UserLogin;
     Crpe.ParamFields[1].Value := taADOUNOMER.Value;
     Crpe.Execute;
  end;
end;

procedure TfmAdou.KNavSearchClick(Sender: TObject);
begin
  fmSearch.LoadSearch(dmInventory.taAdou);
end;
procedure TfmAdou.DBGrid1KeyPress(Sender: TObject; var Key: Char);
begin
  if DBGrid1.SelectedField.FieldName='KODEBRG' then
    Key:=UpCase(Key);
end;
end.
```

Unit UBayarHutang;

```
Interface

Uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  ExtCtrls, ZKNavigator, StdCtrls, Mask, DBCtrls, ToolEdit, RXDBCtrl,
  ComCtrls, Grids, DBGrids, Db, DBTables, Menus;

Type
  TfmBayarHutang = Class(TForm)
    K2Navigator1: TK2Navigator;
    Label1: TLabel;
```

Label2: TLabel;
Label3: TLabel;
Label4: TLabel;
DBEdit1: TDBEdit;
RxDBComboEdit2: TRxDBComboEdit;
DBEdit3: TDBEdit;
Label11: TLabel;
DBGrid1: TRxDBGrid;
StatusBar1: TStatusBar;
Label16: TLabel;
dsNavigator: TDataSource;
dsDetail: TDataSource;
DBCheckBox2: TDBCheckBox;
Panel2: TPanel;
Bevel1: TBevel;
PopupDtl: TPopupMenu;
Tambah1: TMenuItem;
MenuItem1: TMenuItem;
Hapus1: TMenuItem;
Simpan1: TMenuItem;
Batal1: TMenuItem;
DBDateEdit1: TDBDateEdit;
Label14: TLabel;
Label15: TLabel;
DBEdit9: TDBEdit;
DBEdit10: TDBEdit;
Bevel2: TBevel;
Label18: TLabel;
stTotBayar: TStaticText;
dbedit6: TDBMemo;
Label9: TLabel;
DBEdit5: TDBEdit;
Label10: TLabel;
DBDateEdit2: TDBDateEdit;
procedure FormShow(Sender: TObject);
procedure FormClose(Sender: TObject; Var Action: TCloseAction);
procedure dsNavigatorDataChange(Sender: TObject; Field: TField);
procedure RxDBComboEdit2ButtonClick(Sender: TObject);
procedure DBEdit3Enter(Sender: TObject);
procedure K2Navigator1AddClick(Sender: TObject);
procedure dsNavigatorStateChange(Sender: TObject);
procedure K2Navigator1SaveClick(Sender: TObject);
procedure DBGrid1EditButtonClick(Sender: TObject);
procedure dsDetailDataChange(Sender: TObject; Field: TField);
procedure K2Navigator1EditClick(Sender: TObject);
procedure K2Navigator1CancelClick(Sender: TObject);
procedure Tambah1Click(Sender: TObject);
procedure Hapus1Click(Sender: TObject);
procedure Batal1Click(Sender: TObject);
procedure FormCloseQuery(Sender: TObject; Var CanClose: Boolean);

```
    procedure MenuItem1Click(Sender: TObject);
    procedure Simpan1Click(Sender: TObject);
    procedure K2Navigator1PreviewClick(Sender: TObject);
    procedure dsDetailUpdateData(Sender: TObject);
    procedure DBGrid1Exit(Sender: TObject);
    procedure DBGrid1ShowEditor(Sender: TObject; Field: TField;
      Var AllowEdit: Boolean);
    procedure K2Navigator1SearchClick(Sender: TObject);
    procedure DBEdit2Exit(Sender: TObject);
    procedure FormKeydown(Sender: TObject; var Key: Word;
      Shift: TShiftState);
    procedure dsNavigatorUpdateData(Sender: TObject);
    procedure DBGrid1DrawColumnCell(Sender: TObject; const Rect: TRect;
      DataCol: Integer; Column: TColumn; State: TGridDrawState);
  private
    tot_bayar_hut:currency;
    taHutTemp: TTable;
    LastValue: string;
    procedure FilterRecordDetail(DataSet: TDataSet; Var Accept: Boolean);
    procedure display(obj: Tobject);
    procedure UpdatePLHDetil;
    procedure HitungSubtotKurs;
    procedure AfterScroll(DataSet:TDataSet);
  public
  end;

Var
  fmBayarHutang: TfmBayarHutang;

Implementation

Uses uDMHutang, UBrowse, AllUnits, USearch;

{$R *.DFM}
procedure TfmBayarHutang.AfterScroll(DataSet:TDataSet);
begin
  Hitungsubtotkurs;
end;

procedure TfmBayarHutang.UpdatePLHDetil;
begin
  with dmHutang do
  begin
    taPLHDTL.DisableControls;
    GetTotalPlh(taPLHDTL);
    taPLHDTL.EnableControls;
  end;
end;

procedure TfmBayarHutang.display(obj: Tobject);
```

```
begin
  Panel2.Caption := format('%m', [dmHutang.nTotByr]);
end;

procedure TfmBayarHutang.FormShow(Sender: TObject);
begin
  DBSetPath(dmHutang);
  dmHutang.nRunDtl := display;
  taHutTemp := OpenTable('THUT', 'NOMER', fmBayarHutang);
  with dmHutang do
  begin
    taPLH.AfterScroll := AfterScroll;
    taPLHLookup.Open;
    taPLH.Open;
    taPLHDtl.Open;
    taHut.Open;
    taSupl.Open;
  end;

  if dmHutang.taPlhdtl.Active then
  begin
    dmHutang.GetTotalPlh(dmHutang.taPlhdtl);
    StatusBar1.Panels[0].Text := UserLogin;
  end;
  hitungsubtotkurs;
end;

procedure TfmBayarHutang.FormClose(Sender: TObject; var Action: TCloseAction);
begin
  with dmHutang do
  begin
    taPLHDtl.AfterDelete := nil;
    taPLH.Close;
    taPLHLookup.Close;
    taPLH.AfterScroll:=nil;
    taPLHDtl.Close;
    taHut.Close;
    taSupl.Close;
  end;

  taHutTemp.Close;
end;

procedure TfmBayarHutang.dsNavigatorDataChange(Sender: TObject; Field:
TField);
begin
  with dmHutang do
  begin
    if taPlhdtl.Active then
      GetTotalPlh(dmHutang.taPlhdtl);
```

```
    if (taPLH.State in [dsInsert, dsEdit]) then
    begin
      HitungSubTotKurs;
      if Field = taPLHKodeSupl then
      begin
        if not taSupl.FindKey([taPLHKodeSupl.VALUE]) then
          rxdbcomboedit2buttonclick(nil)
        else
          taPLHNAMASUPL.Value := taSupl.FieldByName('NAMA').AsString;
      end;
    end; // end Tabel State
  end; // end with DMHutang
end;

procedure TfmBayarHutang.RxDBComboEdit2ButtonClick(Sender: TObject);
begin
  fmBrowse.brwTSUPL(RxDBComboEdit2.Text, Nil);
  if fmBrowse.ModalResult = mrOk then
  begin
    dmHutang.taPLHKodeSupl.Value := fmBrowse.StringGrid1.Cells[1, 0];
    dmHutang.taPLHNamaSupl.Value := fmBrowse.StringGrid1.Cells[1, 1];
  end;
end;

procedure TfmBayarHutang.DBEdit3Enter(Sender: TObject);
begin
  if dbedit3.text <> '' then
    Postmessage(handle, wm_nextdlgctl, 0, 0);
end;

procedure TfmBayarHutang.K2Navigator1AddClick(Sender: TObject);
begin
  dmHutang.taPLH.Append;
  dmHutang.taPLHNAMAUSER.Value := UserLogin;
  DBDateEdit1.SetFocus;
end;

procedure TfmBayarHutang.dsNavigatorStateChange(Sender: TObject);
begin
  dbgrid1.Readonly := not (dsNavigator.State In [dsEdit, dsinsert]);
  RxDBComboEdit2.Button.Enabled := (dsNavigator.State In [dsEdit, dsInsert]);
  if dsNavigator.State In [dsEdit, dsInsert] then
  begin
    DBGrid1.PopupMenu := PopupDtl;
    Dbgrid1.OnEditButtonClick := DBGrid1EditButtonClick;
  end
  Else
  begin
    DbGrid1.PopupMenu := Nil;
```

```
   Dbgrid1.OnEditButtonClick := Nil;
  end;
end;

procedure TfmBayarHutang.K2Navigator1SaveClick(Sender: TObject);
Var
  i: Integer;
  selisihprebayar,Jumlah, Lebih: Currency;
  TotBayar: Currency;
  Jumlah_Lebih: Currency; // Jumlah dan Lebih di Master
begin
  K2Navigator1.SetFocus;

  with DMHutang do
  begin
     if taPLHTanggal.IsNull                    then DispErr('Tanggal belum diisi',
DBDateEdit1);
     if taPLHTANGGAL.Value <> Date             then DispErr('Tanggal harus =
tanggal sekarang', DBDateEdit1);

    if taPLHKODESUPL.IsNull                    then DispErr('Kode Supplier belum diisi',
rxDBComboEdit2);
     if not taSupl.FindKey([taPLHKODESUPL.VALUE]) then DispErr(Format('Kode
Supplier %s tidak ada/tidak ditemukan', [taPLHKODESUPL.VALUE]),
rxDBComboEdit2);

    if taPLHKELEBIHAN.Value < 0                then Disperr('Nilai kelebihan tidak
boleh bernilai negatif', DBEdit10);

    if taPLH.State=dsInsert then
    begin
      taPLHLookup.Filter:='VOID=false';
      taPLHLookup.Filtered:=true;
      taPLHLookup.Refresh;
      try
        if (DBEdit5.Text<>'') and taPLHLookup.FindKey([DBEdit5.Text]) then
          Disperr('Nomer giro sudah pernah dipakai', DBEdit5);
      finally
        taPLHLookup.Filtered:=false;
      end;
    end else
    begin
      taPLHLookup.Filter:='NOGIRO='"+DBEdit5.Text+'" and void=false';
      taPLHLookup.Filtered:=true;
      taPLHLookup.Refresh;
      try
        I:=0;
        taPLHLookup.First;
        while not taPLHLookup.EOF do
        begin
```

```
        Inc(I);
        taPLHLookup.Next;
      end;

      if ((I > 0) and (LastValue <> Dbedit5.Text)) then
        Disperr('Nomer giro sudah pernah dipakai', DBEdit5);

    finally
      taPLHLookup.Filtered:=false;
    end;
  end;

  taPLHDtl.DisableControls;
  TaPLHDtl.First;
  while not taPLHDtl.EOF do
  begin
    if taPLHDtlBAYAR.Value > taPLHDtlSISAnotA.Value then
    begin
      taPLHDtl.EnableControls;
       DispErr(Format('No nota %s kelebihan pembayaran', [taPLHDtlNonota.Value]),
DBGrid1);
    end;

    if ((taPLHDtlID.Value <> 'BLKR') and (taPLHDtlID.Value <> 'RBKR')) then
    begin
      taPLHDtl.EnableControls;
           DispErr(Format('Entry ID %s tidak ada diantara ID BLKR dan RBKR',
[taPLHDtlID.Value]), DBGrid1);
    end;

    if not taHut.FindKey([dmHutang.taPLHDtlNoNotA.Value]) then begin
      taPLHDtl.EnableControls;
                DispErr(Format('Nomer nota %s tidak ditemukan di hutang',
[taPLHDtlNOnotA.Value]), DBGrid1);
    end else
    begin
      if (taHutID.Value <> taPLHDtlID.Value) then begin
        taPLHDtl.EnableControls;
          Disperr(Format('Nomer nota %s dengan ID=%s tidak ditemukan di hutang',
[taPLHDtlNOnotA.Value, taPLHDtlID.Value]), DBGrid1);
      end;

      if (taHutKodeSupl.Value <> taPLHKodeSupl.Value) then
           DispErr(Format('Nomer nota %s milik supplier %s, bukan milik supplier
%s',[taPLHDTLNonota.Value,taHutKodeSUpl.Value,taPLHKodeSupl.Value]),dbgrid
1);
    end;

    // perpindahan akibat debet note
```

```
            taHut.Filter := '(ID='" + taPLHDtlID.Value + "') and (NOMER='" +
taPLHDtlNOnotA.Value + "')';
      taHut.Filtered := True;
      taHut.First;
      taHut.FindKey([taPLHDtlNonota.Value]);

      if taHutID.Value = 'RBKR' then
      begin
        selisihprebayar := (taHutDEBET.Value - taPLHDtlBAYAR.Value);
        if (selisihprebayar < 0) then
        begin
          taPLHDtl.EnableControls;
          Disperr(Format('No nota %s kelebihan pembayaran', [taPLHDtlNonota.Value]),
DBGrid1);
        end;
      end;

      if taHutID.Value = 'BLKR' then
      begin
        selisihprebayar:=(taHutKREDIT.Value - taPLHDtlBAYAR.Value);
        if (selisihprebayar < 0) then
        begin
          taPLHDtl.EnableControls;
          Disperr(Format('No nota %s kelebihan pembayaran', [taPLHDtlNonota.Value]),
DBGrid1);
        end;
      end;

      taHut.Filtered := False;

      taPLHDtl.Next;
    end; // end Loop cek detil

    TotBayar := 0;
    taPLHdtl.First;
    while not taPLHdtl.EOF do
    begin
      TotBayar := TotBayar + taPLHDtlSubTotal.Value ;
      taPLHDtl.Next;
    end;

    Jumlah := taPLHJumlah.Value;
    Lebih := taPLHKelebihan.Value;
    Jumlah_Lebih := (Jumlah - Lebih);

    if (Jumlah_Lebih <> TotBayar) then begin
      taPLHDtl.EnableControls;
          Disperr(Format('Pembayaran hutang tidak balance, total pembayaran %m',
[TotBayar]), DBEdit9);
    end;
```

```
    //Simpan ketabel2 bersangkutan
    taPlhdtl.First;
    tot_bayar_hut := 0;
    while not taPlhdtl.EOF do
    begin
      if taHut.FindKey([taPlhdtlNonota.Value]) then
      begin
        taHut.Edit;
        taHutBAYAR.Value := taHutBAYAR.Value + taPlhdtlBAYAR.Value;
        taHut.Post;
      end;
      taPlhdtl.Next; // Proses next Record Detil
    end;

    if taPLHKELEBIHAN.Value <> 0 then
    begin
      taHut.Append;
      taHutNOMER.Value      := 'KL'+ Copy(taPlhNOMER.Value,3,10);
      taHutTANGGAL.Value    := taPlhTANGGAL.Value;
      taHutKODESUPL.Value   := taPlhKODESUPL.Value;
      taHutNAMASUPL.Value   := taPlhNAMASUPL.Value;
      taHutID.Value         := 'RBKR';
      taHutNOREFF.Value     := taPlhNOMER.Value;
      taHutIDREFF.Value     := 'PLH';
      taHutKETERANGAN.Value := 'Kelebihan Pembayaran';
      taHutTERM.Value       := 0;
      taHutDEBET.Value      := taPlhKELEBIHAN.Value;
      taHutKREDIT.Value     := 0;
      taHutBAYAR.Value      := 0;
      taHut.Post;
    end;

    DBInitValue2(taPLH);
    taPLH.Post;
    taPLHDtl.CommitUpdates;
    taPLHDtl.EnableControls;
    HitungSubTotKurs;
  end; // end with DMHutang
  DBEdit1.SetFocus;
 end; // end procedure

 procedure TfmBayarHutang.FilterRecordDetail(DataSet: TDataSet; Var Accept:
 Boolean);
 Var ikut: boolean;
 begin
ikut:=(Dataset.fieldbyname('DEBET').asCurrency+
 Dataset.fieldbyname('KREDIT').asCurrency     >
 DataSet.Fieldbyname('BAYAR').asCurrency);
```

```
    Accept := (Dataset.FieldByName('ID').AsString = dmHutang.taPLHDtlID.Value)
 And
(Dataset.FieldByName('KODESUPL').AsString=
 dmHutang.taPLHKODESUPL.Value) And ikut;
 end;

 procedure TfmBayarHutang.DBGrid1EditButtonClick(Sender: TObject);
 var nVar, Bayar: Currency;
 begin
  nVar  := 0;
  Bayar := 0;

  if dmhutang.taPLHDtl.State = dsBrowse then
  begin
    if dmhutang.taPLHdtl.RecordCount > 0 then
      dmhutang.taPLHdtl.edit
    Else
      dmhutang.taPLHdtl.append;
  end;

  fmBrowse.brwTHutang(dbGrid1.InplaceEditor.Text, FilterRecordDetail);
   if (fmBrowse.ModalResult = mrOk) And (fmBrowse.StringGrid1.Cells[1, 0] <> '')
then
  begin
    dmHutang.taPLHDtlNOnotA.Value := fmBrowse.StringGrid1.Cells[1, 0];
    if taHutTemp.FindKey([dmHutang.taPLHDtlNOnotA.Value]) then
    begin
      if DMHutang.taPLHDtlID.Value = 'BLKR' then
        nVar := taHutTemp.FieldByName('Kredit').AsCurrency
      Else
        nVar := taHutTemp.FieldByName('Debet').AsCurrency;
      Bayar := taHutTemp.FieldByName('Bayar').AsCurrency;
    end;
    dmHutang.taPLHDtlTOTALnotA.Value := nVar;
    dmHutang.taPLHDtlSISAnotA.Value := nVar - Bayar;
    dmHutang.taPLHDtlBAYAR.Value := dmHutang.taPLHDtlSISAnotA.Value;
  end;
 end;

 procedure TfmBayarHutang.dsDetailDataChange(Sender: TObject; Field: TField);
 var nVar, Bayar: Currency;
 begin
  nVar  := 0;
  Bayar := 0;

  dsDetail.OnDataChange := Nil;
  with dmHutang do
  begin
    if (taPLHDtl.State In [dsInsert, dsEdit]) then
    begin
```

```
        if Field = TaPLHDtlNOnotA then
        begin
          if not taHut.FindKey([TaPLHDtlNOnotA.Value]) then
            DBGrid1EditButtonClick(Nil)
          Else
          begin
            if taHutTemp.FindKey([taPLHDtlNOnotA.Value]) then
            begin
              if taPLHDtlID.Value = 'BLKR' then
                nVar := taHutTemp.FieldByName('Kredit').AsCurrency
              Else
                nVar := taHutTemp.FieldByName('Debet').AsCurrency;
              Bayar := taHutTemp.FieldByName('Bayar').AsCurrency;
            end;
            taPLHDTLTOTALnotA.Value := nVar;
            taPLHDtlSISAnotA.Value := nVar - Bayar;
            taPLHDtlBAYAR.Value := dmHutang.taPLHDtlSISAnotA.Value;
          end;
        end;

        if Field = taPLHDtlBAYAR then
        begin
          if taPLHDTLBayar.Value > taPLHDtlSISAnotA.value then
          begin
            dsDetail.OnDataChange := dsDetailDataChange;
DispErr(Format('No nota %s kelebihan pembayaran',
          [taPLHDtlNonota.Value]), DBGrid1);
          end;
        end;

        if Field = taPLHDtlSISAnotA then
          taPLHDtlBAYAR.Value := taPLHDtlSISAnota.Value;

        if Field = taPLHDtlBAYAR then
          GetTotalPlh(taPlhdtl);
      end;
    end;
    dsDetail.OnDataChange := dsDetailDataChange;
  end;

  procedure TfmBayarHutang.K2Navigator1EditClick(Sender: TObject);
  begin
    if not Dmhutang.taPLHVOID.Value then
    begin
      LastValue:=DmHutang.taPLHNOGIRO.Value;
      DMHutang.TaPLH.Edit;
      DMHutang.taPLHdtl.Edit;
      dmHutang.taPLHNAMAUSER.Value := UserLogin;

      with DMHutang do
```

```
    begin
      taPLHDtl.DisableControls;
      taPLHDtl.First;
      while not taPLHDtl.Eof do
      begin
        if taHut.FindKey([taPLHDtlNOnotA.Value]) then
        begin
          taHut.Edit;
          taHutBAYAR.Value := taHutBAYAR.Value - taPLHDtlBAYAR.Value;
          tot_bayar_hut:= tot_bayar_hut+taPLHDtlBAYAR.value;
          taHut.Post;
        end;
        taPLHDtl.Next;
      end;
    end;
  end
  Else
    ShowMessage('Tidak dapat diedit, karena sudah void atau posted');
end;

procedure TfmBayarHutang.K2Navigator1CancelClick(Sender: TObject);
begin
  Case dsNavigator.State Of
    dsInsert:
      begin
        DmHutang.taPLHVoid.Value := True;
        DmHutang.taPLHdtl.CommitUpdates;
        DmHutang.taPLH.Post;
      end;
    dsEdit:
    begin
       if (MessageDlg('Data edit mau di void ?', mtConfirmation, [mbYes, mbNo], 0) =
mrYes) then
      begin
        DmHutang.taPLHdtl.CancelUpdates;
        DmHutang.taPLHVoid.Value := True;
        DmHutang.taPLH.Post;
      end Else
      begin
        DmHutang.taPLHdtl.CancelUpdates;
        DmHutang.taPLH.Cancel;
      end;

        with DMHutang do begin
          taPLHDtl.First;
          tot_bayar_hut:=0;
          while not taPLHDtl.Eof do
          begin
                  taHut.Filter := '(ID="' + taPLHDtlID.Value + '") and (NOMER="' +
taPLHDtlNOnotA.Value + '")';
```

```
          taHut.Filtered := True;
          taHut.First;
          while not taHut.Eof do begin
            taHut.Edit;
            taHutBAYAR.Value := taHutBAYAR.Value + taPLHDtlBAYAR.Value;
            tot_bayar_hut:= tot_bayar_hut+taPLHDtlBAYAR.Value;
            taHut.Post;
            taHut.Next;
          end;

          taHut.Filtered := False;
          taPLHDtl.Next;
        end;
      end;
    end; // end case EDIT
  end;
  HitungSubTotKurs;
end;

procedure TfmBayarHutang.Tambah1Click(Sender: TObject);
begin
  if (dsNavigator.State In [dsInsert, dsEdit]) then
    dmHutang.taPLHdtl.Append;
end;

procedure TfmBayarHutang.Hapus1Click(Sender: TObject);
begin
   if (dmHutang.taPLHDtl.RecordCount > 0) And (dsNavigator.State In [dsInsert,
dsEdit]) then
    if MessageDlg('Hapus record ?', mtConfirmation, [mbYes, mbNo], 0) = mrYes then
      dmHutang.taPLHdtl.Delete;
end;

procedure TfmBayarHutang.Batal1Click(Sender: TObject);
begin
  dmHutang.taPLHdtl.Cancel;
end;

procedure TfmBayarHutang.FormCloseQuery(Sender: TObject;
  Var CanClose: Boolean);
begin
  if (dsNavigator.state In [dsedit, dsInsert]) then
    CanClose := False
  Else
    CanClose := True;
end;

procedure TfmBayarHutang.MenuItem1Click(Sender: TObject);
begin
```

```
  if (dmHutang.taPLHDtl.RecordCount > 0) And (dsNavigator.State In [dsInsert,
dsEdit]) then
    dmHutang.taPLHdtl.Edit;
end;

procedure TfmBayarHutang.Simpan1Click(Sender: TObject);
begin
  if (dsDetail.State In [dsInsert, dsEdit]) then
    dmHutang.taPLHdtl.Post;
end;

procedure TfmBayarHutang.K2Navigator1PreviewClick(Sender: TObject);
begin
  with dmHutang do
  begin
    Preview('ntByrHut.rpt', taPLHNOMER.Value, Caption);
  end;
end;

procedure TfmBayarHutang.dsDetailUpdateData(Sender: TObject);
begin
  if dmHutang.taPLHDtl.State = dsInsert then
  begin
    dmHutang.GetTotalPlhUpdate(dmHutang.taPLHDtl);
  end;
end;

procedure TfmBayarHutang.DBGrid1Exit(Sender: TObject);
begin
  if dsDetail.State In [dsInsert, dsEdit] then
  begin
    dmHutang.taPLHDtl.Post;
    dmHutang.GetTotalPlh(dmHutang.taPLHDTL);
  end;
end;

procedure TfmBayarHutang.DBGrid1ShowEditor(Sender: TObject; Field: TField;
  Var AllowEdit: Boolean);
begin
  if dbGrid1.SelectedField = dmHutang.taPLHDtlTOTALnotA then Abort;
  if dbGrid1.SelectedField = dmHutang.taPLHDtlSISAnotA then Abort;
  if dbGrid1.SelectedField = dmHutang.taPLHDtlSUBTOTAL then Abort;
end;

procedure TfmBayarHutang.K2Navigator1SearchClick(Sender: TObject);
begin
  fmSearch.LoadSearch(dmHutang.taPLH);
end;

procedure TfmBayarHutang.DBEdit2Exit(Sender: TObject);
```

```
begin
  if dsNavigator.State In [dsInsert, dsEdit] then
    UpdatePLHDetil;
end;

procedure TfmBayarHutang.FormKeydown(Sender: TObject; var Key: Word;
  Shift: TShiftState);
begin
  if Key=VK_F1 then
    Application.HelpJump('HUTANG');
end;

procedure TfmBayarHutang.dsNavigatorUpdateData(Sender: TObject);
begin
  if dsNavigator.State in [dsEdit,dsInsert] then
    HitungSubTotKurs;
end;

procedure TfmBayarHutang.HitungSubtotKurs;
var subtotkurs : currency;
begin
  subtotkurs:=0;
  with dmHutang do
  begin
    if taPLH.Active then
       subtotkurs:=(taPLHJumlah.Value - taPLHKelebihan.Value);
    stTotBayar.Caption:=Format('%m',[subtotkurs]);
  end;
end;

procedure TfmBayarHutang.DBGrid1DrawColumnCell(Sender: TObject;
  const Rect: TRect; DataCol: Integer; Column: TColumn;
  State: TGridDrawState);
begin
  if not ((DBGrid1.DataSource.DataSet.FieldByName('ID').AsString='BLKR') or
        (DBGrid1.DataSource.DataSet.FieldByName('ID').AsString='SPKR'))
  then
    DBGrid1.Canvas.Font.Color := clRed
  else
    DBGrid1.Canvas.Font.Style := [];

  DBgrid1.DefaultDrawColumnCell(Rect, Datacol, Column, State);

end;

end.
```

Unit UBayarPiutang;

Interface

Uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  ExtCtrls, ZKNavigator, StdCtrls, Mask, DBCtrls, ToolEdit, RXDBCtrl,
  ComCtrls, Grids, DBGrids, Db, DBTables, Menus;

Type
  TfmBayarPiutang = Class(TForm)
    K2Navigator1: TK2Navigator;
    Label1: TLabel;
    Label2: TLabel;
    Label3: TLabel;
    Label4: TLabel;
    DBEdit1: TDBEdit;
    RxDBComboEdit2: TRxDBComboEdit;
    DBEdit3: TDBEdit;
    Label11: TLabel;
    DBEdit9: TDBEdit;
    Label14: TLabel;
    Label15: TLabel;
    DBEdit10: TDBEdit;
    DBGrid1: TRxDBGrid;
    StatusBar1: TStatusBar;
    Label16: TLabel;
    dsNavigator: TDataSource;
    dsDetail: TDataSource;
    DBCheckBox2: TDBCheckBox;
    Panel2: TPanel;
    Bevel1: TBevel;
    PopupDtl: TPopupMenu;
    Tambah1: TMenuItem;
    MenuItem1: TMenuItem;
    Hapus1: TMenuItem;
    Simpan1: TMenuItem;
    Batal1: TMenuItem;
    DBDateEdit1: TDBDateEdit;
    Bevel2: TBevel;
    Label18: TLabel;
    stTotBayar: TStaticText;
    dbedit6: TDBMemo;
    Label9: TLabel;
    DBEdit5: TDBEdit;
    Label10: TLabel;
    DBDateEdit2: TDBDateEdit;
    procedure FormShow(Sender: TObject);
    procedure FormClose(Sender: TObject; var Action: TCloseAction);
    procedure dsNavigatorDataChange(Sender: TObject; Field: TField);

```
    procedure RxDBComboEdit2ButtonClick(Sender: TObject);
    procedure DBEdit3Enter(Sender: TObject);
    procedure K2Navigator1AddClick(Sender: TObject);
    procedure dsNavigatorStateChange(Sender: TObject);
    procedure K2Navigator1SaveClick(Sender: TObject);
    procedure DBGrid1EditButtonClick(Sender: TObject);
    procedure dsDetailDataChange(Sender: TObject; Field: TField);
    procedure K2Navigator1EditClick(Sender: TObject);
    procedure K2Navigator1CancelClick(Sender: TObject);
    procedure Tambah1Click(Sender: TObject);
    procedure Hapus1Click(Sender: TObject);
    procedure Batal1Click(Sender: TObject);
    procedure FormCloseQuery(Sender: TObject; var CanClose: Boolean);
    procedure MenuItem1Click(Sender: TObject);
    procedure Simpan1Click(Sender: TObject);
    procedure AfterDel(DataSet: TDataSet);
    procedure AfterPost(DataSet: TDataSet);
    procedure K2Navigator1PreviewClick(Sender: TObject);
    procedure dsDetailUpdateData(Sender: TObject);
    procedure DBGrid1Exit(Sender: TObject);
    procedure DBGrid1ShowEditor(Sender: TObject; Field: TField;
      var AllowEdit: Boolean);
    procedure K2Navigator1SearchClick(Sender: TObject);
    procedure DBEdit2Exit(Sender: TObject);
    procedure FormKeydown(Sender: TObject; var Key: Word;
      Shift: TShiftState);
    procedure dsNavigatorUpdateData(Sender: TObject);
    procedure DBGrid1DrawColumnCell(Sender: TObject; const Rect: TRect;
      DataCol: Integer; Column: TColumn; State: TGridDrawState);
  private
    { Private declarations }
    LastValue: string;
    tot_bayar_piut: currency;
    procedure display(obj: Tobject);
    procedure UpdatePLPDetil;
    procedure HitungSubTotKurs;
    procedure AfterScroll(DataSet:TDataset);
  public
    { Public declarations }
  end;

var
  fmBayarPiutang: TfmBayarPiutang;


Implementation

Uses uDMHutang, UBrowse, AllUnits, USearch;

{$R *.DFM}
```

```
procedure TfmBayarPiutang.AfterScroll(DataSet:TDataset);
begin
  HitungSubtotkurs;
end;

procedure TfmBayarPiutang.UpdatePLPDetil;
begin
  with dmHutang do
  begin
    taPLPDTL.DisableControls;
    GetTotalPLP(taPLPDTL);
    taPLPDTL.EnableControls;
  end;
end;

procedure TfmBayarPiutang.AfterDel(DataSet: TDataSet);
begin
  dmHutang.GetTotalPLP(dmHutang.taPLPdtl);
end;

procedure TfmBayarPiutang.AfterPost(DataSet: TDataSet);
begin
  dmHutang.GetTotalPLP(dmHutang.taPLPdtl);
end;

procedure TfmBayarPiutang.display(obj: Tobject);
begin
  Panel2.Caption := format('%m', [dmHutang.nTotByr]);
end;

procedure TfmBayarPiutang.FormShow(Sender: TObject);
begin
  DBSetPath(dmHutang);
  K2Navigator1.DataSource := dsNavigator;
  with DMHutang do
  begin
    nRunDtl := display;
    taPLP.AfterScroll:=AfterScroll;
    taPLPDtl.AfterDelete := AfterDel;
    taPLPDtl.AfterDelete := AfterPost;
    taPLPLookup.Open;
    taPLP.Open;
    taPLPDtl.Open;
    taPiut.Open;
    taCust.Open;
    GetTotalPLP(taPLPdtl);
  end;
  StatusBar1.Panels[0].Text := UserLogin;
  hitungsubtotkurs;
end;
```

```
procedure TfmBayarPiutang.FormClose(Sender: TObject;
  var Action: TCloseAction);
begin
  with DMHutang do
  begin
    taPLPDtl.AfterDelete := Nil;
    taPLP.Close;
    taPLPLookup.Close;
    taPLP.AfterScroll:=nil;
    taPLPDtl.Close;
    taPiut.Close;
    taCust.Close;
  end;
  K2Navigator1.DataSource := Nil;
end;

procedure TfmBayarPiutang.dsNavigatorDataChange(Sender: TObject; Field:
TField);
begin
  with DMHutang do
  begin
    if taPLPdtl.Active then
      GetTotalPLP(dmHutang.taPLPdtl);

    if taPLP.State In [dsInsert, dsEdit] then
    begin
      HitungSubTotKurs;
      if Field = taPLPKodeCust then
      begin
        if Not taCust.FindKey([taPLPKodeCust.VALUE]) then
          rxdbcomboedit2buttonclick(Nil)
        else
          taPLPNAMACUST.Value := taCust.FieldByName('NAMA').AsString;
      end;
    end; // end if Tabel State
  end;
end;

procedure TfmBayarPiutang.RxDBComboEdit2ButtonClick(Sender: TObject);
begin
  fmBrowse.brwTCust(RxDBComboEdit2.Text, Nil);
  if fmBrowse.ModalResult = mrOk then
  begin
    dmHutang.taPLPKodeCust.Value := fmBrowse.StringGrid1.Cells[1, 0];
    dmHutang.taPLPNamaCust.Value := fmBrowse.StringGrid1.Cells[1, 1];
  end;
end;

procedure TfmBayarPiutang.DBEdit3Enter(Sender: TObject);
```

```
begin
  if dbedit3.text <> '' then
    Postmessage(handle, wm_nextdlgctl, 0, 0);
end;

procedure TfmBayarPiutang.K2Navigator1AddClick(Sender: TObject);
begin
  dmHutang.taPLP.Append;
  dmHutang.taPLPNAMAUSER.Value := UserLogin;
  DBDateEdit1.SetFocus;
end;

procedure TfmBayarPiutang.dsNavigatorStateChange(Sender: TObject);
begin
  dbgrid1.Readonly := Not (dsNavigator.State In [dsEdit, dsinsert]);
  RxDBComboEdit2.Button.Enabled := (dsNavigator.State In [dsEdit, dsInsert]);
  if dsNavigator.State In [dsEdit, dsInsert] then
  begin
    DBGrid1.PopupMenu := PopupDtl;
    Dbgrid1.OnEditButtonClick := DBGrid1EditButtonClick;
  end
  else
  begin
    DbGrid1.PopupMenu := Nil;
    Dbgrid1.OnEditButtonClick := Nil;
  end;
end;

procedure TfmBayarPiutang.K2Navigator1SaveClick(Sender: TObject);
var
  i: Integer;
  Jumlah, Lebih: Currency;
  selisihPrebayar,TotBayar: Currency;
  Jumlah_Lebih: Currency;
begin
  K2Navigator1.SetFocus;

  with DMHutang do
  begin
    if taPLPTanggal.IsNull                    then DispErr('Tanggal belum diisi',
DBDateEdit1);
      if taPLPTANGGAL.Value <> Date           then DispErr('Tanggal harus =
tanggal sekarang', DBDateEdit1);

    if taPLPKODECust.IsNull                   then DispErr('Kode Customer belum diisi',
rxDBComboEdit2);
       if Not taCust.FindKey([taPLPKODECust.VALUE]) then DispErr(Format('Kode
Customer %s tidak ada/tidak ditemukan', [taPLPKODECust.VALUE]),
rxDBComboEdit2);
```

```
    if taPLPKELEBIHAN.Value < 0                    then Disperr('Nilai kelebihan tidak
boleh bernilai negatif', DBEdit10);

   if taPLP.State = dsInsert then
   begin
     taPLPLookup.Filter:='VOID=false';
     taPLPLookup.Filtered:=true;
     taPLPLookup.Refresh;
     try
       if (DBEdit5.Text<>'') and taPLPLookup.FindKey([DBEdit5.Text]) then
         Disperr('Nomer giro sudah pernah dipakai', DBEdit5);
     finally
       taPLPLookup.Filtered:=false;
     end;
   end else
   begin
     taPLPLookup.Filter:='NOGIRO='"+dbEdit5.Text+'" and void=false';
     taPLPLookup.Filtered:=true;
     taPLPLookup.Refresh;
     try
       I:=0;
       taPLPLookup.First;
       while not taPLPLookup.EOF do
       begin
         Inc(I);
         taPLPLookup.Next;
       end;

       if ((I > 0) and (LastValue <> Dbedit5.Text)) then
         Disperr('Nomer giro sudah pernah dipakai', DBEdit5);

     finally
       taPLPLookup.Filtered:=false;
     end;
   end;

     taPLPDtl.DisableControls;  { supaya scroll record tidak berpengaruh di tampilan
grid }
   taPLPdtl.First;
   While Not taPLPdtl.EOF do
   begin
     if taPLPDtlBAYAR.Value > taPLPDtlSISANOTA.Value then
     begin
       taPLPDtl.EnableControls;
        DispErr(Format('No nota %s kelebihan pembayaran', [taPLPDtlNoNota.Value]),
DBGrid1);
     end;

     if ((taPLPDtlID.Value <> 'JLKR') And (taPLPDtlID.Value <> 'RJKR')) then
     begin
```

```
            taPLPDtl.EnableControls;
                DispErr(Format('Entry ID %s tidak ada diantara ID JLKR dan RJKR',
        [taPLPDtlID.Value]), DBGrid1);
          end;

          if Not taPiut.FindKey([dmHutang.taPLPDtlNONOTA.Value]) then begin
            taPLPDtl.EnableControls;
                    DispErr(Format('Nomer nota %s tidak ditemukan di piutang',
        [taPLPDtlNONOTA.Value]), DBGrid1)
          end else
          begin
            if (taPiutID.Value <> taPLPDtlID.Value) then begin
              taPLPDtl.EnableControls;
                Disperr(Format('Nomer nota %s dengan ID=%s tidak ditemukan di piutang',
        [taPLPDtlNONOTA.Value, taPLPDtlID.Value]), DBGrid1);
            end;

            if (taPiutKodeCust.Value <> taPLPKodeCust.Value) then
                DispErr(format('Nomer Nota %s milik customer %s, bukan milik customer
        %s',[taPLPDTLNonota.Value,taPiutKodeCust.Value,taPLPKodeCust.value]),dbgrid1
        );
          end;

          { check nota-2 yang akan di bayarkan, apakah memenuhi kriteria untuk di bayar }
                taPiut.Filter := '(ID='" + taPLPDtlID.Value + "') and (NOMER='" +
        taPLPDtlNONOTA.Value + "')';
          taPiut.Filtered := True;
          taPiut.First;
          taPiut.FindKey([taPLPDtlNoNota.Value]);

          if taPiutID.Value = 'RJKR' then
          begin
            selisihprebayar:=(taPiutKREDIT.Value - taPLPDtlBAYAR.Value);
            if (selisihprebayar < 0) then
            begin
              taPLPDtl.EnableControls;
Disperr(Format('Nota %s kelebihan pembayaran',
        [taPLPDtlNoNota.Value]), DBGrid1);
            end;
          end;

          if taPiutID.Value = 'JLKR' then
          begin
            selisihprebayar:=(taPiutDEBET.Value - taPLPDtlBAYAR.Value);
            if (selisihprebayar < 0) then
            begin
              taPLPDtl.EnableControls;
Disperr(Format('Nota %s kelebihan pembayaran',
        [taPLPDtlNoNota.Value]), DBGrid1);
            end;
```

```
    end;

    taPiut.Filtered := False;

    taPLPDtl.Next;
  end;

  TotBayar := 0;
  taPLPdtl.First;
  While Not taPLPdtl.EOF do
  begin
    TotBayar := TotBayar + taPLPDtlSUBTOTAL.Value;
    taPLPDtl.Next;
  end;

  Jumlah := taPLPJumlah.Value;
  Lebih := taPLPKelebihan.Value;
  Jumlah_Lebih := (Jumlah - Lebih);

  if (Jumlah_Lebih <> TotBayar) then begin
    taPLPDtl.EnableControls;
        Disperr(Format('Pembayaran piutang tidak balance, total pembayaran %m',
[TotBayar]), DBEdit9);
  end;

  { simpan pembayaran untuk tiap-tiap nota ke TPIUT.DB, di field PREBAYAR }
  taPLPDtl.First;
  tot_bayar_piut:=0;
  While Not taPLPDtl.EOF do
  begin
    if taPiut.FindKey([taPLPdtlNONOTA.Value]) then
    begin
      taPiut.Edit;
      taPiutBAYAR.Value := taPiutBAYAR.Value + taPLPdtlBAYAR.Value;
      taPiut.Post;
    end;
    taPLPdtl.Next;
  end;

  if taPLPKELEBIHAN.Value <> 0 then
  begin
    taPiut.Append;
    taPiutNOMER.Value      := 'KL'+ Copy(taPLPNOMER.Value,3,10);
    taPiutTANGGAL.Value    := taPLPTANGGAL.Value;
    taPiutKODECUST.Value   := taPLPKODECUST.Value;
    taPiutNAMACUST.Value   := taPLPNAMACUST.Value;
    taPiutID.Value         := 'RJKR';
    taPiutNOREFF.Value     := taPLPNOMER.Value;
    taPiutIDREFF.Value     := 'PLP';
    taPiutKETERANGAN.Value := 'Kelebihan Pembayaran';
```

```
    taPiutTERM.Value       := 0;
    taPiutDEBET.Value      := 0;
    taPiutKREDIT.Value     := taPLPKELEBIHAN.Value;
    taPiutBAYAR.Value      := 0;
    taPiut.Post;
   end;

   DBInitValue2(taPLP);  { kosongi semua field di taPLP }
   taPLP.Post;
   taPLPDtl.CommitUpdates;
   taPLPDtl.EnableControls;
   HitungSubTotKurs;
  end;
  DBEdit1.SetFocus;
end;

procedure TfmBayarPiutang.DBGrid1EditButtonClick(Sender: TObject);
var nvar, Bayar: Currency;
begin
  nvar  := 0;
  Bayar := 0;

  if dmhutang.taPLPDtl.State = dsBrowse then
  begin
    if dmhutang.taplpdtl.RecordCount > 0 then
      dmhutang.taplpdtl.edit
    else
      dmhutang.taplpdtl.append;
  end;

    fmBrowse.brwTPiutang2(dbGrid1.InplaceEditor.Text,'select * from tpiut where
DEBET+KREDIT>BAYAR and ID="'+dmHutang.taPLPDtlID.Value+'" and
KODECUST="'+dmHutang.taPLPKODECust.Value+'" order by NOMER');
   if (fmBrowse.ModalResult = mrOk) And (fmBrowse.StringGrid1.Cells[1, 0] <> '')
then
  begin
    dmHutang.taPLPDtlNONOTA.Value := fmBrowse.StringGrid1.Cells[1, 0];
      if  DMHutang.taPiut.Locate('NOMER',  dmHutang.taPLPDtlNONOTA.Value,  [])
then
    begin
      if DMHutang.taPLPDtlID.Value = 'RJKR' then
        nvar := DMHutang.taPiut.FieldByName('Kredit').AsCurrency
      else
        nvar := DMHutang.taPiut.FieldByName('Debet').AsCurrency;
      Bayar := DMHutang.taPiut.FieldByName('Bayar').AsCurrency;
    end;
    dmHutang.taPLPDtlTOTALNOTA.Value := nvar;
    dmHutang.taPLPDtlSISANOTA.Value := nvar - Bayar;
    dmHutang.taPLPDtlBAYAR.Value := dmHutang.taPLPDtlSISANOTA.Value;
  end;
```

```
end;

procedure TfmBayarPiutang.dsDetailDataChange(Sender: TObject; Field: TField);
var nvar, bayar: Currency;
begin
  nvar  := 0;
  Bayar := 0;

  dsDetail.OnDataChange := Nil;
  with dmHutang do
  begin
    if taPLPDtl.State In [dsInsert, dsEdit] then
    begin
      if Field = TaPLPDtlNONOTA then
      begin
        if Not taPiut.FindKey([TaPLPDtlNONOTA.Value]) then
          DBGrid1EditButtonClick(Nil)
        else
        begin
          if taPLPDTLID.Value = 'JLKR' then
            nvar := taPIUT.Fieldbyname('DEBET').asCurrency
          else
            nvar := taPIUT.Fieldbyname('KREDIT').asCurrency;
          Bayar := taPIUT.Fieldbyname('BAYAR').asCurrency;

          taPLPDTLTOTALNOTA.Value := nvar;
          taPLPDTLSISANOTA.Value := nvar - Bayar;
          taPLPDTLBAYAR.Value := taPLPDTLSISANOTA.Value;
        end;
      end;

      if Field = taPLPDtlBAYAR then
        if taPLPDTLBayar.Value > taPLPDtlSISANOTA.value then
        begin
          dsDetail.OnDataChange := dsDetailDataChange;
DispErr(Format('No nota %s kelebihan pembayaran',
[taPLPDtlNoNota.Value]), DBGrid1);
        end;

      if Field = taPLPDtlSISANOTA then
        taPLPDtlBAYAR.Value := taPLPDtlSISANota.Value;

      if Field = taPLPDtlBAYAR then
        GetTotalPLP(taPLPdtl);
    end;
  end;
  dsDetail.OnDataChange := dsDetailDataChange;
end;

procedure TfmBayarPiutang.K2Navigator1EditClick(Sender: TObject);
```

```
begin
  if Not Dmhutang.taPLPVOID.Value then
  begin
    LastValue:=DMHutang.taPLPNOGIRO.Value;
    DMHutang.TaPLP.Edit;
    DMHutang.taPLPdtl.Edit;
    dmHutang.taPLPNAMAUSER.Value := UserLogin;

    with DMHutang do
    begin
      taPLPDtl.First;
      tot_bayar_piut:=0;
      While Not taPLPDtl.Eof do
      begin
        if taPiut.FindKey([taPLPDtlNoNota.Value]) then
        begin
          taPiut.Edit;
          taPiutBAYAR.Value := taPiutBAYAR.Value - taPLPDtlBAYAR.Value;
          tot_bayar_piut:=tot_bayar_piut+taPLPdtlBAYAR.Value;
          taPiut.Post;
        end;
        taPLPDtl.Next;
      end;
    end;
  end
  else
    ShowMessage('Tidak dapat diedit, karena sudah void atau posted');
end;

procedure TfmBayarPiutang.K2Navigator1CancelClick(Sender: TObject);
begin
  Case dsNavigator.State Of
    dsInsert:
      begin
        DmHutang.taPLPVoid.Value := True;
        DmHutang.taPLP.Post;
        DmHutang.taPLPdtl.CommitUpdates;
      end;
    dsEdit:
    begin
       if (MessageDlg('Data edit mau di void ?', mtConfirmation, [mbYes, mbNo], 0) =
mrYes) then
      begin
        DmHutang.taPLPdtl.cancelUpdates;
        DmHutang.taPLPVoid.Value := True;
        DmHutang.taPLP.Post;
      end else
      begin
        DmHutang.taPLPdtl.CancelUpdates;
        DmHutang.taPLP.Cancel;
```

```
    end;

      with DMHutang do begin
       taPLPDtl.First;
       tot_bayar_piut:=0;
       While Not taPLPDtl.Eof do
       begin
                taPiut.Filter := '(ID=''' + taPLPDtlID.Value + ''') and (NOMER=''' +
taPLPDtlNONOTA.Value + ''')';
         taPiut.Filtered := True;
         taPiut.First;
         While Not taPiut.Eof do begin
          taPiut.Edit;
          taPiutBAYAR.Value := taPiutBAYAR.Value + taPLPDtlBAYAR.Value;
          tot_bayar_piut:=tot_bayar_piut+taPLPdtlBAYAR.Value;
          taPiut.Post;
          taPiut.Next;
         end;

         taPiut.Filtered := False;
         taPLPDtl.Next;
        end;
      end;
    end; // end case EDIT
  end;
  HitungSubTotKurs;
end;

procedure TfmBayarPiutang.Tambah1Click(Sender: TObject);
begin
  dmHutang.taPLPdtl.Append;
end;

procedure TfmBayarPiutang.Hapus1Click(Sender: TObject);
begin
   if (dmHutang.taPLPDtl.RecordCount > 0) And (dsNavigator.State In [dsInsert,
dsEdit]) then
   if MessageDlg('Hapus record ?', mtConfirmation, [mbYes, mbNo], 0) = mrYes then
     dmHutang.taPLPdtl.Delete;
end;

procedure TfmBayarPiutang.Batal1Click(Sender: TObject);
begin
  dmHutang.taPLPdtl.Cancel;
end;

procedure TfmBayarPiutang.FormCloseQuery(Sender: TObject;
  var CanClose: Boolean);
begin
  if (dsNavigator.state In [dsedit, dsInsert]) then
```

```
    CanClose := False
  else
    CanClose := True;
end;

procedure TfmBayarPiutang.MenuItem1Click(Sender: TObject);
begin
    if (dmHutang.taPLPDtl.RecordCount > 0) And (dsNavigator.State In [dsInsert,
dsEdit]) then
    dmHutang.taPLPdtl.Edit;
end;

procedure TfmBayarPiutang.Simpan1Click(Sender: TObject);
begin
  if (dsDetail.State In [dsInsert, dsEdit]) then
    dmHutang.taPLPdtl.Post;
end;

procedure TfmBayarPiutang.K2Navigator1PreviewClick(Sender: TObject);
begin
  with dmHutang do
  begin
    Preview('ntByrPiut.rpt', taPLPNOMER.Value, Caption);
  end;
end;

procedure TfmBayarPiutang.dsDetailUpdateData(Sender: TObject);
begin
  if dmHutang.taPLPDtl.State = dsInsert then
    dmHutang.GetTotalPLPUpdate(dmHutang.taPLPDtl);
end;

procedure TfmBayarPiutang.DBGrid1Exit(Sender: TObject);
begin
  if dsDetail.State In [dsInsert, dsEdit] then
  begin
    dmHutang.taPLPDtl.Post;
    dmHutang.GetTotalPLP(dmHutang.taPLPDTL);
  end;
end;

procedure TfmBayarPiutang.DBGrid1ShowEditor(Sender: TObject; Field: TField;
  var AllowEdit: Boolean);
begin
  if DBGrid1.SelectedField = dmHutang.taPLPDtlTotalNota then Abort;
end;

procedure TfmBayarPiutang.K2Navigator1SearchClick(Sender: TObject);
begin
  fmSearch.LoadSearch(dmHutang.taPLP);
```

```
end;

procedure TfmBayarPiutang.DBEdit2Exit(Sender: TObject);
begin
  if dsNavigator.State In [dsInsert, dsEdit] then
    UpdatePLPDetil;
end;

procedure TfmBayarPiutang.FormKeydown(Sender: TObject; var Key: Word;
  Shift: TShiftState);
begin
  if Key=VK_F1 then
    Application.HelpJump('PIUTANG');
end;

procedure TfmBayarPiutang.HitungSubTotKurs;
var subtotkurs : Currency;
begin
  subtotkurs:=0;
  with dmHutang do
   begin
     if taPLP.Active then
       subtotkurs:=(taPLPJumlah.Value - taPLPKelebihan.Value);
     stTotBayar.Caption:=Format('%m',[subtotkurs]);
   end;
end;

procedure TfmBayarPiutang.dsNavigatorUpdateData(Sender: TObject);
begin
  if dsNavigator.State in [dsEdit,dsInsert] then
    HitungSubtotkurs;
end;

procedure TfmBayarPiutang.DBGrid1DrawColumnCell(Sender: TObject;
  const Rect: TRect; DataCol: Integer; Column: TColumn;
  State: TGridDrawState);
begin
  if not ((DBGrid1.DataSource.DataSet.FieldByName('ID').AsString='CSDB') or
         (DBGrid1.DataSource.DataSet.FieldByName('ID').AsString='JLKR'))
  then
    DBGrid1.Canvas.Font.Color := clRed
  else
    DBGrid1.Canvas.Font.Style := [];

  DBgrid1.DefaultDrawColumnCell(Rect, Datacol, Column, State);

end;

end.
```

```
unit UBeli;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  Menus, Grids, DBGrids, ComCtrls, ZKNavigator, ExtCtrls, StdCtrls,
  DBCtrls, Mask, Db, DBTables, MRUList, ToolEdit, RXDBCtrl, RxLookup,
  CurrEdit, Buttons, uCrpe32;

type
  TfmBeli = class(TForm)
    dsWatch: TDataSource;
    Panel4: TPanel;
    Label3: TLabel;
    Label4: TLabel;
    Label5: TLabel;
    Label6: TLabel;
    Label8: TLabel;
    deKodeSupl: TRxDBComboEdit;
    Panel6: TPanel;
    deDisc: TDBEdit;
    dePpn: TDBEdit;
    Label10: TLabel;
    Label11: TLabel;
    Label12: TLabel;
    Label13: TLabel;
    Panel3: TPanel;
    DBEdit2: TDBEdit;
    DBEdit3: TDBEdit;
    DBEdit4: TDBEdit;
    DBEdit5: TDBEdit;
    Panel2: TPanel;
    Label14: TLabel;
    deTerm: TDBEdit;
    ceDisc: TCurrencyEdit;
    cePpn: TCurrencyEdit;
    Panel5: TPanel;
    Panel9: TPanel;
    dsDetail: TDataSource;
    Label30: TLabel;
    DBMemo1: TDBMemo;
    DBGrid1: TRxDBGrid;
    KNav: TK2Navigator;
    DBCheckBox1: TDBCheckBox;
    Bevel3: TBevel;
    CheckBox1: TCheckBox;
    PopupMenu1: TPopupMenu;
    Add1: TMenuItem;
    Edit1: TMenuItem;
```

```
    Delete1: TMenuItem;
    Simpan1: TMenuItem;
    Batal1: TMenuItem;
    Label7: TLabel;
    Panel7: TPanel;
    Label22: TLabel;
    Label23: TLabel;
    Panel8: TPanel;
    DBText1: TDBText;
    deTanggal: TDBDateEdit;
    rgJenisBayar: TDBRadioGroup;
    procedure FormClose(Sender: TObject; var Action: TCloseAction);
    procedure FormShow(Sender: TObject);
    procedure dsWatchStateChange(Sender: TObject);
    procedure KNavCancelClick(Sender: TObject);
    procedure KNavAddClick(Sender: TObject);
    procedure deKodeSuplButtonClick(Sender: TObject);
    procedure KNavSaveClick(Sender: TObject);
    procedure cePpnExit(Sender: TObject);
    procedure ceDiscExit(Sender: TObject);
    procedure dsWatchDataChange(Sender: TObject; Field: TField);
    procedure DBGrid1EditButtonClick(Sender: TObject);
    procedure dsDetailDataChange(Sender: TObject; Field: TField);
    procedure rgJenisBayarChange(Sender: TObject);
    procedure Add1Click(Sender: TObject);
    procedure Edit1Click(Sender: TObject);
    procedure Delete1Click(Sender: TObject);
    procedure Simpan1Click(Sender: TObject);
    procedure Batal1Click(Sender: TObject);
    procedure PopupMenu1Popup(Sender: TObject);
    procedure KNavPreviewClick(Sender: TObject);
    procedure KNavSearchClick(Sender: TObject);
    procedure FormCloseQuery(Sender: TObject; var CanClose: Boolean);
    procedure DBGrid1ColEnter(Sender: TObject);
    procedure DBGrid1KeyPress(Sender: TObject; var Key: Char);
    procedure CheckBox1Enter(Sender: TObject);
  private
    taBrg, taSupl, taGdgBrg : TTable;
    subtotal, grandtotal    : real;
    oldkodebrg              :  string;

    procedure SettingEdit(nVal : boolean);
    procedure Cancel_All_Updates;

    procedure SetEvent;
    procedure ClearEvent;

    procedure taBeliKODESUPLChange(Sender: TField);
    procedure taBeliPPNChange(Sender: TField);
    procedure taBeliDISCChange(Sender: TField);
```

```
    procedure taBelidtlKODEBRGChange(Sender: TField);

    procedure taBeliDtlCalcFields(DataSet: TDataSet);
    procedure taBeliDtlNewRecord(DataSet: TDataSet);
    procedure taBelidtlBeforeDelete(DataSet: TDataSet);
    procedure AfterScroll(DataSet : TDataSet);

    function HitungTotNota:Currency;

  public
  end;

var
  fmBeli: TfmBeli;

implementation

uses AllUnits, UDmPembelian, UBrowse, USearch;

{$R *.DFM}

procedure TfmBeli.AfterScroll(DataSet : TDataSet);
begin
  with dmPembelian Do
  begin
    if taBeliTOTNOTA.Value = 0 Then
      ceDisc.Value := 0
    else ceDisc.Value := taBeliDISC.Value * 100 / taBeliTOTNOTA.Value;

    if taBeliTOTNOTA.Value = 0 Then
      cePpn.Value  := 0
    else cePpn.Value  := taBeliPPN.Value * 100 / taBeliTOTNOTA.Value;

    subtotal       := DBGetTotal(taBelidtlTOTAL);
    Panel5.Caption := format('%m ', [subtotal]);
    grandtotal     := subtotal - taBeliDISC.Value + taBeliPPN.Value;
    Panel9.Caption := format('%m ', [grandtotal]);
  end;
End;

procedure TfmBeli.SetEvent;
begin
  with dmPembelian do
  begin
    taBeliKODESUPL.OnChange   := taBeliKODESUPLChange;
    taBeliPPN.OnChange        := taBeliPPNChange;
    taBeliDISC.OnChange       := taBeliDISCChange;
    taBelidtlKODEBRG.OnChange := taBelidtlKODEBRGChange;
    taBelidtl.OnNewRecord     := taBeliDtlNewRecord;
    taBelidtl.BeforeDelete    := taBelidtlBeforeDelete;
```

```
    taBeliDtl.OnCalcFields    := taBeliDtlCalcFields;
  end;
end;

procedure TfmBeli.ClearEvent;
begin
  with dmPembelian do
  begin
    taBeliKODESUPL.OnChange   := nil;
    taBeliPPN.OnChange        := nil;
    taBeliDISC.OnChange       := nil;
    taBelidtlKODEBRG.OnChange := nil;
    taBelidtl.BeforeInsert    := nil;
    taBelidtl.OnNewRecord     := nil;
    taBelidtl.BeforeDelete    := nil;
    taBeliDtl.OnCalcFields    := nil;
  end;
end;

procedure TfmBeli.Cancel_All_Updates;
begin
  with dmPembelian do
  begin
    taBeliDtl.CancelUpdates;
    taBeli.Cancel;
  end;
end;

procedure TfmBeli.SettingEdit(nVal : boolean);
var n : integer;
begin
  for n := 0 to fmBeli.ComponentCount - 1 do
if fmBeli.Components[n].Tag=100 then
TControl(fmBeli.Components[n]).Enabled := nVal;
end;

procedure TfmBeli.taBeliKODESUPLChange(Sender: TField);
begin
  with dmPembelian do
  begin
    taBeliKODESUPL.OnChange := nil;

    if not taSupl.findkey([Sender.asString]) then deKodeSuplButtonClick(nil);

    taBeliKODESUPL.Value    := taSupl.FieldByName('KODE').asString;
    taBeliNAMASUPL.Value    := taSupl.FieldByName('NAMA').asString;
    taBeliALAMAT.Value      := taSupl.FieldByName('ALAMAT').asString;
    taBeliKOTA.Value        := taSupl.FieldByName('KOTA').asString;
    taBeliNEGARA.Value      := taSupl.FieldByName('NEGARA').asString;
```

```
    { jika Nama, Alamat, Kota, Negara ada isinya maka cuma readonly }
    DBEdit2.ReadOnly := not taSupl.FieldByName('NAMA').isNull;
    DBEdit3.ReadOnly := not taSupl.FieldByName('ALAMAT').isNull;
    DBEdit4.ReadOnly := not taSupl.FieldByName('KOTA').isNull;
    DBEdit5.ReadOnly := not taSupl.FieldByName('NEGARA').isNull;

    taBeliKODESUPL.OnChange := taBeliKODESUPLChange;
  end;
end;

procedure TfmBeli.taBeliPPNChange(Sender: TField);
begin
  if subtotal > 0 then cePpn.Value := sender.asCurrency * 100 / subtotal;
end;

procedure TfmBeli.taBeliDISCChange(Sender: TField);
begin
  if subtotal > 0 then
    ceDisc.Value := sender.asCurrency * 100 / subtotal;
end;

procedure TfmBeli.taBelidtlKODEBRGChange(Sender: TField);
begin
  with dmPembelian do
  begin
    dsDetail.OnDataChange     := nil;

    if not taBrg.FindKey([sender.asString]) then DBGrid1EditButtonClick(nil);

    taBelidtlKODEBRG.OnChange := nil;

    taBelidtl.Edit;
    taBelidtlKODEBRG.Value    := taBrg.FieldByName('KODE').asString;
    taBelidtlNAMABRG.Value    := taBrg.FieldByName('NAMA').asString;
    taBelidtlSATUAN.Value     := taBrg.FieldByName('SATUAN').asString;
    taBelidtlJUMLAH.Value     := 1;
    taBelidtlHARGA.Value      := 0;
    taBelidtlDISC.Value       := 0;

    taBelidtl.Post;
    taBelidtlKODEBRG.OnChange := taBelidtlKODEBRGChange;
    dsDetail.OnDataChange     := dsDetailDataChange;
  end;
end;

procedure TfmBeli.taBeliDtlNewRecord;
begin
  DBInitValue(dmPembelian.taBeliDtl);
  DBgrid1.SelectedIndex := 0;
end;
```

```
procedure TfmBeli.taBeliDtlCalcFields;
begin
  with dmPembelian do
     taBelidtlTOTAL.Value := (taBeliDtlJUMLAH.Value * taBeliDtlHARGA.Value) -
taBeliDtlDISC.Value;
end;

procedure TfmBeli.taBeliDtlBeforeDelete(DataSet: TDataSet);
begin
  if dsWatch.State <> dsBrowse then
     if MessageDlg('Hapus record ... ?',mtConfirmation,[mbYes, mbNo], 0) <> mrYes
then abort;
end;

procedure TfmBeli.FormShow;
begin
  with dmPembelian do
  begin
    DBSetPath(dmPembelian);
    DBLoadLabel(taBelidtl);

    taGdgBrg   := OpenTable('TGDGBRG' , '', fmBeli);
    taSupl     := OpenTable('TSUPL'   , '', fmBeli);
    taBrg      := OpenTable('TBARANG' , '', fmBeli);

    taBelidtl.Open;
    taBeli.Open;

    SetEvent;
    taBeli.AfterScroll := AfterScroll;
    AfterScroll(taBeli);
  end;
  deTanggal.SetFocus;

end;

procedure TfmBeli.FormClose;
begin
  with dmPembelian do
  begin
    taSupl.Close;
    taBrg.Close;
    taGdgBrg.Close;
    taBelidtl.Close;
    taBeli.Close;
    taBeli.AfterScroll := NIL;
  end;
end;
```

```
procedure TfmBeli.KNavCancelClick;
begin
  KNav.SetFocus;
  with dmPembelian do begin
    case dsWatch.State of
      dsInsert : begin
                   taBeliNAMAUSER.Value := UserLogin;
                   taBeliVOID.Value     := true;
                   taBeli.Post;
                   taBelidtl.CommitUpdates;
                 end;
    end;
    Cancel_All_Updates;
    deTanggal.SetFocus;
  end;
end;


procedure TfmBeli.KNavAddClick;
begin
  ClearEvent;
  with dmPembelian do begin
    taBeli.Append;
    DBInitValue(taBeli);
    taBeliNOMER.Value    := DBGetCounter('BL', fmBeli);
    taBeliJENIS.Value    := 0;
    taBeliTERM.Value     := 0;
    taBelidtl.EnableControls;
  end;

  subtotal                := 0;
  deKodeSupl.Button.Enabled := true;
  deTanggal.SetFocus;
  SetEvent;
end;

procedure TfmBeli.KNavSaveClick;
begin
  with dmPembelian do begin

    KNav.SetFocus;
    dsDetail.OnDataChange := nil;  { supaya jangan rekursif }

    taBeli.DisableControls;         { supaya semua proses di dlm save tidak berpengaruh
pada control }
    taBelidtl.DisableControls;
    try
       if taBeliTANGGAL.IsNull                    then DispErr('Tanggal tidak boleh
kosong', deTanggal);
```

```
    if taBeliTANGGAL.Value <> Date                then DispErr('Tanggal harus =
tanggal sekarang', deTanggal);

    if taBeliKODESUPL.IsNull                      then DispErr('Kode Supplier tidak
boleh kosong', deKodeSupl);
    if not taSupl.FindKey([taBeliKODESUPL.Value])  then DispErr('Kode Supplier
tidak ditemukan', deKodeSupl);

    if ((taBeliTERM.Value < 0) and (taBeliJENIS.Value = 1)) then disperr('Term
harus > 0',deTerm);

  { *** check detail *** }
  taBelidtl.First;
  while not taBelidtl.EOF do
  begin
    if taBelidtlKODEBRG.IsNull                    then DispErr('Kode Barang tidak
boleh kosong', DBGrid1);
    if not taBrg.FindKey([taBelidtlKODEBRG.Value]) then DispErr('Kode Barang
tidak ditemukan', DBGrid1);

    if taBelidtlJUMLAH.IsNull                     then DispErr('Jumlah tidak boleh
kosong', DBGrid1);
    if taBelidtlJUMLAH.Value < 0                  then DispErr('Jumlah tidak boleh < 0',
DBGrid1);

    if taBelidtlSATUAN.IsNull                     then DispErr('Satuan tidak boleh
kosong', DBGrid1);
    if taBeliDtlHARGA.IsNull                      then DispErr('Harga tidak boleh
kosong', DBGrid1);
    if taBeliDtlHARGA.Value < 0                   then DispErr('Harga tidak boleh < 0',
DBGrid1);

    taBelidtl.Next;
  end;

  if not taHut.Active then taHut.Open;

  if not taHut.FindKey([taBeliNOMER.Value]) then
    taHut.Append
  else taHut.Edit;

  taHut.Fieldbyname('NOMER').asString       := taBeliNOMER.Value;
  taHut.Fieldbyname('TANGGAL').asDatetime   := taBeliTANGGAL.Value;
  taHut.Fieldbyname('KODESUPL').asString    := taBeliKODESUPL.Value;
  taHut.Fieldbyname('NAMASUPL').asString    := taBeliNAMASUPL.Value;
  taHut.Fieldbyname('NOREFF').asString      := '';
  taHut.Fieldbyname('IDREFF').asString      := '';
  taHut.Fieldbyname('TERM').asInteger       := taBeliTERM.Value;
  taHut.Fieldbyname('DEBET').asCurrency     := 0;
  taHut.Fieldbyname('KREDIT').asCurrency    := GrandTotal;
```

```
      if taBeliJENIS.Value = 0 then
      begin
        taHut.Fieldbyname('KETERANGAN').asString := 'Pembelian Tunai';
        taHut.Fieldbyname('ID').asString         := 'BLTN';
        taHut.Fieldbyname('BAYAR').asCurrency    := GrandTotal;
      end else
      begin
        taHut.Fieldbyname('KETERANGAN').asString := 'Pembelian Kredit';
        taHut.Fieldbyname('ID').asString         := 'BLKR';
        taHut.Fieldbyname('BAYAR').asCurrency    := 0;
      end;

      taHut.Post;

      { force setting }
      taBeliNAMAUSER.Value := UserLogin;
      taBeli.Post;
      taBelidtl.CommitUpdates;

      taBeli.Edit;
      taBeliTOTNOTA.Value  := HitungTotNota;
      DBInitValue2(taBeli);
      taBeli.Post;

      taBelidtl.First;
      deTanggal.SetFocus;

    finally
      dsDetail.OnDataChange := dsDetailDataChange; { supaya jangan rekursif }
      taBeli.EnableControls;
      taBelidtl.EnableControls;
    end;
  end;
end;

procedure TfmBeli.DBGrid1EditButtonClick(Sender: TObject);
begin
  with dmPembelian do
  begin
    oldkodebrg:=taBeliDtlKODEBRG.Value ;

    fmBrowse.brwTBARANG(dbgrid1.InplaceEditor.Text, nil);
    taBelidtl.Edit;
    taBelidtlKODEBRG.Value := fmBrowse.StringGrid1.Cells[1, 0];
  end;
end;

procedure TfmBeli.deKodeSuplButtonClick;
begin
```

```
  fmBrowse.brwTSUPL(deKodeSupl.Text, nil);
  taSupl.FindKey([fmBrowse.StringGrid1.Cells[1, 0]]);
  dmPembelian.taBeliKODESUPL.Value := fmBrowse.StringGrid1.Cells[1, 0];
end;

procedure TfmBeli.cePpnExit(Sender: TObject);
begin
  if dsWatch.State in [dsEdit, dsInsert] then
    dmPembelian.taBeliPPN.Value := (subtotal * cePpn.Value) / 100;
end;

procedure TfmBeli.ceDiscExit(Sender: TObject);
begin
  if dsWatch.State in [dsEdit, dsInsert] then
    dmPembelian.taBeliDISC.Value := (subtotal * ceDisc.Value) / 100;
end;

procedure TfmBeli.dsWatchStateChange;
begin
  if dsWatch.State = dsBrowse then
    SettingEdit(True);

  deKodeSupl.Button.Enabled := dsWatch.State in [dsEdit, dsInsert];

  ceDisc.ReadOnly          := dsWatch.State = dsBrowse;
  cePpn.ReadOnly           := dsWatch.State = dsBrowse;

  if dsWatch.State = dsBrowse then
    DBGrid1.Options := DBGrid1.Options - [dgEditing]
  else
    DBGrid1.Options := DBGrid1.Options + [dgEditing]
end;

procedure TfmBeli.dsWatchDataChange(Sender: TObject; Field: TField);
begin
  dsWatch.OnDataChange := nil;

  try
    with dmPembelian do
    begin
      deDisc.ReadOnly   := subtotal = 0;
      dePpn.ReadOnly    := subtotal = 0;
      CheckBox1.Checked := taBeliPRINTED.Value > 0;

      if (subtotal > 0) and (field = nil) and (dsWatch.State <> dsBrowse) then begin
        taBeliDISC.Value := subtotal * ceDISC.Value / 100;
        taBeliPPN.Value  := subtotal * cePPN.Value / 100;
      end;

      grandtotal     := subtotal - taBeliDISC.Value + taBeliPPN.Value;
```

```
    Panel9.Caption := format('%m ', [grandtotal]);
   end;
  except
  end;

  dsWatch.OnDataChange := dsWatchDataChange;
end;

procedure TfmBeli.dsDetailDataChange(Sender: TObject; Field: TField);
var nSatuan : word;
begin
  dsDetail.OnDataChange := nil;  { supaya jangan rekursif }

  try
   with dmPembelian do
   begin
    { cari posisi kolom untuk field satuan, ini penting
      soalnya kolom dapat dipindah-pindah }
      for nSatuan := 0 to DBgrid1.FieldCount - 1 do if DBgrid1.Fields[nSatuan] =
taBelidtlSATUAN then break;
    DBGrid1.Columns[nSatuan].PickList.Clear;

DBGrid1.Columns[nSatuan].PickList.Add(taBrg.FieldByName('SATUAN').AsString
);

    { menghitung total dari detail }
    subtotal      := DBGetTotal(taBelidtlTOTAL);
    Panel5.Caption := format('%m ', [subtotal]);
    dsWatchDataChange(nil, nil);  { supaya respon ke Discount, PPN dan Grand Total
}
   end;
  except
  end;

  dsDetail.OnDataChange := dsDetailDataChange;
end;

procedure TfmBeli.rgJenisBayarChange(Sender: TObject);
begin
  deTerm.Enabled  := rgJenisBayar.ItemIndex = 1;
  Label14.Enabled := rgJenisBayar.ItemIndex = 1;
  Label7.Enabled  := rgJenisBayar.ItemIndex = 1;

  if dsWatch.State in [dsEdit, dsInsert] then
   if rgJenisBayar.ItemIndex = 0 then dmPembelian.taBeliTERM.Value := 0;
end;

procedure TfmBeli.Add1Click(Sender: TObject);
begin
  dmPembelian.taBelidtl.Append;
```

```
end;

procedure TfmBeli.Edit1Click(Sender: TObject);
begin
  dmPembelian.taBelidtl.Edit;
end;

procedure TfmBeli.Delete1Click(Sender: TObject);
begin
  if not dmPembelian.taBelidtl.EOF then dmPembelian.taBelidtl.Delete;
end;

procedure TfmBeli.Simpan1Click(Sender: TObject);
begin
  if dsDetail.State in [dsEdit, dsInsert] then dmPembelian.taBelidtl.Post;
end;

procedure TfmBeli.Batal1Click(Sender: TObject);
begin
  dmPembelian.taBelidtl.Cancel;
end;

procedure TfmBeli.PopupMenu1Popup(Sender: TObject);
begin
  if dsWatch.State = dsBrowse then abort;
end;

procedure TfmBeli.KNavPreviewClick(Sender: TObject);
begin
  with dmPembelian do begin
    crpe1.Clear;
    crpe1.ReportName              := PathExe + 'Report\ntbeli.rpt';
    crpe1.Output                  := toWindow;
    crpe1.DiscardSavedData        := true;
    crpe1.WindowZoom.Magnification := 100;
    crpe1.WindowStyle.Title       := fmBeli.Caption;

    crpe1.Tables.Retrieve;
    crpe1.Tables[0].Path   := PathData;
    crpe1.Tables.Propagate := True;

    crpe1.ParamFields.Retrieve;
    crpe1.ParamFields[0].Value := Userlogin;
    crpe1.ParamFields[1].Value := taBeliNOMER.Text;
    crpe1.WindowButtonBar.PrintBtn := False;
    crpe1.Execute;

    crpe1.Clear;
    crpe1.ReportName              := PathExe + 'Report\ntbeligdg.rpt';
    crpe1.Output                  := toWindow;
```

```
    crpe1.DiscardSavedData        := true;
    crpe1.WindowZoom.Magnification := 100;
    crpe1.WindowStyle.Title       := fmBeli.Caption + ' (Gudang)';

    crpe1.Tables.Retrieve;
    crpe1.Tables[0].Path   := PathData;
    crpe1.Tables.Propagate := True;

    crpe1.ParamFields.Retrieve;
    crpe1.ParamFields[0].Value := Userlogin;
    crpe1.ParamFields[1].Value := taBeliNOMER.Text;
    crpe1.WindowButtonBar.PrintBtn := False;
    crpe1.Execute;
  end;
end;

procedure TfmBeli.KNavSearchClick(Sender: TObject);
begin
  fmSearch.LoadSearch(dmPembelian.taBeli);
end;

procedure TfmBeli.FormCloseQuery(Sender: TObject; var CanClose: Boolean);
begin
  CanClose := dsWatch.State=dsBrowse;
end;

Function TfmBeli.HitungTotNota:Currency;
Var GrandTotNota : Currency;
Begin
  With dmPembelian do
  begin
    taBeliDtl.DisableControls;
    taBeliDtl.First;
    GrandTotNota := 0;
    While Not taBeliDtl.EOF do
    Begin
      // Sum GrandTotal
                GrandTotNota  :=  GrandTotNota  +  (taBeliDtlJUMLAH.Value  *
taBeliDtlHARGA.Value) - taBeliDtlDISC.Value;

      // Update GdgBrg
      if not taGdgBrg.FindKey([taBelidtlKODEBRG.Value]) then
      begin
        taGdgBrg.Append;
        taGdgBrg.FieldByName('KODEBRG').AsString := taBelidtlKODEBRG.Value;
        taGdgBrg.FieldByName('SALDO').AsFloat    := taBelidtlJUMLAH.Value;
      end else
      begin
        taGdgBrg.Edit;
```

```
taGdgBrg.FieldByName('SALDO').AsFloat
        taGdgBrg.FieldByName('SALDO').AsFloat +
                                        taBelidtlJUMLAH.Value;
          end;

          taGdgBrg.Post;
          taBeliDtl.Next;
        End;
        taBeliDtl.EnableControls;
      End;
      Result := GrandTotNota;
    End;

    procedure TfmBeli.DBGrid1ColEnter(Sender: TObject);
    begin
      oldKodeBrg:=dmPembelian.taBeliDtlKodebrg.value;
    end;

    procedure TfmBeli.DBGrid1KeyPress(Sender: TObject; var Key: Char);
    begin
      if DBGrid1.SelectedField.FieldName='KODEBRG' then
        Key:=UpCase(Key);
    end;

    procedure TfmBeli.CheckBox1Enter(Sender: TObject);
    begin
      PostMessage(Handle, WM_NEXTDLGCTL, 0, 0)
    end;

    end.
```

unit uBrowse;

```
interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  Db, DBTables, StdCtrls, ExtCtrls, Grids, DBGrids, Buttons, ComCtrls,
  DBCtrls;

type
  TfmBrowse = class(TForm)
    DBGrid1: TDBGrid;
    Panel2: TPanel;
    Label1: TLabel;
    Edit1: TEdit;
    dsData: TDataSource;
    taData: TTable;
    SpeedButton1: TSpeedButton;
```

```
    Splitter1: TSplitter;
    StringGrid1: TStringGrid;
    Label2: TLabel;
    dsqrData: TDataSource;
    qrData: TQuery;
    procedure FormShow(Sender: TObject);
    procedure FormKeyPress(Sender: TObject; var Key: Char);
    procedure Edit1Change(Sender: TObject);
    procedure DBGrid1DblClick(Sender: TObject);
    procedure DBGrid1ColEnter(Sender: TObject);
    procedure SpeedButton1Click(Sender: TObject);
    procedure Splitter1Moved(Sender: TObject);
       procedure Edit1MouseDown(Sender: TObject; Button: TMouseButton; Shift:
TShiftState; X, Y: Integer);
    procedure FormClose(Sender: TObject; var Action: TCloseAction);
     procedure StringGrid1DrawCell(Sender: TObject; Col, Row: Integer; Rect: TRect;
State: TGridDrawState);
    procedure dsDataDataChange(Sender: TObject; Field: TField);
    procedure Edit1KeyDown(Sender: TObject; var Key: Word;
      Shift: TShiftState);
  private
    { Private declarations }

    Index   : string;
    Nomer   : string;

    procedure InitBrowse(fTable, cIdx, cPass : string; fproc : TFilterRecordEvent);
    procedure InitBrowse2(SQL,cIdx, cPass: string);
  public
    { Public declarations }
    nPosRec : integer;

    function  GetValue(cfield : string) : variant;
    procedure brwTUSER(_No : string; fproc : TFilterRecordEvent);
    procedure brwTBELI(_No : string; fproc : TFilterRecordEvent);
    procedure brwTJUAL(_No : string; fproc : TFilterRecordEvent);
    procedure brwTCUST(_No : string; fproc : TFilterRecordEvent);
    procedure brwTSALES(_No : string; fproc : TFilterRecordEvent);
    procedure brwTSUPL(_No : string; fproc : TFilterRecordEvent);
    procedure brwTPLH(_No : string; fproc : TFilterRecordEvent);
    procedure brwTPLHDTL(_No : string; fproc : TFilterRecordEvent);
    procedure brwTPLP(_No : string; fproc : TFilterRecordEvent);
    procedure brwTPLPDTL(_No : string; fproc : TFilterRecordEvent);
    procedure brwTGUDANG(_No : string; fproc : TFilterRecordEvent);
    procedure brwTSERVICE(_No : string; fproc : TFilterRecordEvent);
    procedure brwTBARANG(_No : string; fproc : TFilterRecordEvent);
    procedure brwTHutang(_No : string; fproc : TFilterRecordEvent);
    procedure brwTPiutang(_No : string; fproc : TFilterRecordEvent);
    procedure brwTPiutang2(_No, SQL : string);
```

```
  end;

var
  fmBrowse: TfmBrowse;

implementation

uses allunits;

{$R *.DFM}

procedure TfmBrowse.brwTUSER;
begin
  InitBrowse('TUSER', 'USERNAME', _No, fproc);
  AddColumn(Dbgrid1, 'USERNAME');
  AddColumn(Dbgrid1, 'KODEGROUP');
  ShowModal;
End;

procedure TfmBrowse.brwTBELI;
begin
  InitBrowse('TBELI', 'NOMER', _No, fproc);
  AddColumn(Dbgrid1, 'NOMER');
  AddColumn(Dbgrid1, 'TANGGAL');
  AddColumn(Dbgrid1, 'KODESUPL');
  AddColumn(Dbgrid1, 'NAMASUPL');
  AddColumn(Dbgrid1, 'NOMERORDER');
  AddColumn(Dbgrid1, 'KETERANGAN');
  ShowModal;
End;

procedure TfmBrowse.brwTJUAL;
begin
  InitBrowse('TJUAL', 'NOMER', _No, fproc);
  AddColumn(Dbgrid1, 'NOMER');
  AddColumn(Dbgrid1, 'TANGGAL');
  AddColumn(Dbgrid1, 'KODECUST');
  AddColumn(Dbgrid1, 'NAMACUST');
  AddColumn(Dbgrid1, 'NOMERORDER');
  AddColumn(Dbgrid1, 'KETERANGAN');
  ShowModal;
End;

procedure TfmBrowse.brwTPIUTANG;
begin
  InitBrowse('TPiut', 'NOMER', _No, fProc);
  AddColumn(Dbgrid1, 'NOMER');
  AddColumn(Dbgrid1, 'TANGGAL');
  AddColumn(Dbgrid1, 'KODECust');
  AddColumn(Dbgrid1, 'NAMACust');
```

```
  AddColumn(Dbgrid1, 'SATCUR');
  AddColumn(Dbgrid1, 'KURS');
  AddColumn(Dbgrid1, 'PREBAYAR');
  AddColumn(Dbgrid1, 'BAYAR');
  AddColumn(Dbgrid1, 'ID');
  AddColumn(Dbgrid1, 'KETERANGAN');
  AddColumn(Dbgrid1, 'DEBET');
  AddColumn(Dbgrid1, 'KREDIT');
  ShowModal;
end;

procedure TfmBrowse.brwTPIUTANG2;
begin
  InitBrowse2(SQL,'NOMER',_No);
  AddColumn(Dbgrid1, 'NOMER');
  AddColumn(Dbgrid1, 'TANGGAL');
  AddColumn(Dbgrid1, 'KODECust');
  AddColumn(Dbgrid1, 'NAMACust');
  AddColumn(Dbgrid1, 'SATCUR');
  AddColumn(Dbgrid1, 'KURS');
  AddColumn(Dbgrid1, 'PREBAYAR');
  AddColumn(Dbgrid1, 'BAYAR');
  AddColumn(Dbgrid1, 'ID');
  AddColumn(Dbgrid1, 'KETERANGAN');
  AddColumn(Dbgrid1, 'DEBET');
  AddColumn(Dbgrid1, 'KREDIT');
  ShowModal;
end;


procedure TfmBrowse.brwTHUTANG;
begin
  InitBrowse('THut', 'NOMER', _No, fproc);
  AddColumn(Dbgrid1, 'NOMER');
  AddColumn(Dbgrid1, 'TANGGAL');
  AddColumn(Dbgrid1, 'KODESUPL');
  AddColumn(Dbgrid1, 'NAMASUPL');
  AddColumn(Dbgrid1, 'SATCUR');
  AddColumn(Dbgrid1, 'KURS');
  AddColumn(Dbgrid1, 'PREBAYAR');
  AddColumn(Dbgrid1, 'BAYAR');
  AddColumn(Dbgrid1, 'ID');
  AddColumn(Dbgrid1, 'KETERANGAN');
  AddColumn(Dbgrid1, 'DEBET');
  AddColumn(Dbgrid1, 'KREDIT');
  ShowModal;
end;

procedure TfmBrowse.brwTBARANG;
begin
```

```
  InitBrowse('TBARANG', 'KODE', _No, fproc);
  AddColumn(Dbgrid1, 'KODE');
  AddColumn(Dbgrid1, 'NAMA');
  AddColumn(Dbgrid1, 'VENDORCODE');
  AddColumn(Dbgrid1, 'SATUAN1');
  AddColumn(Dbgrid1, 'HRGRATA');
  AddColumn(Dbgrid1, 'DISC');
  AddColumn(Dbgrid1, 'KETERANGAN');
  ShowModal;
end;

procedure TfmBrowse.brwTGUDANG;
begin
  InitBrowse('TGUDANG', 'KODE', _No, fproc);
  AddColumn(Dbgrid1, 'KODE');
  AddColumn(Dbgrid1, 'NAMA');
  ShowModal;
end;

procedure TfmBrowse.brwTSERVICE;
begin
  InitBrowse('TSERVICE', 'KODE', _No, fproc);
  AddColumn(Dbgrid1, 'KODE');
  AddColumn(Dbgrid1, 'NAMA');
  ShowModal;
end;

procedure TfmBrowse.brwTCUST;
begin
  InitBrowse('TCUST', 'KODE', _No, fproc);
  AddColumn(Dbgrid1, 'KODE');
  AddColumn(Dbgrid1, 'NAMA');
  AddColumn(Dbgrid1, 'ALAMAT');
  AddColumn(Dbgrid1, 'KOTA');
  AddColumn(Dbgrid1, 'NEGARA');
  AddColumn(Dbgrid1, 'TERM');
  ShowModal;
end;

procedure TfmBrowse.brwTSUPL;
begin
  InitBrowse('TSUPL', 'KODE', _No, fproc);
  AddColumn(Dbgrid1, 'KODE');
  AddColumn(Dbgrid1, 'NAMA');
  AddColumn(Dbgrid1, 'ALAMAT');
  AddColumn(Dbgrid1, 'KOTA');
  AddColumn(Dbgrid1, 'NEGARA');
  ShowModal;
end;
```

```
procedure TfmBrowse.brwTSALES;
begin
  InitBrowse('TSALES', 'KODE', _No, fproc);
  AddColumn(Dbgrid1, 'KODE');
  AddColumn(Dbgrid1, 'NAMA');
  AddColumn(Dbgrid1, 'ALAMAT');
  AddColumn(Dbgrid1, 'TELEPON');
  AddColumn(Dbgrid1, 'HP');
  AddColumn(Dbgrid1, 'FAX');
  ShowModal;
end;

procedure TfmBrowse.brwTPLH;
begin
  InitBrowse('TPLH', 'NOMER', _No, fproc);
  AddColumn(Dbgrid1, 'NOMER');
  AddColumn(Dbgrid1, 'TANGGAL');
  AddColumn(Dbgrid1, 'KODESUPL');
  AddColumn(Dbgrid1, 'NAMASUPL');
  AddColumn(Dbgrid1, 'JUMLAH');
  ShowModal;
end;

procedure TfmBrowse.brwTPLHDtl;
begin
  InitBrowse('TPLHDTL', 'NOMER', _No, fproc);
  AddColumn(Dbgrid1, 'NOMER');
  AddColumn(Dbgrid1, 'NONOTA');
  AddColumn(Dbgrid1, 'TOTALNOTA');
  AddColumn(Dbgrid1, 'BAYAR');
  ShowModal;
end;

procedure TfmBrowse.brwTPLP;
begin
  InitBrowse('TPLP', 'NOMER', _No, fproc);
  AddColumn(Dbgrid1, 'NOMER');
  AddColumn(Dbgrid1, 'TANGGAL');
  AddColumn(Dbgrid1, 'KODECust');
  AddColumn(Dbgrid1, 'NAMACust');
  AddColumn(Dbgrid1, 'JUMLAH');
  ShowModal;
end;

procedure TfmBrowse.brwTPLPDtl;
begin
  InitBrowse('TPLPDTL', 'NOMER', _No, fproc);
  AddColumn(Dbgrid1, 'NOMER');
  AddColumn(Dbgrid1, 'NONOTA');
  AddColumn(Dbgrid1, 'TOTALNOTA');
```

```
  AddColumn(Dbgrid1, 'BAYAR');
  ShowModal;
end;

{ ********************** main program ************************}
{-------------------------}
function TfmBrowse.getvalue;
{-------------------------}
begin
  result := StringGrid1.Cells[1, StringGrid1.Cols[0].IndexOf(cfield)];
end;

{----------------------------}
procedure TfmBrowse.InitBrowse;
{----------------------------}
begin
  DbGrid1.DataSource:=dsData;
  Dbgrid1.Columns.Clear;
  if assigned(fproc) then begin
    taData.Filtered       := true;
    taData.OnFilterRecord := fproc;
  end
  else
    taData.Filtered := false;

{  if (UpperCase(fTable)='TUSER') or (UpperCase(fTable)='TGROUP') then
  taData.DatabaseName:= AllUnits.PathUser;
}
  taData.TableName      := fTable;
  Nomer                 := cPass;
  Index                 := cIdx;
end;

procedure TfmBrowse.InitBrowse2(SQL,cIdx,cPass : string);
{----------------------------}
begin
  qrData.Close;
  DbGrid1.DataSource:=dsqrData;
  Dbgrid1.Columns.Clear;
  qrData.SQL.Text:=SQL;
  Nomer:=cPass;
  Index:=cIdx;
end;

{--------------------------}
procedure TfmBrowse.FormShow;
{--------------------------}
var n, m, w : word;
begin
  if DBGrid1.DataSource=dsData then
```

```
  begin
    taData.DatabaseName    := pathdata;
    taData.IndexFieldNames := Index;
    taData.Open;
    DBLoadLabel(taData);

    if not taData.Findkey([Nomer]) then taData.First;
  //  taData.FindNearest([Nomer]);
  end else begin
   qrData.DatabaseName:= pathdata;
   qrData.Open;
   DBLoadLabel(qrData);
   if not qrData.Locate(INDEX,NOMER,[]) then qrData.First;
  end;
  Edit1.Text          := Nomer;
  StringGrid1.RowCount := DBgrid1.FieldCount;
  StringGrid1.Canvas.Font.Style := [fsBold];
  m := 0;
  for n := 0 to DBgrid1.FieldCount - 1 do begin
    If DBGrid1.Fields[n]<>NIL Then
    Begin
      StringGrid1.Cells[0, n] := DBGrid1.Fields[n].DisplayLabel + ' :';
      w := StringGrid1.Canvas.TextWidth(StringGrid1.Cells[0, n]);
      if w > m then m := w;
    End;
  end;
  StringGrid1.ColWidths[0] := m + 5;
  StringGrid1.ColWidths[1] := StringGrid1.Width - StringGrid1.ColWidths[0];
  Splitter1Moved(nil);
end;

{------------------------------}
procedure TfmBrowse.FormKeyPress;
{------------------------------}
begin
  if Key = #27 then Close;
  if Key = #13 then DBGrid1DblClick(Sender);
end;

{------------------------------}
procedure TfmBrowse.Edit1Change;
{------------------------------}
begin
  if DBGrid1.DataSource=dsData then
  begin
    if DBGrid1.SelectedField.IsIndexField then
       taData.FindNearest([Edit1.Text])
  end else begin
    if DbGrid1.SelectedField.FieldName=INDEX then
    begin
```

```
      qrData.Locate(INDEX,Edit1.Text,[]);
    end;
  end;
end;

{---------------------------------}
procedure TfmBrowse.DBGrid1DblClick;
{---------------------------------}
begin
  ModalResult := mrOk;
end;

{---------------------------------}
procedure TfmBrowse.DBGrid1ColEnter;
{---------------------------------}
begin
  Edit1.OnChange := nil;
  with dbgrid1.SelectedField do begin
    if (FieldKind = fkData) and (DataType = ftString) then begin
      SpeedButton1.Enabled := true;
      Edit1.Text := asString
    end else
      SpeedButton1.Enabled := false;
  end;
  Edit1.OnChange := Edit1Change;
end;

{-----------------------------------}
procedure TfmBrowse.SpeedButton1Click;
{-----------------------------------}
begin
 if DbGrid1.DataSource=DsData then
 begin
  if not DBGrid1.SelectedField.IsIndexField then
      taData.Locate(DBGrid1.SelectedField.FieldName,  Edit1.Text,  [loCaseInsensitive,
loPartialKey])
  else
    taData.Findkey([Edit1.Text])
 end else begin
  qrData.Locate(DBGrid1.SelectedField.FieldName,Edit1.Text,[]);
 end;

  Edit1.Text := DBGrid1.SelectedField.AsString;
end;

{--------------------------------}
procedure TfmBrowse.Splitter1Moved;
{--------------------------------}
begin
  StringGrid1.ColWidths[1] := StringGrid1.Width - StringGrid1.ColWidths[0];
```

```
  DBGrid1.Columns[1].Width := DBGrid1.Width - DBGrid1.Columns[0].Width - 1;
end;

{--------------------------------}
procedure TfmBrowse.Edit1MouseDown;
{--------------------------------}
begin
  Edit1.SelectAll;
end;

{---------------------------}
procedure TfmBrowse.FormClose;
{---------------------------}
var n : word;
begin
  for n := 0 to DBGrid1.Columns.Count - 1 do
    if DBgrid1.Fields[n] <> NIL Then
    Begin
      if DBgrid1.Fields[n].FieldKind = fkCalculated then
        DBgrid1.Fields[n].Free;
      if dbGrid1.DataSource=dsdata then
      nPosRec := taData.RecNo else
      NPosRec := qrData.RecNo
    End Else Break;
  taData.Close;
  qrData.Close;
end;

{-------------------------------------}
procedure TfmBrowse.StringGrid1DrawCell;
{-------------------------------------}
var str : string;
begin
  if Col = 0 then
    StringGrid1.Canvas.Font.Style := [fsBold]
  else
    StringGrid1.Canvas.Font.Style := [];

  StringGrid1.Canvas.Brush.Color := clWhite;
  str := StringGrid1.Cells[Col, Row];

  StringGrid1.Canvas.FillRect(Rect);
      DrawText(StringGrid1.Canvas.Handle,  PChar(str),  StrLen(PChar(str)),  Rect,
DT_LEFT);
end;

{-----------------------------------------------------------------}
procedure TfmBrowse.dsDataDataChange(Sender: TObject; Field: TField);
{-----------------------------------------------------------------}
var n : byte;
```

```
begin
  dsData.OnDataChange := nil;

  if ActiveControl = DBGrid1 then Edit1.Text := DBgrid1.SelectedField.AsString;
  for n := 0 to DBgrid1.Columns.Count - 1 do
  begin
    if DBGrid1.Fields[n] <> Nil Then
    Begin
        if DBgrid1.Fields[n].DataType = ftCurrency then
          StringGrid1.Cells[1, n] := format('%m', [DBgrid1.fields[n].asCurrency])
        else
          StringGrid1.Cells[1, n] := DBgrid1.fields[n].asString;
    end;
  end;

  dsData.OnDataChange := dsDataDataChange;
end;

procedure TfmBrowse.Edit1KeyDown(Sender: TObject; var Key: Word;
  Shift: TShiftState);
begin
  if KEY=VK_RETURN Then ModalResult := mrOK;
end;

end.
```

unit uDMEntry;

```
interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  Db, DBTables;

type
  TdmEntry = class(TDataModule)
    taSales: TTable;
    dsSales: TDataSource;
    taCust: TTable;
    dsCust: TDataSource;
    taSupl: TTable;
    dsSupl: TDataSource;
    taSuplKODE: TStringField;
    taSuplNAMA: TStringField;
    taSuplALAMAT: TStringField;
    taSuplKOTA: TStringField;
    taSuplNEGARA: TStringField;
    taSuplKODEPOS: TStringField;
    taSuplTELEPON1: TStringField;
```

```
    taSuplTELEPON2: TStringField;
    taSuplTELEPON3: TStringField;
    taSuplTELEPON4: TStringField;
    taSuplFAX: TStringField;
    taSuplHP: TStringField;
    taSuplEMAIL: TStringField;
    taSuplKONTAK: TStringField;
    taSalesKODE: TStringField;
    taSalesNAMA: TStringField;
    taSalesALAMAT: TStringField;
    taSalesKOTA: TStringField;
    taSalesNEGARA: TStringField;
    taSalesKODEPOS: TStringField;
    taSalesTELEPON: TStringField;
    taSalesFAX: TStringField;
    taSalesHP: TStringField;
    taSalesEMAIL: TStringField;
    taSalesPAGER: TStringField;
    taCustKODE: TStringField;
    taCustKODESALES: TStringField;
    taCustNAMA: TStringField;
    taCustALAMAT: TStringField;
    taCustKOTA: TStringField;
    taCustNEGARA: TStringField;
    taCustKODEPOS: TStringField;
    taCustKONTAK: TStringField;
    taCustTELEPON1: TStringField;
    taCustTELEPON2: TStringField;
    taCustTELEPON3: TStringField;
    taCustTELEPON4: TStringField;
    taCustFAX: TStringField;
    taCustHP: TStringField;
    taCustEMAIL: TStringField;
    taCustGRADE: TStringField;
    taCustMEMO: TMemoField;
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  dmEntry: TdmEntry;

implementation
uses allunits;

{$R *.DFM}

end.
```

```
unit uDMHutang;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  Db, DBTables, UCRPE32;

type
  TdmHutang = class(TDataModule)
    taCount: TTable;
    taSupl: TTable;
    taCust: TTable;
    taHut: TTable;
    taPiut: TTable;
    dsPLH: TDataSource;
    taPLH: TTable;
    dsPLHDtl: TDataSource;
    taPLHDtl: TTable;
    dsPLP: TDataSource;
    taPLP: TTable;
    dsPLPDtl: TDataSource;
    taPLPDtl: TTable;
    taPLHDtlSUBTOTAL: TCurrencyField;
    taPLPDtlSUBTOTAL: TCurrencyField;
    taPLPLookup: TTable;
    taPLHLookup: TTable;
    Crpe: TCrpe;
    taCountNAMA: TStringField;
    taCountNOMER: TStringField;
    taCountPANJANG: TSmallintField;
    taCustKODE: TStringField;
    taCustKODESALES: TStringField;
    taCustNAMA: TStringField;
    taCustALAMAT: TStringField;
    taCustKOTA: TStringField;
    taCustNEGARA: TStringField;
    taCustKODEPOS: TStringField;
    taCustKONTAK: TStringField;
    taCustTELEPON1: TStringField;
    taCustTELEPON2: TStringField;
    taCustTELEPON3: TStringField;
    taCustTELEPON4: TStringField;
    taCustFAX: TStringField;
    taCustHP: TStringField;
    taCustEMAIL: TStringField;
    taCustGRADE: TStringField;
    taCustMEMO: TMemoField;
    taHutNOMER: TStringField;
    taHutTANGGAL: TDateField;
```

taHutKODESUPL: TStringField;
taHutNAMASUPL: TStringField;
taHutTERM: TSmallintField;
taHutDEBET: TCurrencyField;
taHutKREDIT: TCurrencyField;
taHutBAYAR: TCurrencyField;
taHutID: TStringField;
taHutNOGIRO: TStringField;
taHutNOREFF: TStringField;
taHutIDREFF: TStringField;
taHutKETERANGAN: TMemoField;
taPLHNOMER: TStringField;
taPLHTANGGAL: TDateField;
taPLHKODESUPL: TStringField;
taPLHNAMASUPL: TStringField;
taPLHNOGIRO: TStringField;
taPLHTGLGIRO: TDateField;
taPLHJUMLAH: TCurrencyField;
taPLHKETERANGAN: TMemoField;
taPLHKELEBIHAN: TCurrencyField;
taPLHNAMAUSER: TStringField;
taPLHNAMAPRINT: TStringField;
taPLHVOID: TBooleanField;
taPLHDtlCOUNTER: TAutoIncField;
taPLHDtlNOMER: TStringField;
taPLHDtlID: TStringField;
taPLHDtlNONOTA: TStringField;
taPLHDtlTOTALNOTA: TCurrencyField;
taPLHDtlBAYAR: TCurrencyField;
taPLHDtlSISANOTA: TCurrencyField;
taPLHLookupNOMER: TStringField;
taPLHLookupTANGGAL: TDateField;
taPLHLookupKODESUPL: TStringField;
taPLHLookupNAMASUPL: TStringField;
taPLHLookupNOGIRO: TStringField;
taPLHLookupTGLGIRO: TDateField;
taPLHLookupJUMLAH: TCurrencyField;
taPLHLookupKETERANGAN: TMemoField;
taPLHLookupKELEBIHAN: TCurrencyField;
taPLHLookupNAMAUSER: TStringField;
taPLHLookupNAMAPRINT: TStringField;
taPLHLookupVOID: TBooleanField;
taPiutNOMER: TStringField;
taPiutTANGGAL: TDateField;
taPiutKODECUST: TStringField;
taPiutNAMACUST: TStringField;
taPiutTERM: TSmallintField;
taPiutDEBET: TCurrencyField;
taPiutKREDIT: TCurrencyField;
taPiutBAYAR: TCurrencyField;

```
  taPiutID: TStringField;
  taPiutNOGIRO: TStringField;
  taPiutNOREFF: TStringField;
  taPiutIDREFF: TStringField;
  taPiutKETERANGAN: TMemoField;
  taPLPLookupNOMER: TStringField;
  taPLPLookupTANGGAL: TDateField;
  taPLPLookupKODECUST: TStringField;
  taPLPLookupNAMACUST: TStringField;
  taPLPLookupJUMLAH: TCurrencyField;
  taPLPLookupNOGIRO: TStringField;
  taPLPLookupTGLGIRO: TDateField;
  taPLPLookupKETERANGAN: TMemoField;
  taPLPLookupKELEBIHAN: TCurrencyField;
  taPLPLookupNAMAUSER: TStringField;
  taPLPLookupNAMAPRINT: TStringField;
  taPLPLookupVOID: TBooleanField;
  taPLPDtlCOUNTER: TAutoIncField;
  taPLPDtlNOMER: TStringField;
  taPLPDtlID: TStringField;
  taPLPDtlNONOTA: TStringField;
  taPLPDtlTOTALNOTA: TCurrencyField;
  taPLPDtlBAYAR: TCurrencyField;
  taPLPDtlSISANOTA: TCurrencyField;
  taPLPNOMER: TStringField;
  taPLPTANGGAL: TDateField;
  taPLPKODECUST: TStringField;
  taPLPNAMACUST: TStringField;
  taPLPJUMLAH: TCurrencyField;
  taPLPNOGIRO: TStringField;
  taPLPTGLGIRO: TDateField;
  taPLPKETERANGAN: TMemoField;
  taPLPKELEBIHAN: TCurrencyField;
  taPLPNAMAUSER: TStringField;
  taPLPNAMAPRINT: TStringField;
  taPLPVOID: TBooleanField;
  procedure taPLHNewRecord(DataSet: TDataSet);
  procedure taPLHDtlNewRecord(DataSet: TDataSet);
  procedure taPLPNewRecord(DataSet: TDataSet);
  procedure taPLPDtlNewRecord(DataSet: TDataSet);
  procedure taPLHDtlAfterDelete(DataSet: TDataSet);
  procedure taPLPDtlAfterDelete(DataSet: TDataSet);
  procedure taPLHDtlCalcFields(DataSet: TDataSet);
  procedure taPLPDtlCalcFields(DataSet: TDataSet);

private
public
  nRunDtl : procedure(Sender: TObject) of object;

  nTotDtl, nTotByr, nTotNota : currency;
```

```
    cfield  : string;

    procedure GetTotal(Dataset : TDataset);
    procedure GetTotalPlh(Dataset : TDataset);
    procedure GetTotalPlhUpdate(Dataset : TDataset);
    procedure GetTotalPLP(Dataset : TDataset);
    procedure GetTotalPLPUpdate(Dataset : TDataset);
    function GetCount(id : string) : string;
    procedure Preview(NamaReport, NomerNota, vCaption: String);

  end;

var
  dmHutang: TdmHutang;

implementation

uses allunits;

{$R *.DFM}

function tDMHutang.GetCount;
var tsetup:ttable;
begin
  tsetup:= OpenTable('TSETUP', '', nil);

  taCount.open;
  taCount.First;
  if not taCount.FindKey([id]) then
  begin
    taCount.Append;
    taCountNAMA.Value    := id;
    taCountNOMER.Value   := '0000001';
    taCountPANJANG.Value := 7;
  end else
  begin
    taCount.Edit;
    taCountNOMER.Value := Inc1StrNum(taCountNOMER.Value, 7);
  end;

Result:=id+
 tsetup.fieldbyname('KODECAB').asString+strright(taCountNOMER.Value, 7);
  taCount.Post;
  taCount.Close;
  tsetup.close;
end;

procedure TdmHutang.GetTotal;
var oldrec   : TBookmark;
    oldstate : TDatasetState;
```

```
begin
  if (Dataset.State<>dsInsert) then
  begin
    Dataset.DisableControls;
    nTotDtl := 0;
    oldState := Dataset.State;
    oldrec  := Dataset.GetBookmark;
    Dataset.First;
    while not Dataset.EOF do
    begin
      nTotDtl := nTotDtl + Dataset.fieldbyname(cField).asCurrency;
      Dataset.Next;
    end;
    Dataset.GotoBookmark(oldrec);
    Dataset.FreeBookmark(oldrec);
    if assigned(nRunDtl) then nRunDtl(self);
    Dataset.EnableControls;
    if oldstate = dsEdit then DataSet.Edit;
  end;
end;

//******************** TdmHutang.GetTotalPlh ************************
procedure TdmHutang.GetTotalPlh;
var oldrec : TBookmark;
    OldState : TDataSetState;
    ID     : String;
begin
  if (Dataset.State<>dsInsert) then
  begin
      Dataset.DisableControls;
      nTotNota := 0;
      nTotByr  := 0;
      oldrec   := Dataset.GetBookmark;
      OldState := DataSet.State;
      Dataset.First;
      while not Dataset.EOF do
      begin
        nTotNota := nTotNota + Dataset.fieldbyname('TOTALNOTA').asCurrency;
        ID := DataSet.FieldByName('ID').AsString;
        if ID = 'BLKR' then
          nTotByr := nTotByr + Dataset.fieldbyname('BAYAR').asCurrency
        else
          nTotByr := nTotByr - Dataset.fieldbyname('BAYAR').asCurrency;
        Dataset.Next;
      end;

      Dataset.GotoBookmark(oldrec);
      Dataset.FreeBookmark(oldrec);

      if assigned(nRunDtl) then nRunDtl(self);
```

```
    Dataset.EnableControls;
    if oldstate = dsEdit then DataSet.Edit;
  end;
end;

//********************      TdmHutang.GetTotalPlhUpdate
*************************
procedure TdmHutang.GetTotalPlhUpdate;
var oldrec : TBookmark;
   ID     : String;
begin
  Dataset.DisableControls;
  nTotNota := 0;
  nTotByr  := 0;
  oldrec   := Dataset.GetBookmark;
  Dataset.First;
  while not Dataset.EOF do
  begin
    nTotNota := nTotNota + Dataset.fieldbyname('TOTALNOTA').asCurrency;
    ID := DataSet.FieldByName('ID').AsString;
    if ID='BLKR' then
      nTotByr := nTotByr + Dataset.fieldbyname('BAYAR').asCurrency
    else
      nTotByr := nTotByr - Dataset.fieldbyname('BAYAR').asCurrency;
    Dataset.Next;
  end;
  Dataset.GotoBookmark(oldrec);
  Dataset.FreeBookmark(oldrec);
  if assigned(nRunDtl) then nRunDtl(self);
  Dataset.EnableControls;
end;

//******************** TdmHutang.GetTotalPLP ************************
procedure TdmHutang.GetTotalPLP;
var oldrec : TBookmark;
   oldState : TDataSetState;
   ID     : String;
begin
  if (Dataset.State<>dsInsert) then
  begin
    Dataset.DisableControls;
    nTotNota := 0;
    nTotByr  := 0;
    oldrec  := Dataset.GetBookmark;
    OldState := Dataset.State;
    Dataset.First;
    while not Dataset.EOF do
    begin
      nTotNota := nTotNota + Dataset.fieldbyname('TOTALNOTA').asCurrency;
      ID := DataSet.FieldByName('ID').AsString;
```

```
      if ID = 'JLKR' then
        nTotByr := nTotByr + Dataset.fieldbyname('BAYAR').asCurrency
      else
        nTotByr := nTotByr - Dataset.fieldbyname('BAYAR').asCurrency;
      Dataset.Next;
    end;

    Dataset.GotoBookmark(oldrec);
    Dataset.FreeBookmark(oldrec);
    if assigned(nRunDtl) then nRunDtl(self);
    Dataset.EnableControls;
    if oldstate = dsEdit then DataSet.Edit;
  end;
end;

//********************    TdmHutang.GetTotalPLPUpdate
*************************
procedure TdmHutang.GetTotalPLPUpdate;
var oldrec : TBookmark;
   ID     : String;
begin
  Dataset.DisableControls;
  nTotNota := 0;
  nTotByr  := 0;
  oldrec  := Dataset.GetBookmark;
  Dataset.First;
  while not Dataset.EOF do
  begin
    nTotNota := nTotNota + Dataset.fieldbyname('TOTALNOTA').asCurrency;
    ID := DataSet.FieldByName('ID').AsString;
    if ID='JLKR' then
      nTotByr := nTotByr + Dataset.fieldbyname('BAYAR').asCurrency
    else
      nTotByr := nTotByr - Dataset.fieldbyname('BAYAR').asCurrency;
    Dataset.Next;
  end;
  Dataset.GotoBookmark(oldrec);
  Dataset.FreeBookmark(oldrec);
  if assigned(nRunDtl) then nRunDtl(self);
  Dataset.EnableControls;
end;

procedure TdmHutang.taPLHNewRecord(DataSet: TDataSet);
begin
  taPLH.DisableControls;                { supaya nggak berkedip }
  taPLHNOMER.Value   := GetCount('PH');
  DBInitValue(taPLH);
  taPLHTglGiro.AsString := '';
  taPLH.EnableControls;
end;
```

```
procedure TdmHutang.taPLHDtlNewRecord(DataSet: TDataSet);
begin
  taPLHDtlID.Value := 'BLKR';
  DBInitValue(taPLHDtl);
end;

procedure TdmHutang.taPLPNewRecord(DataSet: TDataSet);
begin
  taPLP.DisableControls;                { supaya nggak berkedip }
  taPLPNOMER.Value := GetCount('PP');

  DBInitValue(taPLP);

  taPLPTglGiro.AsString := '';
  taPLP.EnableControls;
end;

procedure TdmHutang.taPLPDtlNewRecord(DataSet: TDataSet);
begin
  taPLPDtlID.Value := 'JLKR';
  DBInitValue(taPLPDtl);
end;

procedure TdmHutang.taPLHDtlAfterDelete(DataSet: TDataSet);
begin
  GetTotalPlh(taPlhdtl);
end;

procedure TdmHutang.taPLPDtlAfterDelete(DataSet: TDataSet);
begin
  GetTotalPLP(taPlPdtl);
end;

procedure TdmHutang.taPLHDtlCalcFields(DataSet: TDataSet);
begin
  if taPLHDTLID.Value = 'BLKR' then
    taPLHDtlSUBTOTAL.Value := taPLHDtlBAYAR.Value
  else
    taPLHDtlSUBTOTAL.Value := taPLHDtlBAYAR.Value * -1;
end;

procedure TdmHutang.taPLPDtlCalcFields(DataSet: TDataSet);
begin
  if taPLPDTLID.Value = 'JLKR' then
    taPLPDtlSUBTOTAL.Value := taPLPDtlBAYAR.Value
  else
    taPLPDtlSUBTOTAL.Value := taPLPDtlBAYAR.Value* -1;
end;
```

```
procedure tDMHutang.Preview(NamaReport, NomerNota, vCaption: String);
begin
  Crpe.Clear;
  Crpe.ReportName := PathReport + '\'+ NamaReport;
  Crpe.Tables.Retrieve;
  Crpe.Tables[0].Path := PathData;
  Crpe.Tables.Propagate := True;
  Crpe.WindowStyle.Title := vCaption;
  Crpe.WindowSize.Height := Screen.Height;
  Crpe.WindowSize.Width := Screen.Width;
  Crpe.WindowSize.Left := 0;
  Crpe.WindowSize.Top := 0;
  Crpe.ParamFields.Retrieve;
  Crpe.ParamFields[0].Value := UserLogin;
  Crpe.ParamFields[1].Value := NomerNota;
  Crpe.Execute;
end;

end.
```

```
unit UDmInventory;
interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  Db, DBTables, UCRPE32;

type
  TdmInventory = class(TDataModule)
    taBarang: TTable;
    dsGudang: TDataSource;
    dsBarang: TDataSource;
    taGdgBrg: TTable;
    Crpe: TCrpe;
    taBarangKODE: TStringField;
    taBarangNAMA: TStringField;
    taBarangVENDORCODE: TStringField;
    taBarangSATUAN: TStringField;
    taBarangKETERANGAN: TMemoField;
    taAdin: TTable;
    taAdinNOMER: TStringField;
    taAdinTANGGAL: TDateField;
    taAdinNAMAUSER: TStringField;
    taAdinNAMAPRINT: TStringField;
    taAdinTOTAL: TCurrencyField;
    taAdinKETERANGAN: TMemoField;
    taAdinPRINTED: TSmallintField;
    taAdinDtl: TTable;
    taAdinDtlCOUNTER: TAutoIncField;
```

taAdinDtlNOMER: TStringField;
taAdinDtlKODEBRG: TStringField;
taAdinDtlNAMABRG: TStringField;
taAdinDtlJUMLAH: TFloatField;
taAdinDtlSATUAN: TStringField;
taAdinDtlHARGA: TCurrencyField;
taAdinDtlTOTAL: TCurrencyField;
dsAdin: TDataSource;
dsAdinDtl: TDataSource;
taAdou: TTable;
taAdouNOMER: TStringField;
taAdouTANGGAL: TDateField;
taAdouKETERANGAN: TMemoField;
taAdouPRINTED: TSmallintField;
taAdouNAMAUSER: TStringField;
taAdouNAMAPRINT: TStringField;
taAdouTOTAL: TCurrencyField;
taAdouDtl: TTable;
taAdouDtlCOUNTER: TAutoIncField;
taAdouDtlNOMER: TStringField;
taAdouDtlKODEBRG: TStringField;
taAdouDtlNAMABRG: TStringField;
taAdouDtlJUMLAH: TFloatField;
taAdouDtlSATUAN: TStringField;
taAdouDtlHARGA: TCurrencyField;
taAdouDtlTOTAL: TCurrencyField;
dsAdou: TDataSource;
dsAdouDtl: TDataSource;
taGdgBrgKODEBRG: TStringField;
taGdgBrgSALDO: TFloatField;
taService: TTable;
taServiceKODE: TStringField;
taServiceNAMA: TStringField;
taServiceHARGA: TCurrencyField;
taServiceKETERANGAN: TMemoField;
dsService: TDataSource;
taSrv: TTable;
dsSrv: TDataSource;
taSrvDtl: TTable;
dsSrvDtl: TDataSource;
taSrvDtlCOUNTER: TAutoIncField;
taSrvDtlNOMER: TStringField;
taSrvDtlKODEBRG: TStringField;
taSrvDtlNAMABRG: TStringField;
taSrvDtlJUMLAH: TFloatField;
taSrvDtlHARGA: TCurrencyField;
taSrvNOMER: TStringField;
taSrvTANGGAL: TDateField;
taSrvNAMAUSER: TStringField;
taSrvNAMAPRINT: TStringField;

```
    taSrvTOTAL: TCurrencyField;
    taSrvKETERANGAN: TMemoField;
    taSrvPRINTED: TSmallintField;
    taSrvDtlTOTAL: TCurrencyField;
  private
  public
  end;

var
  dmInventory: TdmInventory;

implementation

{$R *.DFM}

end.
```

unit UDmPembelian;

```
interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  Db, DBTables, UCRPE32;

type
  TdmPembelian = class(TDataModule)
    dsBeli: TDataSource;
    dsBelidtl: TDataSource;
    taRtbl: TTable;
    taRtbldtl: TTable;
    dsRtbl: TDataSource;
    dsRtbldtl: TDataSource;
    taHut: TTable;
    Crpe1: TCrpe;
    taRtblNOMER: TStringField;
    taRtblTANGGAL: TDateField;
    taRtblKODESUPL: TStringField;
    taRtblNAMASUPL: TStringField;
    taRtblALAMAT: TStringField;
    taRtblKOTA: TStringField;
    taRtblNEGARA: TStringField;
    taRtblKETERANGAN: TMemoField;
    taRtblJENIS: TSmallintField;
    taRtblTERM: TSmallintField;
    taRtblDISC: TCurrencyField;
    taRtblPPN: TCurrencyField;
    taRtblPRINTED: TSmallintField;
    taRtblNAMAUSER: TStringField;
```

```
    taRtblNAMAPRINT: TStringField;
    taRtblVOID: TBooleanField;
    taRtblTOTNOTA: TCurrencyField;
    taRtbldtlCOUNTER: TAutoIncField;
    taRtbldtlNOMER: TStringField;
    taRtbldtlKODEBRG: TStringField;
    taRtbldtlNAMABRG: TStringField;
    taRtbldtlJUMLAH: TFloatField;
    taRtbldtlSATUAN: TStringField;
    taRtbldtlHARGA: TCurrencyField;
    taRtbldtlDISC: TFloatField;
    taRtbldtlTOTAL: TCurrencyField;
    taBelidtl: TTable;
    taBelidtlCOUNTER: TAutoIncField;
    taBelidtlNOMER: TStringField;
    taBelidtlKODEBRG: TStringField;
    taBelidtlNAMABRG: TStringField;
    taBelidtlJUMLAH: TFloatField;
    taBelidtlSATUAN: TStringField;
    taBelidtlHARGA: TCurrencyField;
    taBelidtlDISC: TFloatField;
    taBelidtlTOTAL: TCurrencyField;
    taBeli: TTable;
    taBeliNOMER: TStringField;
    taBeliTANGGAL: TDateField;
    taBeliKODESUPL: TStringField;
    taBeliNAMASUPL: TStringField;
    taBeliALAMAT: TStringField;
    taBeliKOTA: TStringField;
    taBeliNEGARA: TStringField;
    taBeliKETERANGAN: TMemoField;
    taBeliTERM: TSmallintField;
    taBeliJENIS: TSmallintField;
    taBeliDISC: TCurrencyField;
    taBeliPPN: TCurrencyField;
    taBeliNAMAUSER: TStringField;
    taBeliNAMAPRINT: TStringField;
    taBeliPRINTED: TSmallintField;
    taBeliVOID: TBooleanField;
    taBeliTOTNOTA: TCurrencyField;
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  dmPembelian: TdmPembelian;

implementation
```

```
{$R *.DFM}

end.
```

```
unit UDmPenjualan;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  Db, DBTables, Ucrpe32;

type
  TdmPenjualan = class(TDataModule)
    taJual: TTable;
    taJualDtl: TTable;
    dsJual: TDataSource;
    dsJualdtl: TDataSource;
    taRtjl: TTable;
    taRtjldtl: TTable;
    dsRtjl: TDataSource;
    dsRtjldtl: TDataSource;
    taJualDtlCOUNTER: TAutoIncField;
    taJualDtlNOMER: TStringField;
    taJualDtlKODEBRG: TStringField;
    taJualDtlNAMABRG: TStringField;
    taJualDtlSATUAN: TStringField;
    taJualDtlJUMLAH: TFloatField;
    taJualDtlHARGA: TCurrencyField;
    taJualDtlDISC: TFloatField;
    taJualDtlTOTAL: TCurrencyField;
    taRtjldtlCOUNTER: TAutoIncField;
    taRtjldtlNOMER: TStringField;
    taRtjldtlKODEBRG: TStringField;
    taRtjldtlNAMABRG: TStringField;
    taRtjldtlJUMLAH: TFloatField;
    taRtjldtlSATUAN: TStringField;
    taRtjldtlHARGA: TCurrencyField;
    taRtjldtlDISC: TCurrencyField;
    taRtjldtlTOTAL: TCurrencyField;
    taRtjldtlSTOKKODEBRG: TStringField;
    taRtjldtlIS_PAKET: TBooleanField;
    taPiut: TTable;
    Crpe1: TCrpe;
    taRtjlNOMER: TStringField;
    taRtjlTANGGAL: TDateField;
    taRtjlKODECUST: TStringField;
    taRtjlNAMACUST: TStringField;
    taRtjlALAMAT: TStringField;
```

```
    taRtjlKOTA: TStringField;
    taRtjlNEGARA: TStringField;
    taRtjlKETERANGAN: TMemoField;
    taRtjlJENIS: TSmallintField;
    taRtjlTERM: TSmallintField;
    taRtjlDISC: TCurrencyField;
    taRtjlPPN: TCurrencyField;
    taRtjlPRINTED: TSmallintField;
    taRtjlNAMAUSER: TStringField;
    taRtjlNAMAPRINT: TStringField;
    taRtjlVOID: TBooleanField;
    taRtjlTOTNOTA: TCurrencyField;
    taJualNOMER: TStringField;
    taJualTANGGAL: TDateField;
    taJualKODECUST: TStringField;
    taJualNAMACUST: TStringField;
    taJualALAMAT: TStringField;
    taJualKOTA: TStringField;
    taJualNEGARA: TStringField;
    taJualKETERANGAN: TMemoField;
    taJualTERM: TSmallintField;
    taJualJENIS: TSmallintField;
    taJualDISC: TCurrencyField;
    taJualPPN: TCurrencyField;
    taJualNAMAUSER: TStringField;
    taJualNAMAPRINT: TStringField;
    taJualPRINTED: TSmallintField;
    taJualSTATPESAN: TStringField;
    taJualVOID: TBooleanField;
    taJualTOTNOTA: TCurrencyField;
    procedure Crpe1Error(const ErrorNum: Smallint;
      const ErrorString: String; var IgnoreError: Boolean);
  private
    { Private declarations }

  public
    { Public declarations }
  end;

var
  dmPenjualan: TdmPenjualan;

implementation
{$R *.DFM}



procedure TdmPenjualan.Crpe1Error(const ErrorNum: Smallint;
  const ErrorString: String; var IgnoreError: Boolean);
begin
```

```
  if ErrorNum=656 then
    IgnoreError:=True;
end;

end.
```

```
unit UJual;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  Menus, Grids, DBGrids, ComCtrls, ZKNavigator, ExtCtrls, StdCtrls,
  DBCtrls, Mask, Db, DBTables, MRUList, ToolEdit, RXDBCtrl, RxLookup,
  CurrEdit, Buttons, uCrpe32, RXCtrls;

type
  TfmJual = class(TForm)
    dsWatch: TDataSource;
    Panel4: TPanel;
    Label3: TLabel;
    Label4: TLabel;
    Label5: TLabel;
    Label6: TLabel;
    Label8: TLabel;
    deKodeCust: TRxDBComboEdit;
    Panel6: TPanel;
    dePpn: TDBEdit;
    Label10: TLabel;
    Label12: TLabel;
    Label13: TLabel;
    Panel3: TPanel;
    DBEdit2: TDBEdit;
    DBEdit3: TDBEdit;
    DBEdit4: TDBEdit;
    DBEdit5: TDBEdit;
    Panel2: TPanel;
    Label14: TLabel;
    deTerm: TDBEdit;
    cePpn: TCurrencyEdit;
    Panel5: TPanel;
    Panel9: TPanel;
    dsDetail: TDataSource;
    Label30: TLabel;
    DBMemo1: TDBMemo;
    DBGrid1: TRxDBGrid;
    Panel7: TPanel;
    Label22: TLabel;
    Label23: TLabel;
```

```
  Panel8: TPanel;
  DBText1: TDBText;
  KNav: TK2Navigator;
  DBCheckBox1: TDBCheckBox;
  Bevel3: TBevel;
  CheckBox1: TCheckBox;
  PopupMenu1: TPopupMenu;
  Add1: TMenuItem;
  Edit1: TMenuItem;
  Delete1: TMenuItem;
  Simpan1: TMenuItem;
  Batal1: TMenuItem;
  deTanggal: TDBDateEdit;
  DBRadioGroup1: TDBRadioGroup;
  rgJenisBayar: TDBRadioGroup;
  Label11: TLabel;
  ceDisc: TCurrencyEdit;
  deDisc: TDBEdit;
  Label7: TLabel;
  procedure FormClose(Sender: TObject; var Action: TCloseAction);
  procedure FormShow(Sender: TObject);
  procedure dsWatchStateChange(Sender: TObject);
  procedure KNavCancelClick(Sender: TObject);
  procedure KNavAddClick(Sender: TObject);
  procedure deKodeCustButtonClick(Sender: TObject);
  procedure KNavSaveClick(Sender: TObject);
  procedure cePpnExit(Sender: TObject);
  procedure dsWatchDataChange(Sender: TObject; Field: TField);
  procedure DBGrid1EditButtonClick(Sender: TObject);
  procedure dsDetailDataChange(Sender: TObject; Field: TField);
  procedure CheckBox1Enter(Sender: TObject);
  procedure Add1Click(Sender: TObject);
  procedure Edit1Click(Sender: TObject);
  procedure Delete1Click(Sender: TObject);
  procedure Simpan1Click(Sender: TObject);
  procedure Batal1Click(Sender: TObject);
  procedure PopupMenu1Popup(Sender: TObject);
  procedure KNavPreviewClick(Sender: TObject);
  procedure KNavSearchClick(Sender: TObject);
  procedure FormCloseQuery(Sender: TObject; var CanClose: Boolean);
  procedure DBGrid1ColEnter(Sender: TObject);
  procedure DBGrid1KeyPress(Sender: TObject; var Key: Char);
  procedure rgJenisBayarChange(Sender: TObject);
  procedure ceDiscExit(Sender: TObject);
private
  taBrg, taCust, taGdgBrg : TTable;
  subtotal, grandtotal    : real;
  oldkodebrg              : string;

  procedure SettingEdit(nVal : boolean);
```

```
    procedure Cancel_All_Updates;

    procedure SetEvent ;
    procedure ClearEvent;

    procedure taJualKODECUSTChange(Sender: TField);

    procedure taJualPPNChange(Sender: TField);
    procedure taJualDISCChange(Sender: TField);

    procedure taJualdtlKODEBRGChange(Sender: TField);

    procedure taJualDtlCalcFields(DataSet: TDataSet);
    procedure taJualDtlNewRecord(DataSet: TDataSet);
    procedure taJualDtlBeforeDelete(DataSet: TDataSet);

    procedure AfterScroll(DataSet : TDataSet);

    Function HitungTotNota:Currency;
  public
  end;

var
  fmJual: TfmJual;

implementation

uses AllUnits, UDmPenjualan, UBrowse, uSearch;

{$R *.DFM}

Procedure TfmJual.AfterScroll(DataSet : TDataSet);
Begin
  with dmPenjualan do
  begin
    if taJualTOTNOTA.Value = 0 Then
      cePpn.Value  := 0
    else cePpn.Value  := taJualPPN.Value * 100 / taJualTOTNOTA.Value;

    if taJualTOTNOTA.Value = 0 Then
      cePpn.Value  := 0
    else cePpn.Value  := taJualPPN.Value * 100 / taJualTOTNOTA.Value;

    subtotal       := DBGetTotal(taJualdtlTOTAL);
    Panel5.Caption := format('%m ', [subtotal]);
    grandtotal     := subtotal - taJualDISC.Value + taJualPPN.Value;
    Panel9.Caption := format('%m ', [grandtotal]);
  end;
end;
```

```
procedure TfmJual.SetEvent;
begin
  with dmPenjualan do
  begin
    taJualKODECUST.OnChange   := taJualKODECUSTChange;
    taJualPPN.OnChange        := taJualPPNChange;
    taJualDISC.OnChange       := taJualDISCChange;
    taJualDtlKODEBRG.OnChange := taJualdtlKODEBRGChange;
    taJualdtl.OnNewRecord     := taJualDtlNewRecord;
    taJualdtl.BeforeDelete    := taJualDtlBeforeDelete;
    taJualDtl.OnCalcFields    := taJualDtlCalcFields;
  end;
end;

procedure TfmJual.ClearEvent;
begin
  with dmPenjualan do
  begin
    taJualKODECUST.OnChange   := nil;
    taJualPPN.OnChange        := nil;
    taJualDISC.OnChange       := nil;
    taJualDtlKODEBRG.OnChange := nil;
    taJualdtl.BeforeInsert    := nil;
    taJualdtl.OnNewRecord     := nil;
    taJualdtl.BeforeDelete    := nil;
    taJualDtl.OnCalcFields    := nil;
  end;
end;

procedure TfmJual.Cancel_All_Updates;
begin
  with dmPenjualan do
  begin
    taJualDtl.CancelUpdates;
    taJual.Cancel;
  end;
end;

procedure TfmJual.SettingEdit(nVal : boolean);
var n : integer;
begin
  for n := 0 to fmJual.ComponentCount - 1 do
if fmJual.Components[n].Tag=100 then
TControl(fmJual.Components[n]).Enabled := nVal;
end;

procedure TfmJual.taJualKODECUSTChange(Sender: TField);
begin
  with dmPenjualan do
  begin
```

```
    taJualKODECUST.OnChange := nil;          { supaya jangan rekursif }

    if not taCust.FindKey([Sender.asString]) then deKodeCUSTButtonClick(nil);

    taJualKODECUST.Value := taCust.FieldByName('KODE').asString;
    taJualNAMACUST.Value := taCust.FieldByName('NAMA').asString;
    taJualALAMAT.Value   := taCust.FieldByName('ALAMAT').asString;
    taJualKOTA.Value     := taCust.FieldByName('KOTA').asString;
    taJualNEGARA.Value   := taCust.FieldByName('NEGARA').asString;

    { jika Nama, Alamat, Kota, Negara ada isinya maka cuma readonly }
    DBEdit2.ReadOnly := not taCust.FieldByName('NAMA').isNull;
    DBEdit3.ReadOnly := not taCust.FieldByName('ALAMAT').isNull;
    DBEdit4.ReadOnly := not taCust.FieldByName('KOTA').isNull;
    DBEdit5.ReadOnly := not taCust.FieldByName('NEGARA').isNull;

    taJualKODECUST.OnChange := taJualKODECUSTChange;
  end;
end;

procedure TfmJual.taJualPPNChange(Sender: TField);
begin
  if subtotal > 0 then cePpn.Value := sender.asCurrency * 100 / subtotal;
end;

procedure TfmJual.taJualDISCChange(Sender: TField);
begin
  if subtotal > 0 then
    ceDisc.Value := sender.asCurrency * 100 / subtotal;
end;

procedure TfmJual.taJualdtlKODEBRGChange(Sender: TField);
begin
  with dmPenjualan do
  begin
    dsDetail.OnDataChange     := nil;

    if not taBrg.FindKey([Sender.asString]) then DBGrid1EditButtonClick(nil);

    taJualdtlKODEBRG.OnChange := nil;

    taJualDtl.Edit;
    taJualDtlKODEBRG.Value := taBrg.FieldByName('KODE').asString;
    taJualDtlNAMABRG.Value := taBrg.FieldByName('NAMA').asString;
    taJualDtlSATUAN.Value  := taBrg.FieldByName('SATUAN').asString;
    taJualDtlJUMLAH.Value  := 1;
    taJualDtlHARGA.Value   := 0;
    taJualdtlDISC.Value    := 0;

    taJualdtl.Post;
```

```
    taJualdtlKODEBRG.OnChange := taJualdtlKODEBRGChange;
    dsDetail.OnDataChange     := dsDetailDataChange;
  end;
end;

procedure TfmJual.taJualDtlNewRecord;
begin
  DBInitValue(dmPenjualan.taJualDtl);
  DBgrid1.SelectedIndex := 0;
end;

procedure TfmJual.taJualDtlCalcFields;
begin
  with dmPenjualan do
     taJualdtlTOTAL.Value := (taJualDtlJUMLAH.Value * taJualDtlHARGA.Value) -
taJualDtlDISC.Value ;
end;

procedure TfmJual.taJualDtlBeforeDelete(DataSet: TDataSet);
begin
  if dsWatch.State <> dsBrowse then
     if MessageDlg('Hapus record ... ?',mtConfirmation, [mbYes, mbNo], 0) <> mrYes
then abort;
end;

procedure TfmJual.FormShow;
begin
  with dmPenjualan do
  begin
    DBSetPath(dmPenjualan);
    DBLoadLabel(taJualdtl);

    taGdgBrg   := OpenTable('TGdgBrg' , '', fmJual); { updatable }
    taCust     := OpenTable('TCUST'   , '', fmJual);
    taBrg      := OpenTable('TBARANG' , '', fmJual);

    taJualdtl.Open;
    taJual.Open;

    SetEvent;
    taJual.AfterScroll := AfterScroll;
    AfterScroll(taJual);
  end;
  deTanggal.SetFocus;

end;

procedure TfmJual.FormClose;
begin
  with dmPenjualan do
```

```
  begin
    taCust.Close;
    taBrg.Close;
    taGdgBrg.Close;
    taJualdtl.Close;
    taJual.Close;
    taJual.AfterScroll := NIL;
  end;
end;

procedure TfmJual.KNavCancelClick;
begin
  KNav.SetFocus;
  with dmPenjualan do begin
    case dsWatch.State of
      dsInsert : begin
                   taJualNAMAUSER.Value := UserLogin;
                   taJualVOID.Value     := true;
                   taJual.Post;
                   taJualdtl.CommitUpdates;
                 end;
    end;
    Cancel_All_Updates;
    deTanggal.SetFocus;
  end;
end;


procedure TfmJual.KNavAddClick;
begin
  ClearEvent;
  with dmPenjualan do begin
    taJual.Append;
    DBInitValue(taJual);
    taJualNOMER.Value    := DBGetCounter('JL', fmJual);
    taJualJENIS.Value    := 0;
    taJualTERM.Value     := 0;
    taJualSTATPESAN.value:='Ambil';
    taJualDtl.EnableControls;
  end;

  subtotal                := 0;
  deKodeCust.Button.Enabled := true;
  deTanggal.SetFocus;
  SetEvent;
end;

procedure TfmJual.KNavSaveClick;
begin
  with dmPenjualan do begin
```

```
  KNav.SetFocus;
  dsDetail.OnDataChange := nil;  { supaya jangan rekursif }

   taJual.DisableControls;        { supaya semua proses di dlm save tidak berpengaruh
pada control }
  taJualdtl.DisableControls;
  try
     if taJualTANGGAL.IsNull                    then DispErr('Tanggal tidak boleh
kosong', deTanggal);
     if taJualTANGGAL.Value <> Date             then DispErr('Tanggal harus =
tanggal sekarang', deTanggal);

    if taJualKODECUST.IsNull                    then DispErr('Kode Customer tidak
boleh kosong', deKodeCust);
     if not taCust.FindKey([taJualKODECUST.Text])  then DispErr('Kode Customer
tidak ditemukan', deKodeCust);

      if ((taJualTERM.Value < 0) and (taJualJENIS.Value = 1)) then disperr('Term
harus > 0',deTerm);

    { *** check detail *** }
    taJualdtl.First;
    while not taJualdtl.EOF do
    begin
       if taJualdtlKODEBRG.IsNull               then DispErr('Kode Barang tidak
boleh kosong', DBGrid1);
      if not taBrg.FindKey([taJualdtlKODEBRG.Value]) then DispErr('Kode Barang
tidak ditemukan', DBGrid1);

     if not taGdgBrg.FindKey([taJualdtlKODEBRG.Value]) then
       DispErr('Barang ' + taJualdtlKODEBRG.Value + ' tidak ditemukan di gudang ',
DBGrid1);

      if taGdgBrg.FieldByName('SALDO').AsInteger - taJualDtlJUMLAH.Value < 0
then
       DispErr('Stok barang ' + taJualdtlKODEBRG.Value + ' tidak mencukupi (' +
            FloatToStr(taGdgBrg.FieldByName('SALDO').AsInteger -
taJualDtlJUMLAH.Value) + ')', DBGrid1);

       if taJualdtlJUMLAH.IsNull                then DispErr('Jumlah tidak boleh
kosong', DBGrid1);
      if taJualDtlJUMLAH.Value < 0              then DispErr('Jumlah tidak boleh <
0', DBGrid1);

      if taJualdtlSATUAN.IsNull                 then DispErr('Satuan tidak boleh
kosong', DBGrid1);
       if taJualDtlHARGA.IsNull                 then DispErr('Harga tidak boleh
kosong', DBGrid1);
```

```
      if taJualDtlHARGA.Value < 0                then DispErr('Harga tidak boleh < 0',
DBGrid1);

      taJualdtl.Next;
    end;

    if not taPiut.Active then taPiut.Open;

    if not taPiut.FindKey([taJualNOMER.Value]) then
      taPiut.Append
    else taPiut.Edit;

    taPiut.Fieldbyname('NOMER').asString        := taJualNOMER.Value;
    taPiut.Fieldbyname('TANGGAL').asDateTime    := taJualTANGGAL.Value;
    taPiut.Fieldbyname('KODECUST').asString     := taJualKODECUST.Value;
    taPiut.Fieldbyname('NAMACUST').asString     := taJualNAMACUST.Value;
    taPiut.Fieldbyname('NOREFF').asString       := '';
    taPiut.Fieldbyname('IDREFF').asString       := '';
    taPiut.Fieldbyname('TERM').asInteger        := taJualTERM.Value;
    taPiut.Fieldbyname('DEBET').asCurrency      := GrandTotal;
    taPiut.Fieldbyname('KREDIT').asCurrency     := 0;

    if taJualJENIS.Value = 0 then
    begin
      taPiut.Fieldbyname('KETERANGAN').asString := 'Penjualan Tunai';
      taPiut.Fieldbyname('ID').asString         := 'JLTN';
      taPiut.Fieldbyname('BAYAR').asCurrency    := GrandTotal;
    end else
    begin
      taPiut.Fieldbyname('KETERANGAN').asString := 'Penjualan Kredit';
      taPiut.Fieldbyname('ID').asString         := 'JLKR';
      taPiut.Fieldbyname('BAYAR').asCurrency    := 0;
    end;

    taPiut.Post;

    { force setting }
    taJualNAMAUSER.Value := UserLogin;
    taJual.Post;
    taJualdtl.CommitUpdates;

    taJual.Edit;
    taJualTOTNOTA.Value  := HitungTotNota;
    DBInitValue2(taJual);
    taJual.Post;

    taJualdtl.First;
    deTanggal.SetFocus;

  finally
```

```
    dsDetail.OnDataChange := dsDetailDataChange; { supaya jangan rekursif }
    taJual.EnableControls;
    taJualdtl.EnableControls;
   end;
  end;
end;

procedure TfmJual.DBGrid1EditButtonClick(Sender: TObject);
begin
  with dmPenjualan do
  begin
    oldkodebrg:=taJualDtlKODEBRG.Value ;

    fmBrowse.brwTBARANG(DBGrid1.SelectedField.AsString, nil);
    taJualDtl.Edit;
    taJualDtlKODEBRG.Value := fmBrowse.StringGrid1.Cells[1, 0];
  end;
end;

procedure TfmJual.deKodeCustButtonClick;
begin
  fmBrowse.brwTCUST(deKodeCust.Text, nil);
  taCust.FindKey([fmBrowse.StringGrid1.Cells[1, 0]]);
  dmPenjualan.taJualKODECUST.Value := fmBrowse.StringGrid1.Cells[1, 0];
end;

procedure TfmJual.cePpnExit(Sender: TObject);
begin
  if dsWatch.State in [dsEdit, dsInsert] then
    dmPenjualan.taJualPPN.Value := (subtotal * cePpn.Value) / 100;
end;

procedure TfmJual.ceDiscExit(Sender: TObject);
begin
  if dsWatch.State in [dsEdit, dsInsert] then
    dmPenjualan.taJualDISC.Value := (subtotal * ceDisc.Value) / 100;
end;

procedure TfmJual.dsWatchStateChange;
begin
  if (dsWatch.State = dsEdit) then
    SettingEdit(True);

  deKodeCust.Button.Enabled := dsWatch.State in [dsEdit, dsInsert];

  ceDisc.ReadOnly          := dsWatch.State = dsBrowse;
  cePpn.ReadOnly           := dsWatch.State = dsBrowse;

  if dsWatch.State = dsBrowse then
    DBGrid1.Options := DBGrid1.Options - [dgEditing]
```

```
  else
    DBGrid1.Options := DBGrid1.Options + [dgEditing];
end;

procedure TfmJual.dsWatchDataChange(Sender: TObject; Field: TField);
begin
  dsWatch.OnDataChange := nil;

  try
    with dmPenjualan do
    begin
      deDisc.ReadOnly   := subtotal = 0;
      dePpn.ReadOnly    := subtotal = 0;
      CheckBox1.Checked := taJualPRINTED.Value > 0;

      if (subtotal > 0) and (field = nil) and (dsWatch.State <> dsBrowse) then begin
        taJualDISC.Value := subtotal * ceDISC.Value / 100;
        taJualPPN.Value  := subtotal * cePPN.Value / 100;
      end;

      grandtotal     := subtotal - tajualdisc.value + taJualPPN.Value;
      Panel9.Caption := format('%m ', [grandtotal]);
    end;
  except
  end;

  dsWatch.OnDataChange := dsWatchDataChange;
end;

procedure TfmJual.dsDetailDataChange(Sender: TObject; Field: TField);
var nSatuan : word;
begin
  dsDetail.OnDataChange := nil;  { supaya jangan rekursif }

  try
    with dmPenjualan do
    begin
      { cari posisi kolom untuk field satuan, ini penting
        soalnya kolom dapat dipindah-pindah }
       for nSatuan := 0 to DBgrid1.FieldCount - 1 do if DBgrid1.Fields[nSatuan] =
taJualdtlSATUAN then break;
      DBGrid1.Columns[nSatuan].PickList.Clear;

DBGrid1.Columns[nSatuan].PickList.Add(taBrg.FieldByName('SATUAN').AsString
);

      { menghitung total dari detail }
      subtotal       := DBGetTotal(taJualdtlTOTAL);
      Panel5.Caption := format('%m ', [subtotal]);
```

```
      dsWatchDataChange(nil, nil);  { supaya respon ke Discount, PPN dan Grand Total
}
    end;
  except
  end;

  dsDetail.OnDataChange := dsDetailDataChange;
end;

procedure TfmJual.rgJenisBayarChange(Sender: TObject);
begin
  deTerm.Enabled  := rgJenisBayar.ItemIndex = 1;
  Label14.Enabled := rgJenisBayar.ItemIndex = 1;
  Label7.Enabled  := rgJenisBayar.ItemIndex = 1;

  if dsWatch.State in [dsEdit, dsInsert] then
    if rgJenisBayar.ItemIndex = 0 then dmPenjualan.taJualTERM.Value := 0;
end;

procedure TfmJual.Add1Click(Sender: TObject);
begin
  dmPenjualan.taJualdtl.Append;
end;

procedure TfmJual.Edit1Click(Sender: TObject);
begin
  dmPenjualan.taJualdtl.Edit;
end;

procedure TfmJual.Delete1Click(Sender: TObject);
begin
  if not dmPenjualan.taJualDtl.EOF then dmPenjualan.taJualdtl.Delete;
end;

procedure TfmJual.Simpan1Click(Sender: TObject);
begin
  if dsDetail.State in [dsEdit, dsInsert] then dmPenjualan.taJualdtl.Post;
end;

procedure TfmJual.Batal1Click(Sender: TObject);
begin
  dmPenjualan.taJualdtl.Cancel;
end;

procedure TfmJual.PopupMenu1Popup(Sender: TObject);
begin
  if dsWatch.State = dsBrowse then abort;
end;

procedure TfmJual.KNavPreviewClick(Sender: TObject);
```

```
begin
  with dmPenjualan do begin
    crpe1.Clear;
    crpe1.ReportName              := PathExe + 'Report\ntJual.rpt';
    crpe1.Output                  := toWindow;
    crpe1.DiscardSavedData        := true;
    crpe1.WindowZoom.Magnification := 100;
    crpe1.WindowStyle.Title       := fmJual.Caption;

    crpe1.Tables.Retrieve;
    crpe1.Tables[0].Path   := AllUnits.PathData;
    crpe1.Tables.Propagate := True;

    crpe1.ParamFields.Retrieve;
    crpe1.ParamFields[0].Value := Userlogin;
    crpe1.ParamFields[1].Value := taJualNOMER.Text;
    crpe1.WindowButtonBar.PrintBtn := False;
    crpe1.Execute;

    crpe1.Clear;
    crpe1.ReportName              := PathExe + 'Report\ntsj.rpt';
    crpe1.Output                  := toWindow;
    crpe1.DiscardSavedData        := true;
    crpe1.WindowZoom.Magnification := 100;
    crpe1.WindowStyle.Title       := 'Surat Jalan';

    crpe1.Tables.Retrieve;
    crpe1.Tables[0].Path   := AllUnits.PathData;
    crpe1.Tables.Propagate := True;

    crpe1.ParamFields.Retrieve;
    crpe1.ParamFields[0].Value := Userlogin;
    crpe1.ParamFields[1].Value := taJualNOMER.Text;

    crpe1.WindowButtonBar.PrintBtn := False;
    crpe1.Execute;
  end;
end;

procedure TfmJual.KNavSearchClick(Sender: TObject);
begin
  fmSearch.LoadSearch(dmPenjualan.taJual);
end;

procedure TfmJual.FormCloseQuery(Sender: TObject; var CanClose: Boolean);
begin
  CanClose := dsWatch.State=dsBrowse;
end;

Function TfmJual.HitungTotNota:Currency;
```

```
Var GrandTotNota : Currency;
Begin
  With dmPenjualan do
  begin
    taJualDtl.DisableControls;
    taJualDtl.First;
    GrandTotNota := 0;
    While Not taJualDtl.EOF do
    Begin
      // Sum GrandTotal
                GrandTotNota := GrandTotNota + (taJualDtlJUMLAH.Value *
taJualDtlHARGA.Value) - taJualDtlDISC.Value;

      // Update GdgBrg
      if taGdgBrg.FindKey([taJualdtlKODEBRG.Value]) then
      begin
        taGdgBrg.Edit;
    taGdgBrg.FieldByName('SALDO').AsFloat:=
taGdgBrg.FieldByName('SALDO').AsFloat -
                                    taJualdtlJUMLAH.Value;
      end;

      taGdgBrg.Post;
      taJualDtl.Next;
    End;
    taJualDtl.EnableControls;
  End;
  Result := GrandTotNota;
End;

procedure TfmJual.DBGrid1ColEnter(Sender: TObject);
begin
  oldKodeBrg:=dmPenjualan.taJualDtlKodebrg.value;
end;

procedure TfmJual.DBGrid1KeyPress(Sender: TObject; var Key: Char);
begin
  if DBGrid1.SelectedField.FieldName='KODEBRG' then
    Key:=UpCase(Key);
end;

procedure TfmJual.CheckBox1Enter(Sender: TObject);
begin
  PostMessage(Handle, WM_NEXTDLGCTL, 0, 0)
end;

end.
```

```
unit uLap;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  SplshWnd, Ucrpe32;

type
  TdmLap = class(TDataModule)
    SaveDialog1: TSaveDialog;
    Crpe1: TCrpe;
    procedure dmLapCreate(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
    fnamexport : string;

    procedure InitReport(vReportName : string; Drilldown : Boolean);
    function CrwDate(vDate : TDateTime) : string;
  end;

var
  dmLap: TdmLap;

implementation
uses Allunits;

{$R *.DFM}

{ Mengkonversikan tanggal Delphi menjadi tanggal Crystal Report }
function TdmLap.CrwDate(vDate : TDateTime) : string;
var tgl, bln, thn : word;
begin
  DecodeDate(vDate, thn, bln, tgl);
  CrwDate := 'DateTime(' + IntToStr(thn) + ',' + IntToStr(bln) + ',' + IntToStr(tgl) + ')';
end;

{ Inisialisasi engine Crystal Report }
procedure TdmLap.InitReport(vReportName : string; Drilldown : Boolean);
var fmSplash : TForm;
begin
   fmSplash := ShowSplashWindow(Application.Icon, 'Loading report engine... Please
wait.', False, Nil);

  with Crpe1 do begin
    CloseJob;
    ReportName    := PathExe + 'Report\' + vReportName;
    SendOnExecute := True;
```

```
    WindowState   := wsMaximized;

    with export do
    begin
      FileType      := ExcelXLS;
      Excel.XlsType := Excel5;
      Destination   := toFile;
    end;

       { button di atas report }
    with WindowButtonBar do begin
      AllowDrillDown := Drilldown;
      GroupTree      := Drilldown;

      SearchBtn      := True;
    end;

    { Database path di arahkan ke variabel Allunits.PathData}
    Tables.Retrieve;
    Tables[0].Path   := PathData;
    Tables.Propagate := true;
  end;

  fmSplash.Free;
end;

procedure TdmLap.dmLapCreate(Sender: TObject);
begin
  SaveDialog1.InitialDir := PathData;
end;

end.
```

unit uMenuUtama;

```
interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  Menus, ExtCtrls, ComCtrls, StdCtrls, Db, DBTables;

type
  TfmMenuUtama = class(TForm)
    MainMenu1: TMainMenu;
    Inventory1: TMenuItem;
    Pembelian1: TMenuItem;
    Penjualan1: TMenuItem;
    Hutang1: TMenuItem;
    Piutang1: TMenuItem;
```

Utility1: TMenuItem;
Login1: TMenuItem;
Logout1: TMenuItem;
N1: TMenuItem;
N2: TMenuItem;
Laporan1: TMenuItem;
SisaStokdigudang1: TMenuItem;
Supplier1: TMenuItem;
N3: TMenuItem;
Pembelian2: TMenuItem;
ReturPembelian1: TMenuItem;
N4: TMenuItem;
Laporan2: TMenuItem;
DaftarSupplier1: TMenuItem;
N5: TMenuItem;
PembelianperSupplier1: TMenuItem;
PembelianBersih1: TMenuItem;
ReturPembelianperSupplier1: TMenuItem;
Sales1: TMenuItem;
Customer1: TMenuItem;
N6: TMenuItem;
Penjualan2: TMenuItem;
ReturPenjualan1: TMenuItem;
N7: TMenuItem;
Laporan3: TMenuItem;
DaftarSupplier2: TMenuItem;
DaftarCustomer1: TMenuItem;
N8: TMenuItem;
Penjualan3: TMenuItem;
PenjualanperSales1: TMenuItem;
PenjualanBersih1: TMenuItem;
N9: TMenuItem;
ReturPenjualanperCustomer1: TMenuItem;
PenjualanperSales2: TMenuItem;
PembayaranHutang1: TMenuItem;
N10: TMenuItem;
Laporan4: TMenuItem;
UsiaHutang1: TMenuItem;
KartuHutang1: TMenuItem;
PembayaranHutang2: TMenuItem;
N11: TMenuItem;
Laporan5: TMenuItem;
PiutangJatuhTempo1: TMenuItem;
N12: TMenuItem;
SetAlias1: TMenuItem;
UbahPassword1: TMenuItem;
N13: TMenuItem;
About1: TMenuItem;
StatusBar1: TStatusBar;
Timer1: TTimer;

```
    Barang1: TMenuItem;
    DaftarBarang1: TMenuItem;
    LogoutdanKeluar1: TMenuItem;
    Panel1: TPanel;
    Image1: TImage;
    AdjustmentIn1: TMenuItem;
    AdjustmentOut1: TMenuItem;
    N14: TMenuItem;
    AdjustmentIn2: TMenuItem;
    AdjustmentOut2: TMenuItem;
    N15: TMenuItem;
    Service1: TMenuItem;
    TransaksiService1: TMenuItem;
    ProfilePerusahaan1: TMenuItem;
    procedure FormCreate(Sender: TObject);
    procedure Timer1Timer(Sender: TObject);
    procedure Login1Click(Sender: TObject);
    procedure Logout1Click(Sender: TObject);
    procedure Barang1Click(Sender: TObject);
    procedure LogoutdanKeluar1Click(Sender: TObject);
    procedure Supplier1Click(Sender: TObject);
    procedure Sales1Click(Sender: TObject);
    procedure Customer1Click(Sender: TObject);
    procedure UbahPassword1Click(Sender: TObject);
    procedure SetAlias1Click(Sender: TObject);
    procedure About1Click(Sender: TObject);
    procedure FormShow(Sender: TObject);
    procedure Pembelian2Click(Sender: TObject);
    procedure ReturPembelian1Click(Sender: TObject);
    procedure Penjualan2Click(Sender: TObject);
    procedure ReturPenjualan1Click(Sender: TObject);
    procedure PembayaranHutang1Click(Sender: TObject);
    procedure PembayaranHutang2Click(Sender: TObject);
    procedure AdjustmentIn1Click(Sender: TObject);
    procedure AdjustmentOut1Click(Sender: TObject);
    procedure Service1Click(Sender: TObject);
    procedure TransaksiService1Click(Sender: TObject);
    procedure ProfilePerusahaan1Click(Sender: TObject);
    procedure DaftarBarang1Click(Sender: TObject);
  private
  public
    SystemFile : TStringList;
    procedure InitMenu;
  end;

var
  fmMenuUtama: TfmMenuUtama;

implementation
```

```
uses uLogin, uBarang, uSupplier, uSales, uCust, uChgPass, uSetAlias,
     uAlias, uAbout, AllUnits, uBeli, uRtBl, UJual, uRtJl, UBayarHutang,
     UBayarPiutang, UAdin, UAdou, uService, uSrv, uProcom, uRepBarang;

{$R *.DFM}

procedure TfmMenuUtama.FormShow(Sender: TObject);
begin
  if FileExists(ExtractFilePath(ParamStr(0)) + 'Wallpaper.bmp') then
    Image1.Picture.LoadFromFile(ExtractFilePath(ParamStr(0)) + 'Wallpaper.bmp');


end;

procedure TfmMenuUtama.InitMenu;
begin
  MainMenu1.Items[0].Enabled := False;
  MainMenu1.Items[1].Enabled := False;
  MainMenu1.Items[2].Enabled := False;
  MainMenu1.Items[3].Enabled := False;
  MainMenu1.Items[4].Enabled := False;
  MainMenu1.Items[5].Enabled := True;
  MainMenu1.Items[5].Items[0].Enabled := True;
  MainMenu1.Items[5].Items[1].Enabled := False;
  MainMenu1.Items[5].Items[2].Enabled := False;
  MainMenu1.Items[5].Items[3].Enabled := False;
  MainMenu1.Items[5].Items[4].Enabled := False;
  MainMenu1.Items[5].Items[5].Enabled := True;
  MainMenu1.Items[5].Items[6].Enabled := False;
  MainMenu1.Items[5].Items[7].Enabled := False;
  MainMenu1.Items[5].Items[8].Enabled := True;
end;

procedure TfmMenuUtama.FormCreate(Sender: TObject);
begin
  InitMenu;
  StatusBar1.Panels[0].Text := DateToStr(Date);
end;

procedure TfmMenuUtama.Timer1Timer(Sender: TObject);
begin
  StatusBar1.Panels[1].Text := TimeToStr(Time);
end;

procedure TfmMenuUtama.Login1Click(Sender: TObject);
var fmLogin : TfmLogin;
begin
  SystemFile := TStringList.Create;
  if not FileExists(ExtractFilePath(ParamStr(0)) + 'System.dat') then
    DispErr('Direktori belum diset', fmMenuUtama);
```

```
  fmLogin := TfmLogin.Create(Self);
  fmLogin.ShowModal;

  SystemFile.LoadFromFile(ExtractFilePath(ParamStr(0)) + 'System.dat');

  if SystemFile.Values['ShowAbout'] = 'Yes' then
    fmAbout.Show;
end;

procedure TfmMenuUtama.Logout1Click(Sender: TObject);
begin
  InitMenu;
end;

procedure TfmMenuUtama.LogoutdanKeluar1Click(Sender: TObject);
begin
  Close;
end;

procedure TfmMenuUtama.Barang1Click(Sender: TObject);
var fmBarang : TfmBarang;
begin
  fmBarang := TfmBarang.Create(Self);
  fmBarang.ShowModal;
  fmBarang.Free;
end;

procedure TfmMenuUtama.Supplier1Click(Sender: TObject);
var fmSupplier : TfmSupplier;
begin
  fmSupplier := TfmSupplier.Create(Self);
  fmSupplier.ShowModal;
  fmSupplier.Free;
end;

procedure TfmMenuUtama.Sales1Click(Sender: TObject);
var fmSales : TfmSales;
begin
  fmSales := TfmSales.Create(Self);
  fmSales.ShowModal;
  fmSales.Free;
end;

procedure TfmMenuUtama.Customer1Click(Sender: TObject);
var fmCustomer : TfmCustomer;
begin
  fmCustomer := TfmCustomer.Create(Self);
  fmCustomer.ShowModal;
  fmCustomer.Free;
```

```
end;

procedure TfmMenuUtama.UbahPassword1Click(Sender: TObject);
var fmChangePassword : TfmChangePassword;
begin
  fmChangePassword := TfmChangePassword.Create(Self);
  fmChangePassword.ShowModal;
  fmChangePassword.Free;
end;

procedure TfmMenuUtama.SetAlias1Click(Sender: TObject);
var fmSetAlias : TfmSetAlias;
begin
  fmSetAlias := TfmSetAlias.Create(Self);
  fmSetAlias.ShowModal;
  fmSetAlias.Free;
end;

procedure TfmMenuUtama.About1Click(Sender: TObject);
begin
  fmAbout.Show;
end;

procedure TfmMenuUtama.Pembelian2Click(Sender: TObject);
begin
  fmBeli.ShowModal;
end;

procedure TfmMenuUtama.ReturPembelian1Click(Sender: TObject);
begin
  fmReturBeli.ShowModal;
end;

procedure TfmMenuUtama.Penjualan2Click(Sender: TObject);
begin
  fmJual.ShowModal;
end;

procedure TfmMenuUtama.ReturPenjualan1Click(Sender: TObject);
begin
  fmReturJual.ShowModal;
end;

procedure TfmMenuUtama.PembayaranHutang1Click(Sender: TObject);
begin
  fmBayarHutang.ShowModal;
end;

procedure TfmMenuUtama.PembayaranHutang2Click(Sender: TObject);
begin
```

```
  fmBayarPiutang.ShowModal;
end;

procedure TfmMenuUtama.AdjustmentIn1Click(Sender: TObject);
begin
  fmAdin.ShowModal;
end;

procedure TfmMenuUtama.AdjustmentOut1Click(Sender: TObject);
begin
  fmAdou.ShowModal;
end;

procedure TfmMenuUtama.Service1Click(Sender: TObject);
var fmService : TfmService;
begin
  fmService := TfmService.Create(Self);
  fmService.ShowModal;
  fmService.Free;
end;

procedure TfmMenuUtama.TransaksiService1Click(Sender: TObject);
begin
  fmSrv.ShowModal;
end;

procedure TfmMenuUtama.ProfilePerusahaan1Click(Sender: TObject);
var fmProcom : TfmProcom;
begin
  fmProcom := TfmProcom.Create(Self);
  fmProcom.ShowModal;
  fmProcom.Free;
end;

procedure TfmMenuUtama.DaftarBarang1Click(Sender: TObject);
var fmRepBarang : TfmRepBarang;
begin
  fmRepBarang := TfmRepBarang.Create(Self);
  fmRepBarang.ShowModal;
  fmRepBarang.Free;
end;

end.
```

```
unit uRepBarang;
interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  ExtCtrls, StdCtrls, Mask, ToolEdit, uCrpe32;

type
  TfmRepBarang = class(TForm)
    vKodeAwal: TComboEdit;
    Label1: TLabel;
    Label2: TLabel;
    Button1: TButton;
    Button2: TButton;
    Bevel1: TBevel;
    vKodeAkhir: TComboEdit;
    procedure Button2Click(Sender: TObject);
    procedure Button1Click(Sender: TObject);
    procedure vKodeAwalButtonClick(Sender: TObject);
    procedure vKodeAkhirButtonClick(Sender: TObject);
  private
  public
    procedure CheckEntry;
    procedure BuildReport;
  end;

var
  fmRepBarang: TfmRepBarang;

implementation

uses AllUnits, uLap, uBrowse;

{$R *.DFM}

procedure TfmRepBarang.CheckEntry;
begin
  if vKodeAwal.Text > vKodeAkhir.Text then
    disperr('Kode Barang Awal harus <= Kode Barang Akhir', vKodeAwal);
end;

procedure TfmRepBarang.buildreport;
var str : string;
begin
  str:='';

  if vKodeAwal.Text <> '' then
  begin
    if str <> '' then str := str + ' and ';
    str:=str + '{TBARANG.Kode}' + ' >= "' + vKodeAwal.Text + '"';
```

```
  end;

  if vKodeAkhir.Text<>'' then
  begin
    if str <> '' then str := str + ' and ';
    str := str + '{TBARANG.Kode}' + ' <= "' + vKodeAkhir.Text+'"';
  end;

  dmLap.InitReport('MastBarang.RPT', false);

  with dmLap.Crpe1 do
  begin
    Selection.Clear;
    Selection.Replace := True;
    Selection.Formula.Add(str);

    ParamFields.Retrieve;

    ParamFields[0].Value := vKodeAwal.Text+ ' - '+vKodeAkhir.Text
  end;
end;

procedure TfmRepBarang.Button2Click(Sender: TObject);
begin
  Close;
end;

procedure TfmRepBarang.Button1Click(Sender: TObject);
begin
  CheckEntry;
  BuildReport;
  dmLap.Crpe1.Output := toWindow;
  dmLap.Crpe1.Execute;
  dmLap.Crpe1.CloseJob;
end;

procedure TfmRepBarang.vKodeAwalButtonClick(Sender: TObject);
begin
  fmBrowse.brwTBARANG(vKodeAwal.Text, nil);
  if fmBrowse.ModalResult = mrOk then
    vKodeAwal.Text:=fmBrowse.StringGrid1.Cells[1,0];
end;

procedure TfmRepBarang.vKodeAkhirButtonClick(Sender: TObject);
begin
  fmBrowse.brwTBARANG(vKodeAkhir.Text, nil);
  if fmBrowse.ModalResult = mrOk then
    vKodeAkhir.Text:=fmBrowse.StringGrid1.Cells[1,0];
end;
end.
```

```
unit uRtBl;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  Menus, Grids, DBGrids, ComCtrls, ZKNavigator, ExtCtrls, StdCtrls,
  DBCtrls, Mask, Db, DBTables, MRUList, ToolEdit, RXDBCtrl, RxLookup,
  CurrEdit, Buttons, uCrpe32;

type
  TfmReturBeli = class(TForm)
    Bevel3: TBevel;
    Panel4: TPanel;
    Label3: TLabel;
    Label4: TLabel;
    Label5: TLabel;
    Label6: TLabel;
    Label8: TLabel;
    deKodeSupl: TRxDBComboEdit;
    Panel3: TPanel;
    DBEdit2: TDBEdit;
    DBEdit3: TDBEdit;
    DBEdit4: TDBEdit;
    DBEdit5: TDBEdit;
    Panel6: TPanel;
    Label10: TLabel;
    Label11: TLabel;
    Label12: TLabel;
    Label13: TLabel;
    deDisc: TDBEdit;
    dePpn: TDBEdit;
    ceDisc: TCurrencyEdit;
    cePpn: TCurrencyEdit;
    Panel5: TPanel;
    Panel9: TPanel;
    Panel2: TPanel;
    Label14: TLabel;
    Label30: TLabel;
    Label7: TLabel;
    deTerm: TDBEdit;
    DBMemo1: TDBMemo;
    rgJenisBayar: TDBRadioGroup;
    DBGrid1: TRxDBGrid;
    KNav: TK2Navigator;
    DBCheckBox1: TDBCheckBox;
    CheckBox1: TCheckBox;
    Panel7: TPanel;
    Label22: TLabel;
    Label23: TLabel;
```

```
    Panel8: TPanel;
    DBText1: TDBText;
    deTanggal: TDBDateEdit;
    dsWatch: TDataSource;
    dsDetail: TDataSource;
    PopupMenu1: TPopupMenu;
    Add1: TMenuItem;
    Edit1: TMenuItem;
    Delete1: TMenuItem;
    Simpan1: TMenuItem;
    Batal1: TMenuItem;
    procedure FormClose(Sender: TObject; var Action: TCloseAction);
    procedure FormShow(Sender: TObject);
    procedure dsWatchStateChange(Sender: TObject);
    procedure KNavCancelClick(Sender: TObject);
    procedure KNavAddClick(Sender: TObject);
    procedure deKodeSuplButtonClick(Sender: TObject);
    procedure KNavSaveClick(Sender: TObject);
    procedure ceDiscExit(Sender: TObject);
    procedure cePpnExit(Sender: TObject);
    procedure dsWatchDataChange(Sender: TObject; Field: TField);
    procedure DBGrid1EditButtonClick(Sender: TObject);
    procedure dsDetailDataChange(Sender: TObject; Field: TField);
    procedure rgJenisBayarChange(Sender: TObject);
    procedure Add1Click(Sender: TObject);
    procedure Edit1Click(Sender: TObject);
    procedure Delete1Click(Sender: TObject);
    procedure Simpan1Click(Sender: TObject);
    procedure Batal1Click(Sender: TObject);
    procedure PopupMenu1Popup(Sender: TObject);
    procedure KNavPreviewClick(Sender: TObject);
    procedure KNavSearchClick(Sender: TObject);
    procedure FormCloseQuery(Sender: TObject; var CanClose: Boolean);
    procedure DBGrid1ColEnter(Sender: TObject);
    procedure DBGrid1KeyPress(Sender: TObject; var Key: Char);
    procedure CheckBox1Enter(Sender: TObject);
  private
    taBrg, taSupl, taGdgBrg : TTable;
    subtotal, grandtotal    : real;
    oldkodebrg              :  string;

    procedure SettingEdit(nVal : boolean);
    procedure Cancel_All_Updates;

    procedure SetEvent;
    procedure ClearEvent;

    procedure taRtBlKODESUPLChange(Sender: TField);
    procedure taRtBlPPNChange(Sender: TField);
    procedure taRtBlDISCChange(Sender: TField);
```

```
    procedure taRtBldtlKODEBRGChange(Sender: TField);

    procedure taRtBlDtlCalcFields(DataSet: TDataSet);
    procedure taRtBlDtlNewRecord(DataSet: TDataSet);
    procedure taRtBldtlBeforeDelete(DataSet: TDataSet);
    procedure AfterScroll(DataSet : TDataSet);

    function HitungTotNota:Currency;
  public
  end;

var
  fmReturBeli: TfmReturBeli;

implementation

uses AllUnits, UDmPembelian, UBrowse, USearch;

{$R *.DFM}

procedure TfmReturBeli.AfterScroll(DataSet : TDataSet);
begin
  with dmPembelian Do
  begin
    if taRtBlTOTNOTA.Value = 0 Then
      ceDisc.Value := 0
    else ceDisc.Value := taRtBlDISC.Value * 100 / taRtBlTOTNOTA.Value;

    if taRtBlTOTNOTA.Value = 0 Then
      cePpn.Value  := 0
    else cePpn.Value  := taRtBlPPN.Value * 100 / taRtBlTOTNOTA.Value;

    subtotal       := DBGetTotal(taRtBldtlTOTAL);
    Panel5.Caption := format('%m ', [subtotal]);
    grandtotal     := subtotal - taRtBlDISC.Value + taRtBlPPN.Value;
    Panel9.Caption := format('%m ', [grandtotal]);
  end;
End;

procedure TfmReturBeli.SetEvent;
begin
  with dmPembelian do
  begin
    taRtBlKODESUPL.OnChange   := taRtBlKODESUPLChange;
    taRtBlPPN.OnChange        := taRtBlPPNChange;
    taRtBlDISC.OnChange       := taRtBlDISCChange;
    taRtBldtlKODEBRG.OnChange := taRtBldtlKODEBRGChange;
    taRtBldtl.OnNewRecord     := taRtBlDtlNewRecord;
    taRtBldtl.BeforeDelete    := taRtBldtlBeforeDelete;
    taRtBlDtl.OnCalcFields    := taRtBlDtlCalcFields;
```

```
  end;
end;

procedure TfmReturBeli.ClearEvent;
begin
  with dmPembelian do
  begin
    taRtBlKODESUPL.OnChange   := nil;
    taRtBlPPN.OnChange        := nil;
    taRtBlDISC.OnChange       := nil;
    taRtBldtlKODEBRG.OnChange := nil;
    taRtBldtl.BeforeInsert    := nil;
    taRtBldtl.OnNewRecord     := nil;
    taRtBldtl.BeforeDelete    := nil;
    taRtBlDtl.OnCalcFields    := nil;
  end;
end;

procedure TfmReturBeli.Cancel_All_Updates;
begin
  with dmPembelian do
  begin
    taRtBlDtl.CancelUpdates;
    taRtBl.Cancel;
  end;
end;

procedure TfmReturBeli.SettingEdit(nVal : boolean);
var n : integer;
begin
  for n := 0 to fmReturBeli.ComponentCount - 1 do
if fmReturBeli.Components[n].Tag = 100 then
TControl(fmReturBeli.Components[n]).Enabled := nVal;
end;

procedure TfmReturBeli.taRtBlKODESUPLChange(Sender: TField);
begin
  with dmPembelian do
  begin
    taRtBlKODESUPL.OnChange := nil;

    if not taSupl.findkey([Sender.asString]) then deKodeSuplButtonClick(nil);

    taRtBlKODESUPL.Value    := taSupl.FieldByName('KODE').asString;
    taRtBlNAMASUPL.Value    := taSupl.FieldByName('NAMA').asString;
    taRtBlALAMAT.Value      := taSupl.FieldByName('ALAMAT').asString;
    taRtBlKOTA.Value        := taSupl.FieldByName('KOTA').asString;
    taRtBlNEGARA.Value      := taSupl.FieldByName('NEGARA').asString;

    { jika Nama, Alamat, Kota, Negara ada isinya maka cuma readonly }
```

```
    DBEdit2.ReadOnly := not taSupl.FieldByName('NAMA').isNull;
    DBEdit3.ReadOnly := not taSupl.FieldByName('ALAMAT').isNull;
    DBEdit4.ReadOnly := not taSupl.FieldByName('KOTA').isNull;
    DBEdit5.ReadOnly := not taSupl.FieldByName('NEGARA').isNull;

    taRtBlKODESUPL.OnChange := taRtBlKODESUPLChange;
  end;
end;

procedure TfmReturBeli.taRtBlPPNChange(Sender: TField);
begin
  if subtotal > 0 then cePpn.Value := sender.asCurrency * 100 / subtotal;
end;

procedure TfmReturBeli.taRtBlDISCChange(Sender: TField);
begin
  if subtotal > 0 then
    ceDisc.Value := sender.asCurrency * 100 / subtotal;
end;

procedure TfmReturBeli.taRtBldtlKODEBRGChange(Sender: TField);
begin
  with dmPembelian do
  begin
    dsDetail.OnDataChange     := nil;

    if not taBrg.FindKey([sender.asString]) then DBGrid1EditButtonClick(nil);

    taRtBldtlKODEBRG.OnChange := nil;

    taRtBldtl.Edit;
    taRtBldtlKODEBRG.Value    := taBrg.FieldByName('KODE').asString;
    taRtBldtlNAMABRG.Value    := taBrg.FieldByName('NAMA').asString;
    taRtBldtlSATUAN.Value     := taBrg.FieldByName('SATUAN').asString;
    taRtBldtlJUMLAH.Value     := 1;
    taRtBldtlHARGA.Value      := 0;
    taRtBldtlDISC.Value       := 0;

    taRtBldtl.Post;
    taRtBldtlKODEBRG.OnChange := taRtBldtlKODEBRGChange;
    dsDetail.OnDataChange     := dsDetailDataChange;
  end;
end;

procedure TfmReturBeli.taRtBlDtlNewRecord;
begin
  DBInitValue(dmPembelian.taRtBlDtl);
  DBgrid1.SelectedIndex := 0;
end;
```

```
procedure TfmReturBeli.taRtBlDtlCalcFields;
begin
  with dmPembelian do
     taRtBldtlTOTAL.Value := (taRtBlDtlJUMLAH.Value * taRtBlDtlHARGA.Value)
- taRtBlDtlDISC.Value;
end;

procedure TfmReturBeli.taRtBlDtlBeforeDelete(DataSet: TDataSet);
begin
  if dsWatch.State <> dsBrowse then
     if MessageDlg('Hapus record ... ?',mtConfirmation,[mbYes, mbNo], 0) <> mrYes
then abort;
end;

procedure TfmReturBeli.FormShow;
begin
  with dmPembelian do
  begin
    DBSetPath(dmPembelian);
    DBLoadLabel(taRtBldtl);

    taGdgBrg   := OpenTable('TGDGBRG' , '', fmReturBeli);
    taSupl     := OpenTable('TSUPL'   , '', fmReturBeli);
    taBrg      := OpenTable('TBARANG' , '', fmReturBeli);

    taRtBldtl.Open;
    taRtBl.Open;

    SetEvent;
    taRtBl.AfterScroll := AfterScroll;
    AfterScroll(taRtBl);
  end;
  deTanggal.SetFocus;
end;

procedure TfmReturBeli.FormClose;
begin
  with dmPembelian do
  begin
    taSupl.Close;
    taBrg.Close;
    taGdgBrg.Close;
    taRtBldtl.Close;
    taRtBl.Close;
    taRtBl.AfterScroll := NIL;
  end;
end;

procedure TfmReturBeli.KNavCancelClick;
begin
```

```
  KNav.SetFocus;
  with dmPembelian do begin
    case dsWatch.State of
      dsInsert : begin
                   taRtBlNAMAUSER.Value := UserLogin;
                   taRtBlVOID.Value     := true;
                   taRtBlDISC.Value     := 0;
                   taRtBlPPN.Value      := 0;
                   ceDisc.Value         := 0;
                   cePpn.Value          := 0;
                   taRtBl.Post;
                   taRtBldtl.CommitUpdates;
                 end;
    end;
    Cancel_All_Updates;
    deTanggal.SetFocus;
  end;
end;


procedure TfmReturBeli.KNavAddClick;
begin
  ClearEvent;
  with dmPembelian do
  begin
    taRtBl.Append;
    DBInitValue(taRtBl);
    taRtBlNOMER.Value    := DBGetCounter('RB', fmReturBeli);
    taRtBlJENIS.Value    := 0;
    taRtBlTERM.Value     := 0;
  end;

  subtotal                := 0;
  deKodeSupl.Button.Enabled := true;
  deTanggal.SetFocus;
  SetEvent;
end;

procedure TfmReturBeli.KNavSaveClick;
begin
  with dmPembelian do begin

    KNav.SetFocus;
    dsDetail.OnDataChange := nil;  { supaya jangan rekursif }

     taRtBl.DisableControls;         { supaya semua proses di dlm save tidak berpengaruh
pada control }
    taRtBldtl.DisableControls;
    try
```

```
      if taRtBlTANGGAL.IsNull                    then DispErr('Tanggal tidak boleh
kosong', deTanggal);
      if taRtBlTANGGAL.Value <> Date             then DispErr('Tanggal harus =
tanggal sekarang', deTanggal);

      if taRtBlKODESUPL.IsNull                   then DispErr('Kode Supplier tidak
boleh kosong', deKodeSupl);
      if not taSupl.FindKey([taRtBlKODESUPL.Value])  then DispErr('Kode Supplier
tidak ditemukan', deKodeSupl);

       if ((taRtBlTERM.Value < 0) and (taRtBlJENIS.Value = 1)) then disperr('Term
harus > 0',deTerm);

     { *** check detail *** }
     taRtBldtl.First;
     while not taRtBldtl.EOF do
     begin
      if taRtBldtlKODEBRG.IsNull                 then DispErr('Kode Barang tidak
boleh kosong', DBGrid1);
       if not taBrg.FindKey([taRtBldtlKODEBRG.Value]) then DispErr('Kode Barang
tidak ditemukan', DBGrid1);

      if not taGdgBrg.FindKey([taRtBldtlKODEBRG.Value]) then
       DispErr('Barang ' + taRtBldtlKODEBRG.Value + ' tidak ditemukan di gudang ',
DBGrid1);

       if taGdgBrg.FieldByName('SALDO').AsInteger - taRtBlDtlJUMLAH.Value < 0
then
       DispErr('Stok barang ' + taRtBldtlKODEBRG.Value + ' tidak mencukupi (' +
            FloatToStr(taGdgBrg.FieldByName('SALDO').AsInteger -
taRtBlDtlJUMLAH.Value) + ')', DBGrid1);

        if taRtBldtlJUMLAH.IsNull                 then DispErr('Jumlah tidak boleh
kosong', DBGrid1);
      if taRtBldtlJUMLAH.Value < 0               then DispErr('Jumlah tidak boleh <
0', DBGrid1);

        if taRtBldtlSATUAN.IsNull                 then DispErr('Satuan tidak boleh
kosong', DBGrid1);
        if taRtBlDtlHARGA.IsNull                  then DispErr('Harga tidak boleh
kosong', DBGrid1);
      if taRtBlDtlHARGA.Value < 0                then DispErr('Harga tidak boleh < 0',
DBGrid1);

      taRtBldtl.Next;
     end;

     if not taHut.Active then taHut.Open;

     if not taHut.FindKey([taRtBlNOMER.Value]) then
```

```
      taHut.Append
    else taHut.Edit;

    taHut.Fieldbyname('NOMER').asString        := taRtBlNOMER.Value;
    taHut.Fieldbyname('TANGGAL').asDatetime    := taRtBlTANGGAL.Value;
    taHut.Fieldbyname('KODESUPL').asString     := taRtBlKODESUPL.Value;
    taHut.Fieldbyname('NAMASUPL').asString     := taRtBlNAMASUPL.Value;
    taHut.Fieldbyname('NOREFF').asString       := '';
    taHut.Fieldbyname('IDREFF').asString       := '';
    taHut.Fieldbyname('TERM').asInteger        := taRtBlTERM.Value;
    taHut.Fieldbyname('DEBET').asCurrency      := 0;
    taHut.Fieldbyname('KREDIT').asCurrency     := GrandTotal;

    if taRtBlJENIS.Value = 0 then
    begin
      taHut.Fieldbyname('KETERANGAN').asString := 'Retur Pembelian Tunai';
      taHut.Fieldbyname('ID').asString         := 'RBTN';
      taHut.Fieldbyname('BAYAR').asCurrency    := GrandTotal;
    end else
    begin
      taHut.Fieldbyname('KETERANGAN').asString := 'Retur Pembelian Kredit';
      taHut.Fieldbyname('ID').asString         := 'RBKR';
      taHut.Fieldbyname('BAYAR').asCurrency    := 0;
    end;

    taHut.Post;

    { force setting }
    taRtBlNAMAUSER.Value := UserLogin;
    taRtBl.Post;
    taRtBldtl.CommitUpdates;

    taRtBl.Edit;
    taRtBlTOTNOTA.Value  := HitungTotNota;
    DBInitValue2(taRtBl);
    taRtBl.Post;

    taRtBldtl.First;
    deTanggal.SetFocus;

   finally
    dsDetail.OnDataChange := dsDetailDataChange; { supaya jangan rekursif }
    taRtBl.EnableControls;
    taRtBldtl.EnableControls;
   end;
  end;
end;

procedure TfmReturBeli.DBGrid1EditButtonClick(Sender: TObject);
begin
```

```
  with dmPembelian do
  begin
    oldkodebrg:=taRtBlDtlKODEBRG.Value ;

    fmBrowse.brwTBARANG(dbgrid1.InplaceEditor.Text, nil);
    taRtBldtl.Edit;
    taRtBldtlKODEBRG.Value := fmBrowse.StringGrid1.Cells[1, 0];
  end;
end;

procedure TfmReturBeli.deKodeSuplButtonClick;
begin
  fmBrowse.brwTSUPL(deKodeSupl.Text, nil);
  taSupl.FindKey([fmBrowse.StringGrid1.Cells[1, 0]]);
  dmPembelian.taRtBlKODESUPL.Value := fmBrowse.StringGrid1.Cells[1, 0];
end;

procedure TfmReturBeli.cePpnExit(Sender: TObject);
begin
  if dsWatch.State in [dsEdit, dsInsert] then
    dmPembelian.taRtBlPPN.Value := (subtotal * cePpn.Value) / 100;
end;

procedure TfmReturBeli.ceDiscExit(Sender: TObject);
begin
  if dsWatch.State in [dsEdit, dsInsert] then
    dmPembelian.taRtBlDISC.Value := (subtotal * ceDisc.Value) / 100;
end;

procedure TfmReturBeli.dsWatchStateChange;
begin
  try
    if dsWatch.State = dsBrowse then
      SettingEdit(True);
  except
  end;

  deKodeSupl.Button.Enabled := dsWatch.State in [dsEdit, dsInsert];

  ceDisc.ReadOnly          := dsWatch.State = dsBrowse;
  cePpn.ReadOnly           := dsWatch.State = dsBrowse;

  if dsWatch.State = dsBrowse then
    DBGrid1.Options := DBGrid1.Options - [dgEditing]
  else
    DBGrid1.Options := DBGrid1.Options + [dgEditing]
end;

procedure TfmReturBeli.dsWatchDataChange(Sender: TObject; Field: TField);
begin
```

```
  dsWatch.OnDataChange := nil;

  try
    with dmPembelian do
    begin
      deDisc.ReadOnly   := subtotal = 0;
      dePpn.ReadOnly    := subtotal = 0;
      CheckBox1.Checked := taRtBlPRINTED.Value > 0;

      if (subtotal > 0) and (field = nil) and (dsWatch.State <> dsBrowse) then begin
        taRtBlDISC.Value := subtotal * ceDISC.Value / 100;
        taRtBlPPN.Value  := subtotal * cePPN.Value / 100;
      end;

      grandtotal     := subtotal - taRtBlDISC.Value + taRtBlPPN.Value;
      Panel9.Caption := format('%m ', [grandtotal]);
    end;
  except
  end;

  dsWatch.OnDataChange := dsWatchDataChange;
end;

procedure TfmReturBeli.dsDetailDataChange(Sender: TObject; Field: TField);
var nSatuan : word;
begin
  dsDetail.OnDataChange := nil;  { supaya jangan rekursif }

  try
    with dmPembelian do
    begin
      { cari posisi kolom untuk field satuan, ini penting
        soalnya kolom dapat dipindah-pindah }
        for nSatuan := 0 to DBgrid1.FieldCount - 1 do if DBgrid1.Fields[nSatuan] =
taRtBldtlSATUAN then break;
      DBGrid1.Columns[nSatuan].PickList.Clear;

DBGrid1.Columns[nSatuan].PickList.Add(taBrg.FieldByName('SATUAN').AsString
);

      { menghitung total dari detail }
      subtotal       := DBGetTotal(taRtBldtlTOTAL);
      Panel5.Caption := format('%m ', [subtotal]);
      dsWatchDataChange(nil, nil);  { supaya respon ke Discount, PPN dan Grand Total
}
    end;
  except
  end;

  dsDetail.OnDataChange := dsDetailDataChange;
```

```
end;

procedure TfmReturBeli.rgJenisBayarChange(Sender: TObject);
begin
  deTerm.Enabled  := rgJenisBayar.ItemIndex = 1;
  Label14.Enabled := rgJenisBayar.ItemIndex = 1;
  Label7.Enabled  := rgJenisBayar.ItemIndex = 1;

  if dsWatch.State in [dsEdit, dsInsert] then
    if rgJenisBayar.ItemIndex = 0 then dmPembelian.taRtBlTERM.Value := 0;
end;

procedure TfmReturBeli.Add1Click(Sender: TObject);
begin
  dmPembelian.taRtBldtl.Append;
end;

procedure TfmReturBeli.Edit1Click(Sender: TObject);
begin
  dmPembelian.taRtBldtl.Edit;
end;

procedure TfmReturBeli.Delete1Click(Sender: TObject);
begin
  if not dmPembelian.taRtBldtl.EOF then dmPembelian.taRtBldtl.Delete;
end;

procedure TfmReturBeli.Simpan1Click(Sender: TObject);
begin
  if dsDetail.State in [dsEdit, dsInsert] then dmPembelian.taRtBldtl.Post;
end;

procedure TfmReturBeli.Batal1Click(Sender: TObject);
begin
  dmPembelian.taRtBldtl.Cancel;
end;

procedure TfmReturBeli.PopupMenu1Popup(Sender: TObject);
begin
  if dsWatch.State = dsBrowse then abort;
end;

procedure TfmReturBeli.KNavPreviewClick(Sender: TObject);
begin
  with dmPembelian do begin
    crpe1.Clear;
    crpe1.ReportName              := PathExe + 'Report\ntrtbl.rpt';
    crpe1.Output                  := toWindow;
    crpe1.DiscardSavedData        := true;
    crpe1.WindowZoom.Magnification := 100;
```

```
    crpe1.WindowStyle.Title        := fmReturBeli.Caption;

    crpe1.Tables.Retrieve;
    crpe1.Tables[0].Path   := PathData;
    crpe1.Tables.Propagate := True;

    crpe1.ParamFields.Retrieve;
    crpe1.ParamFields[0].Value := Userlogin;
    crpe1.ParamFields[1].Value := taRtBlNOMER.Text;
    crpe1.WindowButtonBar.PrintBtn := False;
    crpe1.Execute;

    crpe1.Clear;
    crpe1.ReportName               := PathExe + 'Report\ntrtblgdg.rpt';
    crpe1.Output                   := toWindow;
    crpe1.DiscardSavedData         := true;
    crpe1.WindowZoom.Magnification := 100;
    crpe1.WindowStyle.Title        := fmReturBeli.Caption + ' (Gudang)';

    crpe1.Tables.Retrieve;
    crpe1.Tables[0].Path   := PathData;
    crpe1.Tables.Propagate := True;

    crpe1.ParamFields.Retrieve;
    crpe1.ParamFields[0].Value := Userlogin;
    crpe1.ParamFields[1].Value := taRtBlNOMER.Text;
    crpe1.WindowButtonBar.PrintBtn := False;
    crpe1.Execute;
  end;
end;

procedure TfmReturBeli.KNavSearchClick(Sender: TObject);
begin
  fmSearch.LoadSearch(dmPembelian.taRtBl);
end;

procedure TfmReturBeli.FormCloseQuery(Sender: TObject; var CanClose: Boolean);
begin
  CanClose := dsWatch.State=dsBrowse;
end;

Function TfmReturBeli.HitungTotNota:Currency;
Var GrandTotNota : Currency;
Begin
   With dmPembelian do
   begin
      taRtBlDtl.DisableControls;
      taRtBlDtl.First;
      GrandTotNota := 0;
      While Not taRtBlDtl.EOF do
```

```
    Begin
      // Sum GrandTotal
                GrandTotNota := GrandTotNota + (taRtBlDtlJUMLAH.Value *
taRtBlDtlHARGA.Value) - taRtBlDtlDISC.Value;

      // Update GdgBrg
      taGdgBrg.FindKey([taRtBldtlKODEBRG.Value]);
      taGdgBrg.Edit;
   taGdgBrg.FieldByName('SALDO').AsFloat :=
taGdgBrg.FieldByName('SALDO').AsFloat -
                                  taRt BldtlJUMLAH.Value;
      taGdgBrg.Post;
      taRtBlDtl.Next;
    End;
    taRtBlDtl.EnableControls;
  End;
  Result := GrandTotNota;
End;

procedure TfmReturBeli.DBGrid1ColEnter(Sender: TObject);
begin
 oldKodeBrg:=dmPembelian.taRtBlDtlKodebrg.value;
end;

procedure TfmReturBeli.DBGrid1KeyPress(Sender: TObject; var Key: Char);
begin
 if DBGrid1.SelectedField.FieldName='KODEBRG' then
   Key:=UpCase(Key);
end;

procedure TfmReturBeli.CheckBox1Enter(Sender: TObject);
begin
 PostMessage(Handle, WM_NEXTDLGCTL, 0, 0)
end;

end.
```

unit uRtJl;

```
interface

uses
 Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
 Menus, Grids, DBGrids, ComCtrls, ZKNavigator, ExtCtrls, StdCtrls,
 DBCtrls, Mask, Db, DBTables, MRUList, ToolEdit, RXDBCtrl, RxLookup,
 CurrEdit, Buttons, uCrpe32, RXCtrls;

type
 TfmReturJual = class(TForm)
```

Bevel3: TBevel;
Panel4: TPanel;
Label3: TLabel;
Label4: TLabel;
Label5: TLabel;
Label6: TLabel;
Label8: TLabel;
deKodeCust: TRxDBComboEdit;
Panel3: TPanel;
DBEdit2: TDBEdit;
DBEdit3: TDBEdit;
DBEdit4: TDBEdit;
DBEdit5: TDBEdit;
Panel6: TPanel;
Label10: TLabel;
Label12: TLabel;
Label13: TLabel;
Label11: TLabel;
dePpn: TDBEdit;
cePpn: TCurrencyEdit;
Panel5: TPanel;
Panel9: TPanel;
ceDisc: TCurrencyEdit;
deDisc: TDBEdit;
Panel2: TPanel;
Label14: TLabel;
Label30: TLabel;
Label7: TLabel;
deTerm: TDBEdit;
DBMemo1: TDBMemo;
rgJenisBayar: TDBRadioGroup;
DBGrid1: TRxDBGrid;
Panel7: TPanel;
Label22: TLabel;
Label23: TLabel;
Panel8: TPanel;
DBText1: TDBText;
deTanggal: TDBDateEdit;
KNav: TK2Navigator;
DBCheckBox1: TDBCheckBox;
CheckBox1: TCheckBox;
dsWatch: TDataSource;
dsDetail: TDataSource;
PopupMenu1: TPopupMenu;
Add1: TMenuItem;
Edit1: TMenuItem;
Delete1: TMenuItem;
Simpan1: TMenuItem;
Batal1: TMenuItem;
procedure deKodeCustButtonClick(Sender: TObject);

```
  procedure DBGrid1EditButtonClick(Sender: TObject);
  procedure dsDetailDataChange(Sender: TObject; Field: TField);
  procedure dsWatchDataChange(Sender: TObject; Field: TField);
  procedure dsWatchStateChange(Sender: TObject);
  procedure FormClose(Sender: TObject; var Action: TCloseAction);
  procedure FormCloseQuery(Sender: TObject; var CanClose: Boolean);
  procedure FormShow(Sender: TObject);
  procedure KNavAddClick(Sender: TObject);
  procedure KNavCancelClick(Sender: TObject);
  procedure KNavPreviewClick(Sender: TObject);
  procedure KNavSaveClick(Sender: TObject);
  procedure ceDiscExit(Sender: TObject);
  procedure cePpnExit(Sender: TObject);
  procedure rgJenisBayarChange(Sender: TObject);
  procedure Add1Click(Sender: TObject);
  procedure Edit1Click(Sender: TObject);
  procedure Delete1Click(Sender: TObject);
  procedure Simpan1Click(Sender: TObject);
  procedure Batal1Click(Sender: TObject);
  procedure PopupMenu1Popup(Sender: TObject);
  procedure KNavSearchClick(Sender: TObject);
  procedure DBGrid1ColEnter(Sender: TObject);
  procedure DBGrid1KeyPress(Sender: TObject; var Key: Char);
  procedure CheckBox1Enter(Sender: TObject);
private
  taBrg, taCust, taGdgBrg : TTable;
  subtotal, grandtotal    : real;
  oldkodebrg              : string;

  procedure SettingEdit(nVal : boolean);
  procedure Cancel_All_Updates;

  procedure SetEvent ;
  procedure ClearEvent;

  procedure taRtJlKODECUSTChange(Sender: TField);

  procedure taRtJlPPNChange(Sender: TField);
  procedure taRtJlDISCChange(Sender: TField);

  procedure taRtJldtlKODEBRGChange(Sender: TField);

  procedure taRtJlDtlCalcFields(DataSet: TDataSet);
  procedure taRtJlDtlNewRecord(DataSet: TDataSet);
  procedure taRtJlDtlBeforeDelete(DataSet: TDataSet);

  procedure AfterScroll(DataSet : TDataSet);

  Function HitungTotNota:Currency;
public
```

```
  end;

var
  fmReturJual: TfmReturJual;

implementation

uses AllUnits, UDmPenjualan, UBrowse, uSearch;

{$R *.DFM}

Procedure TfmReturJual.AfterScroll(DataSet : TDataSet);
Begin
  with dmPenjualan do
  begin
    if taRtJlTOTNOTA.Value = 0 Then
      cePpn.Value  := 0
    else cePpn.Value  := taRtJlPPN.Value * 100 / taRtJlTOTNOTA.Value;

    if taRtJlTOTNOTA.Value = 0 Then
      cePpn.Value  := 0
    else cePpn.Value  := taRtJlPPN.Value * 100 / taRtJlTOTNOTA.Value;

    subtotal       := DBGetTotal(taRtJldtlTOTAL);
    Panel5.Caption := format('%m ', [subtotal]);
    grandtotal     := subtotal - taRtJlDISC.Value + taRtJlPPN.Value;
    Panel9.Caption := format('%m ', [grandtotal]);
  end;
end;

procedure TfmReturJual.SetEvent;
begin
  with dmPenjualan do
  begin
    taRtJlKODECUST.OnChange   := taRtJlKODECUSTChange;
    taRtJlPPN.OnChange        := taRtJlPPNChange;
    taRtJlDISC.OnChange       := taRtJlDISCChange;
    taRtJlDtlKODEBRG.OnChange := taRtJldtlKODEBRGChange;
    taRtJldtl.OnNewRecord     := taRtJlDtlNewRecord;
    taRtJldtl.BeforeDelete    := taRtJlDtlBeforeDelete;
    taRtJlDtl.OnCalcFields    := taRtJlDtlCalcFields;
  end;
end;

procedure TfmReturJual.ClearEvent;
begin
  with dmPenjualan do
  begin
    taRtJlKODECUST.OnChange   := nil;
    taRtJlPPN.OnChange        := nil;
```

```
    taRtJlDISC.OnChange       := nil;
    taRtJlDtlKODEBRG.OnChange := nil;
    taRtJldtl.BeforeInsert    := nil;
    taRtJldtl.OnNewRecord     := nil;
    taRtJldtl.BeforeDelete    := nil;
    taRtJlDtl.OnCalcFields    := nil;
  end;
end;

procedure TfmReturJual.Cancel_All_Updates;
begin
  with dmPenjualan do
  begin
    taRtJlDtl.CancelUpdates;
    taRtJl.Cancel;
  end;
end;

procedure TfmReturJual.SettingEdit(nVal : boolean);
var n : integer;
begin
  for n := 0 to fmReturJual.ComponentCount - 1 do
if fmReturJual.Components[n].Tag = 100 then
TControl(fmReturJual.Components[n]).Enabled := nVal;
end;

procedure TfmReturJual.taRtJlKODECUSTChange(Sender: TField);
begin
  with dmPenjualan do
  begin
    taRtJlKODECUST.OnChange := nil;          { supaya jangan rekursif }

    if not taCust.FindKey([Sender.asString]) then deKodeCUSTButtonClick(nil);

    taRtJlKODECUST.Value := taCust.FieldByName('KODE').asString;
    taRtJlNAMACUST.Value := taCust.FieldByName('NAMA').asString;
    taRtJlALAMAT.Value   := taCust.FieldByName('ALAMAT').asString;
    taRtJlKOTA.Value     := taCust.FieldByName('KOTA').asString;
    taRtJlNEGARA.Value   := taCust.FieldByName('NEGARA').asString;

    { jika Nama, Alamat, Kota, Negara ada isinya maka cuma readonly }
    DBEdit2.ReadOnly := not taCust.FieldByName('NAMA').isNull;
    DBEdit3.ReadOnly := not taCust.FieldByName('ALAMAT').isNull;
    DBEdit4.ReadOnly := not taCust.FieldByName('KOTA').isNull;
    DBEdit5.ReadOnly := not taCust.FieldByName('NEGARA').isNull;

    taRtJlKODECUST.OnChange := taRtJlKODECUSTChange;
  end;
end;
```

```
procedure TfmReturJual.taRtJlPPNChange(Sender: TField);
begin
  if subtotal > 0 then cePpn.Value := sender.asCurrency * 100 / subtotal;
end;

procedure TfmReturJual.taRtJlDISCChange(Sender: TField);
begin
  if subtotal > 0 then
    ceDisc.Value := sender.asCurrency * 100 / subtotal;
end;

procedure TfmReturJual.taRtJldtlKODEBRGChange(Sender: TField);
begin
  with dmPenjualan do
  begin
    dsDetail.OnDataChange     := nil;

    if not taBrg.FindKey([Sender.asString]) then DBGrid1EditButtonClick(nil);

    taRtJldtlKODEBRG.OnChange := nil;

    taRtJlDtl.Edit;
    taRtJlDtlKODEBRG.Value := taBrg.FieldByName('KODE').asString;
    taRtJlDtlNAMABRG.Value := taBrg.FieldByName('NAMA').asString;
    taRtJlDtlSATUAN.Value  := taBrg.FieldByName('SATUAN').asString;
    taRtJlDtlJUMLAH.Value  := 1;
    taRtJlDtlHARGA.Value   := 0;
    taRtJldtlDISC.Value    := 0;

    taRtJldtl.Post;
    taRtJldtlKODEBRG.OnChange := taRtJldtlKODEBRGChange;
    dsDetail.OnDataChange     := dsDetailDataChange;
  end;
end;

procedure TfmReturJual.taRtJlDtlNewRecord;
begin
  DBInitValue(dmPenjualan.taRtJlDtl);
  DBgrid1.SelectedIndex := 0;
end;

procedure TfmReturJual.taRtJlDtlCalcFields;
begin
  with dmPenjualan do
     taRtJldtlTOTAL.Value := (taRtJlDtlJUMLAH.Value * taRtJlDtlHARGA.Value) -
taRtJlDtlDISC.Value ;
end;

procedure TfmReturJual.taRtJlDtlBeforeDelete(DataSet: TDataSet);
begin
```

```
  if dsWatch.State <> dsBrowse then
    if MessageDlg('Hapus record ... ?',mtConfirmation, [mbYes, mbNo], 0) <> mrYes
then abort;
end;

procedure TfmReturJual.FormShow;
begin
  with dmPenjualan do
  begin
    DBSetPath(dmPenjualan);
    DBLoadLabel(taRtJldtl);

    taGdgBrg   := OpenTable('TGdgBrg' , '', fmReturJual); { updatable }
    taCust     := OpenTable('TCUST'   , '', fmReturJual);
    taBrg      := OpenTable('TBARANG' , '', fmReturJual);

    taRtJldtl.Open;
    taRtJl.Open;

    SetEvent;
    taRtJl.AfterScroll := AfterScroll;
    AfterScroll(taRtJl);
  end;
  deTanggal.SetFocus;

end;

procedure TfmReturJual.FormClose;
begin
  with dmPenjualan do
  begin
    taCust.Close;
    taBrg.Close;
    taGdgBrg.Close;
    taRtJldtl.Close;
    taRtJl.Close;
    taRtJl.AfterScroll := NIL;
  end;
end;

procedure TfmReturJual.KNavCancelClick;
begin
  KNav.SetFocus;
  with dmPenjualan do begin
    case dsWatch.State of
      dsInsert : begin
                   taRtJlNAMAUSER.Value := UserLogin;
                   taRtJlVOID.Value     := true;
                   taRtJl.Post;
                   taRtJldtl.CommitUpdates;
```

```
            end;
    end;
    Cancel_All_Updates;
    deTanggal.SetFocus;
  end;
end;


procedure TfmReturJual.KNavAddClick;
begin
  ClearEvent;
  with dmPenjualan do begin
    taRtJl.Append;
    DBInitValue(taRtJl);
    taRtJlNOMER.Value    := DBGetCounter('RJ', fmReturJual);
    taRtJlJENIS.Value    := 0;
    taRtJlTERM.Value     := 0;
    taRtJlDtl.EnableControls;
  end;

  subtotal                := 0;
  deKodeCust.Button.Enabled := true;
  deTanggal.SetFocus;
  SetEvent;
end;

procedure TfmReturJual.KNavSaveClick;
begin
  with dmPenjualan do begin

    KNav.SetFocus;
    dsDetail.OnDataChange := nil;  { supaya jangan rekursif }

     taRtJl.DisableControls;         { supaya semua proses di dlm save tidak berpengaruh
pada control }
    taRtJldtl.DisableControls;
    try
       if taRtJlTANGGAL.IsNull                    then DispErr('Tanggal tidak boleh
kosong', deTanggal);
       if taRtJlTANGGAL.Value <> Date             then DispErr('Tanggal harus =
tanggal sekarang', deTanggal);

      if taRtJlKODECUST.IsNull                    then DispErr('Kode Customer tidak
boleh kosong', deKodeCust);
     if not taCust.FindKey([taRtJlKODECUST.Text])  then DispErr('Kode Customer
tidak ditemukan', deKodeCust);

        if ((taRtJlTERM.Value < 0) and (taRtJlJENIS.Value = 1)) then disperr('Term
harus > 0',deTerm);
```

```
{ *** check detail *** }
taRtJldtl.First;
while not taRtJldtl.EOF do
begin
   if taRtJldtlKODEBRG.IsNull                then DispErr('Kode Barang tidak
boleh kosong', DBGrid1);
    if not taBrg.FindKey([taRtJldtlKODEBRG.Value]) then DispErr('Kode Barang
tidak ditemukan', DBGrid1);

  if not taGdgBrg.FindKey([taRtJldtlKODEBRG.Value]) then
    DispErr('Barang ' + taRtJldtlKODEBRG.Value + ' tidak ditemukan di gudang ',
DBGrid1);

   if taGdgBrg.FieldByName('SALDO').AsInteger - taRtJlDtlJUMLAH.Value < 0
then
    DispErr('Stok barang ' + taRtJldtlKODEBRG.Value + ' tidak mencukupi (' +
        FloatToStr(taGdgBrg.FieldByName('SALDO').AsInteger -
taRtJlDtlJUMLAH.Value) + ')', DBGrid1);

    if taRtJldtlJUMLAH.IsNull                then DispErr('Jumlah tidak boleh
kosong', DBGrid1);
   if taRtJlDtlJUMLAH.Value < 0              then DispErr('Jumlah tidak boleh <
0', DBGrid1);

    if taRtJldtlSATUAN.IsNull                then DispErr('Satuan tidak boleh
kosong', DBGrid1);
    if taRtJlDtlHARGA.IsNull                 then DispErr('Harga tidak boleh
kosong', DBGrid1);
   if taRtJlDtlHARGA.Value < 0               then DispErr('Harga tidak boleh < 0',
DBGrid1);

  taRtJldtl.Next;
end;

if not taPiut.Active then taPiut.Open;

if not taPiut.FindKey([taRtJlNOMER.Value]) then
  taPiut.Append
else taPiut.Edit;

taPiut.Fieldbyname('NOMER').asString       := taRtJlNOMER.Value;
taPiut.Fieldbyname('TANGGAL').asDateTime   := taRtJlTANGGAL.Value;
taPiut.Fieldbyname('KODECUST').asString    := taRtJlKODECUST.Value;
taPiut.Fieldbyname('NAMACUST').asString    := taRtJlNAMACUST.Value;
taPiut.Fieldbyname('NOREFF').asString      := '';
taPiut.Fieldbyname('IDREFF').asString      := '';
taPiut.Fieldbyname('TERM').asInteger       := taRtJlTERM.Value;
taPiut.Fieldbyname('DEBET').asCurrency     := GrandTotal;
taPiut.Fieldbyname('KREDIT').asCurrency    := 0;
```

```
      if taRtJlJENIS.Value = 0 then
      begin
        taPiut.Fieldbyname('KETERANGAN').asString := 'Retur Penjualan Tunai';
        taPiut.Fieldbyname('ID').asString         := 'RJTN';
        taPiut.Fieldbyname('BAYAR').asCurrency    := GrandTotal;
      end else
      begin
        taPiut.Fieldbyname('KETERANGAN').asString := 'Retur Penjualan Kredit';
        taPiut.Fieldbyname('ID').asString         := 'RJKR';
        taPiut.Fieldbyname('BAYAR').asCurrency    := 0;
      end;

      taPiut.Post;

      { force setting }
      taRtJlNAMAUSER.Value := UserLogin;
      taRtJl.Post;
      taRtJldtl.CommitUpdates;

      taRtJl.Edit;
      taRtJlTOTNOTA.Value  := HitungTotNota;
      DBInitValue2(taRtJl);
      taRtJl.Post;

      taRtJldtl.First;
      deTanggal.SetFocus;

    finally
      dsDetail.OnDataChange := dsDetailDataChange; { supaya jangan rekursif }
      taRtJl.EnableControls;
      taRtJldtl.EnableControls;
    end;
  end;
end;

procedure TfmReturJual.DBGrid1EditButtonClick(Sender: TObject);
begin
  with dmPenjualan do
  begin
    oldkodebrg:=taRtJlDtlKODEBRG.Value ;

    fmBrowse.brwTBARANG(DBGrid1.SelectedField.AsString, nil);
    taRtJlDtl.Edit;
    taRtJlDtlKODEBRG.Value := fmBrowse.StringGrid1.Cells[1, 0];
  end;
end;

procedure TfmReturJual.deKodeCustButtonClick;
begin
  fmBrowse.brwTCUST(deKodeCust.Text, nil);
```

```
  taCust.FindKey([fmBrowse.StringGrid1.Cells[1, 0]]);
  dmPenjualan.taRtJlKODECUST.Value := fmBrowse.StringGrid1.Cells[1, 0];
end;

procedure TfmReturJual.cePpnExit(Sender: TObject);
begin
  if dsWatch.State in [dsEdit, dsInsert] then
    dmPenjualan.taRtJlPPN.Value := (subtotal * cePpn.Value) / 100;
end;

procedure TfmReturJual.ceDiscExit(Sender: TObject);
begin
  if dsWatch.State in [dsEdit, dsInsert] then
    dmPenjualan.taRtJlDISC.Value := (subtotal * ceDisc.Value) / 100;
end;

procedure TfmReturJual.dsWatchStateChange;
begin
  if (dsWatch.State = dsEdit) then
    SettingEdit(True);

  deKodeCust.Button.Enabled := dsWatch.State in [dsEdit, dsInsert];

  ceDisc.ReadOnly          := dsWatch.State = dsBrowse;
  cePpn.ReadOnly           := dsWatch.State = dsBrowse;

  if dsWatch.State = dsBrowse then
    DBGrid1.Options := DBGrid1.Options - [dgEditing]
  else
    DBGrid1.Options := DBGrid1.Options + [dgEditing];
end;

procedure TfmReturJual.dsWatchDataChange(Sender: TObject; Field: TField);
begin
  dsWatch.OnDataChange := nil;

  try
    with dmPenjualan do
    begin
      deDisc.ReadOnly   := subtotal = 0;
      dePpn.ReadOnly    := subtotal = 0;
      CheckBox1.Checked := taRtJlPRINTED.Value > 0;

      if (subtotal > 0) and (field = nil) and (dsWatch.State <> dsBrowse) then begin
        taRtJlDISC.Value := subtotal * ceDISC.Value / 100;
        taRtJlPPN.Value  := subtotal * cePPN.Value / 100;
      end;

      grandtotal     := subtotal - taRtJldisc.value + taRtJlPPN.Value;
      Panel9.Caption := format('%m ', [grandtotal]);
```

```
    end;
  except
  end;

  dsWatch.OnDataChange := dsWatchDataChange;
end;

procedure TfmReturJual.dsDetailDataChange(Sender: TObject; Field: TField);
var nSatuan : word;
begin
  dsDetail.OnDataChange := nil;  { supaya jangan rekursif }

  try
    with dmPenjualan do
    begin
      { cari posisi kolom untuk field satuan, ini penting
        soalnya kolom dapat dipindah-pindah }
        for nSatuan := 0 to DBgrid1.FieldCount - 1 do if DBgrid1.Fields[nSatuan] =
taRtJldtlSATUAN then break;
      DBGrid1.Columns[nSatuan].PickList.Clear;

DBGrid1.Columns[nSatuan].PickList.Add(taBrg.FieldByName('SATUAN').AsString
);

      { menghitung total dari detail }
      subtotal       := DBGetTotal(taRtJldtlTOTAL);
      Panel5.Caption := format('%m ', [subtotal]);
      dsWatchDataChange(nil, nil);  { supaya respon ke Discount, PPN dan Grand Total
}
    end;
  except
  end;

  dsDetail.OnDataChange := dsDetailDataChange;
end;

procedure TfmReturJual.rgJenisBayarChange(Sender: TObject);
begin
  deTerm.Enabled  := rgJenisBayar.ItemIndex = 1;
  Label14.Enabled := rgJenisBayar.ItemIndex = 1;
  Label7.Enabled  := rgJenisBayar.ItemIndex = 1;

  if dsWatch.State in [dsEdit, dsInsert] then
    if rgJenisBayar.ItemIndex = 0 then dmPenjualan.taRtJlTERM.Value := 0;
end;

procedure TfmReturJual.Add1Click(Sender: TObject);
begin
  dmPenjualan.taRtJldtl.Append;
end;
```

```
procedure TfmReturJual.Edit1Click(Sender: TObject);
begin
  dmPenjualan.taRtJldtl.Edit;
end;

procedure TfmReturJual.Delete1Click(Sender: TObject);
begin
  if not dmPenjualan.taRtJlDtl.EOF then dmPenjualan.taRtJldtl.Delete;
end;

procedure TfmReturJual.Simpan1Click(Sender: TObject);
begin
  if dsDetail.State in [dsEdit, dsInsert] then dmPenjualan.taRtJldtl.Post;
end;

procedure TfmReturJual.Batal1Click(Sender: TObject);
begin
  dmPenjualan.taRtJldtl.Cancel;
end;

procedure TfmReturJual.PopupMenu1Popup(Sender: TObject);
begin
  if dsWatch.State = dsBrowse then abort;
end;

procedure TfmReturJual.KNavPreviewClick(Sender: TObject);
begin
  with dmPenjualan do begin
    crpe1.Clear;
    crpe1.ReportName              := PathExe + 'Report\ntJual.rpt';
    crpe1.Output                  := toWindow;
    crpe1.DiscardSavedData        := true;
    crpe1.WindowZoom.Magnification := 100;
    crpe1.WindowStyle.Title       := fmReturJual.Caption;

    crpe1.Tables.Retrieve;
    crpe1.Tables[0].Path   := AllUnits.PathData;
    crpe1.Tables.Propagate := True;

    crpe1.ParamFields.Retrieve;
    crpe1.ParamFields[0].Value := Userlogin;
    crpe1.ParamFields[1].Value := taRtJlNOMER.Text;
    crpe1.WindowButtonBar.PrintBtn := False;
    crpe1.Execute;

    crpe1.Clear;
    crpe1.ReportName              := PathExe + 'Report\ntsj.rpt';
    crpe1.Output                  := toWindow;
    crpe1.DiscardSavedData        := true;
```

```
    crpe1.WindowZoom.Magnification := 100;
    crpe1.WindowStyle.Title        := 'Surat Jalan';

    crpe1.Tables.Retrieve;
    crpe1.Tables[0].Path   := AllUnits.PathData;
    crpe1.Tables.Propagate := True;

    crpe1.ParamFields.Retrieve;
    crpe1.ParamFields[0].Value := Userlogin;
    crpe1.ParamFields[1].Value := taRtJlNOMER.Text;
    crpe1.WindowButtonBar.PrintBtn := False;
    crpe1.Execute;
  end;
end;

procedure TfmReturJual.KNavSearchClick(Sender: TObject);
begin
  fmSearch.LoadSearch(dmPenjualan.taRtJl);
end;

procedure TfmReturJual.FormCloseQuery(Sender: TObject; var CanClose: Boolean);
begin
  CanClose := dsWatch.State=dsBrowse;
end;

Function TfmReturJual.HitungTotNota:Currency;
Var GrandTotNota : Currency;
Begin
  With dmPenjualan do
  begin
    taRtJlDtl.DisableControls;
    taRtJlDtl.First;
    GrandTotNota := 0;
    While Not taRtJlDtl.EOF do
    Begin
      // Sum GrandTotal
                GrandTotNota   :=   GrandTotNota   +   (taRtJlDtlJUMLAH.Value      *
taRtJlDtlHARGA.Value) - taRtJlDtlDISC.Value;

      // Update GdgBrg
      if taGdgBrg.FindKey([taRtJldtlKODEBRG.Value]) then
      begin
        taGdgBrg.Edit;
    taGdgBrg.FieldByName('SALDO').AsFloat :=
taGdgBrg.FieldByName('SALDO').AsFloat +
                                    taRtJldtlJUMLAH.Value;
      end;

      taGdgBrg.Post;
      taRtJlDtl.Next;
```

```
    End;
    taRtJlDtl.EnableControls;
  End;
  Result := GrandTotNota;
End;

procedure TfmReturJual.DBGrid1ColEnter(Sender: TObject);
begin
  oldKodeBrg:=dmPenjualan.taRtJlDtlKodebrg.value;
end;

procedure TfmReturJual.DBGrid1KeyPress(Sender: TObject; var Key: Char);
begin
  if DBGrid1.SelectedField.FieldName='KODEBRG' then
    Key:=UpCase(Key);
end;

procedure TfmReturJual.CheckBox1Enter(Sender: TObject);
begin
  PostMessage(Handle, WM_NEXTDLGCTL, 0, 0)
end;

end.
```

Unit USearch;

```
Interface

Uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  Db, DBTables, Grids, DBGrids, StdCtrls, ToolWin,
  ComCtrls, Buttons, RXDBCtrl;

Type
  TfmSearch = Class(TForm)
    dsNav: TDataSource;
    DBGrid1: TRxDBGrid;
    taSearch: TTable;
    CoolBar1: TCoolBar;
    edKey: TEdit;
    btnGo: TBitBtn;
    cbKeyField: TComboBox;
    Procedure LoadSearch(myTable: TTable);
    Procedure btnGoClick(Sender: TObject);
    Procedure edKeyChange(Sender: TObject);
    Procedure edKeyKeyDown(Sender: TObject; Var Key: Word;
      Shift: TShiftState);
    Procedure DBGrid1DblClick(Sender: TObject);
    Procedure FormClose(Sender: TObject; Var Action: TCloseAction);
```

```
    Procedure FormShow(Sender: TObject);
    Procedure cbKeyFieldChange(Sender: TObject);
    Procedure DBGrid1KeyDown(Sender: TObject; Var Key: Word;
      Shift: TShiftState);
  private
    { Private declarations }
  public
    { Public declarations }
    nValue: String;
  End;

Var
  fmSearch: TfmSearch;

Implementation

{$R *.DFM}

Uses Allunits;

Procedure TFmSearch.LoadSearch(myTable: TTable);
Begin
  taSearch.DataBaseName := myTable.DatabaseName;
  taSearch.TableName := myTable.TableName;
  taSearch.Open;
  DBLoadLabel(taSearch);
  taSearch.Locate(myTable.Fields[0].FieldName, myTable.Fields[0].AsString, []);
  fmSearch.ShowModal;

  //Return Value
  If nValue <> '' Then
    MyTable.Locate(myTable.Fields[0].FieldName, fmSearch.nValue, []);
End;

Procedure TfmSearch.btnGoClick(Sender: TObject);
Var
   BM : TBookMark;
   Junk: Double;
Begin
  If Not taSearch.Filtered Then
  Begin
//    taSearch.Filter := cbKeyField.Text + '=''' + edKey.Text + '*''';


    if
(taSearch.FieldByName(DBGrid1.Columns[cbKeyField.ItemIndex].FieldName).Data
Type=ftString) Then
        taSearch.Filter := DBGrid1.Columns[cbKeyField.ItemIndex].FieldName + '=''' +
edKey.Text + '*'''
    else
```

```
    begin
      try
        Junk:=StrToDate(edkey.Text);
         taSearch.Filter := DBGrid1.Columns[cbKeyField.ItemIndex].FieldName + '=''' +
edKey.Text + '''';
      except
        taSearch.Filter:='';
      end;
    end;
    taSearch.Filtered := True;
    btnGo.Caption := '&Filter On';
  End Else
  Begin
    BM := taSearch.GetBookmark;
    btnGo.Caption := '&Filter Off';
    taSearch.Filtered := False;
    taSearch.GotoBookmark(BM);
    taSearch.FreeBookmark(BM);
  End;
End;

Procedure TfmSearch.edKeyChange(Sender: TObject);
var
  Junk: Double;
Begin
  If edKey.Text <> '' Then
  begin
    if taSearch.Filtered then
      if
(taSearch.FieldByName(DBGrid1.Columns[cbKeyField.ItemIndex].FieldName).Data
Type=ftString) Then
        taSearch.Filter := DBGrid1.Columns[cbKeyField.ItemIndex].FieldName + '=''' +
edKey.Text + '*'''
      else
      begin
        try
          Junk:=StrToDate(edkey.Text);
            taSearch.Filter := DBGrid1.Columns[cbKeyField.ItemIndex].FieldName + '='''
+ edKey.Text + '''';
        except
          taSearch.Filter:='';
        end;
      end;
    If     (Length(edKey.Text)     <=
taSearch.FieldByName(DBGrid1.Columns[cbKeyField.ItemIndex].FieldName).Size)
or
(taSearch.FieldByName(DBGrid1.Columns[cbKeyField.ItemIndex].FieldName).Data
Type=ftDate) Then
    try
```

```
      if
taSearch.FieldByName(DBGrid1.Columns[cbKeyField.ItemIndex].FieldName).Data
Type=ftDate then
       Junk:=StrToDate(edkey.Text);
      TaSearch.Locate(DBGrid1.Columns[cbKeyField.ItemIndex].FieldName,
edKey.Text, [loCaseInsensitive, loPartialKey]);
    except
    end;
  End;
End;

Procedure TfmSearch.edKeyKeyDown(Sender: TObject; Var Key: Word;
  Shift: TShiftState);
Begin
  If Key = VK_RETURN Then
  Begin
    nValue := taSearch.Fields[0].AsString;
    Close;
  End;

  If Key = VK_ESCAPE Then
    Close;
End;

Procedure TfmSearch.DBGrid1DblClick(Sender: TObject);
Begin
  nValue := taSearch.Fields[0].AsString;
  Close;
End;

Procedure TfmSearch.FormClose(Sender: TObject; Var Action: TCloseAction);
Begin
  taSearch.Filter := '';
  taSearch.Filtered := False;
  btnGo.Caption := '&Filter Off';
  taSearch.Close;
End;

Procedure TfmSearch.FormShow(Sender: TObject);
Var
  i, j: Byte;
Begin
  Dbgrid1.Columns.Clear;
  DBGrid1.TitleFont.Style := [fsBold];
  cbKeyField.Clear;
  edKey.Clear;

  i := 0;
  j := 0;
  While (i < 8) And (j <= taSearch.FieldCount - 1) Do
```

```
  Begin
                if (taSearch.Fields[j].DataType in [ftString,ftDate]) And Not
((taSearch.Fields[j].FieldName = 'NAMAUSER') Or (taSearch.Fields[j].FieldName =
'NAMAPRINT')) Then
    Begin
      i := i + 1;
      cbKeyField.Items.Add(taSearch.Fields[j].DisplayLabel);
      AddColumn(DbGrid1, taSearch.Fields[j].FieldName);
    End;
    J := J + 1;
  End;
  cbKeyField.ItemIndex := 0;
End;

Procedure TfmSearch.cbKeyFieldChange(Sender: TObject);
var
  junk: Double;
Begin
  If edKey.Text <> '' Then
  begin
    if taSearch.Filtered then
      if
(taSearch.FieldByName(DBGrid1.Columns[cbKeyField.ItemIndex].FieldName).Data
Type=ftString) Then
        taSearch.Filter := DBGrid1.Columns[cbKeyField.ItemIndex].FieldName + '=''' +
edKey.Text + '*'''
      else
      begin
        try
          Junk:=StrToDate(edkey.Text);
            taSearch.Filter := DBGrid1.Columns[cbKeyField.ItemIndex].FieldName + '='''
+ edKey.Text + '';
        except
          taSearch.Filter:='';
        end;
      end;
          If  (Length(edKey.Text)  <=
taSearch.FieldByName(DBGrid1.Columns[cbKeyField.ItemIndex].FieldName).Size)
or
(taSearch.FieldByName(DBGrid1.Columns[cbKeyField.ItemIndex].FieldName).Data
Type=ftDate) Then
    try
      if
taSearch.FieldByName(DBGrid1.Columns[cbKeyField.ItemIndex].FieldName).Data
Type=ftDate then
        Junk:=StrToDate(edkey.Text);
      TaSearch.Locate(DBGrid1.Columns[cbKeyField.ItemIndex].FieldName,
edKey.Text, [loCaseInsensitive, loPartialKey]);
    except
    end;
```

```
  End;
  edKey.SetFocus;
End;

Procedure TfmSearch.DBGrid1KeyDown(Sender: TObject; Var Key: Word;
  Shift: TShiftState);
Begin
  If Key = VK_RETURN Then
  Begin
    nValue := taSearch.Fields[0].AsString;
    Close;
  End;
End;

End.
```

unit uSrv;

```
interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  Menus, Grids, DBGrids, ComCtrls, ZKNavigator, ExtCtrls, StdCtrls,
  DBCtrls, Mask, Db, DBTables, ToolEdit, RXDBCtrl;

type
  TfmSrv = class(TForm)
    Bevel1: TBevel;
    DBGrid1: TDBGrid;
    Panel3: TPanel;
    Label1: TLabel;
    Label2: TLabel;
    Label3: TLabel;
    Panel2: TPanel;
    DBText1: TDBText;
    deTanggal: TDBDateEdit;
    deketerangan: TDBMemo;
    KNav: TK2Navigator;
    dsWatch: TDataSource;
    dsDetail: TDataSource;
    PopupMenu1: TPopupMenu;
    Add1: TMenuItem;
    Edit1: TMenuItem;
    Delete1: TMenuItem;
    Simpan1: TMenuItem;
    Batal1: TMenuItem;
    Panel4: TPanel;
    Label4: TLabel;
    Panel1: TPanel;
```

```
    procedure DBGrid1EditButtonClick(Sender: TObject);
    procedure FormShow(Sender: TObject);
    procedure FormClose(Sender: TObject; var Action: TCloseAction);
    procedure KNavAddClick(Sender: TObject);
    procedure KNavCancelClick(Sender: TObject);
    procedure KNavSaveClick(Sender: TObject);
    procedure dsWatchStateChange(Sender: TObject);
    procedure dsDetailDataChange(Sender: TObject; Field: TField);
    procedure Add1Click(Sender: TObject);
    procedure Edit1Click(Sender: TObject);
    procedure Delete1Click(Sender: TObject);
    procedure Simpan1Click(Sender: TObject);
    procedure Batal1Click(Sender: TObject);
    procedure PopupMenu1Popup(Sender: TObject);
    procedure KNavPreviewClick(Sender: TObject);
    procedure KNavSearchClick(Sender: TObject);
    procedure DBGrid1KeyPress(Sender: TObject; var Key: Char);
  private
    subtotal : currency;

    procedure taSrvDtlKODEBRGChange(Sender: TField);
    procedure taSrvDtlNewRecord(DataSet: TDataSet);
    procedure taSrvDtlBeforeDelete(DataSet: TDataSet);
    procedure taSrvDtlCalcFields(DataSet: TDataSet);
  public
  end;

var
  fmSrv: TfmSrv;

implementation

uses UdmInventory, AllUnits, UBrowse, USearch;

{$R *.DFM}

procedure TfmSrv.taSrvDtlKODEBRGChange(Sender: TField);
begin
  with dmInventory do
  begin
    taSrvDtlKODEBRG.OnChange := nil;

    if not taService.FindKey([Sender.asString]) then DBGrid1EditButtonClick(nil);
    taSrvDtl.Edit;
    taSrvDtlKODEBRG.Value := taService.FieldByName('KODE').asString;
    taSrvDtlNAMABRG.Value := taService.FieldByName('NAMA').asString;
    taSrvDtlHARGA.Value   := taService.FieldByName('HARGA').asCurrency;
    taSrvDtlJUMLAH.Value  := 1;
    taSrvDtl.Post;
```

```
   taSrvDtlKODEBRG.OnChange := taSrvDtlKODEBRGChange;
  end;
end;

procedure TfmSrv.taSrvDtlCalcFields;
begin
  with dmInventory do
    If taSrvDtl.Active Then
      taSrvDtlTOTAL.Value := taSrvDtlJUMLAH.Value * taSrvDtlHARGA.Value;
end;

procedure TfmSrv.taSrvDtlNewRecord;
begin
  DBInitValue(dmInventory.taSrvdtl);
  DBGrid1.SelectedIndex := 0;
end;

procedure TfmSrv.taSrvDtlBeforeDelete(DataSet: TDataSet);
begin
  if dsWatch.State <> dsBrowse then
     if MessageDlg('Hapus record ... ?',mtConfirmation,[mbYes, mbNo], 0) <> mrYes
then abort;
end;

procedure TfmSrv.FormShow;
begin
  DBSetPath(dmInventory);                              { set databasename to pathdata
}

  with dmInventory do
  begin
    taService.Open;
    taSrv.Open;
    taSrvdtl.Open;

     DBLoadLabel(dmInventory.taSrvdtl);                  { field names's short
description used to DBGrid Title }

    taSrvDtlKODEBRG.OnChange := taSrvDtlKODEBRGChange;
    taSrvdtl.OnNewRecord     := taSrvdtlNewRecord;
    taSrvdtl.BeforeDelete    := taSrvDtlBeforeDelete;
    taSrvdtl.OnCalcFields    := taSrvdtlCalcFields;
  end;
end;

procedure TfmSrv.FormClose;
begin
  dmInventory.taSrvdtl.Close;
  dmInventory.taSrv.Close;
  dmInventory.taService.Close;
```

```
end;

procedure TfmSrv.KNavAddClick;
begin
  with dmInventory do begin
    taSrv.Append;
    DBInitValue(taSrv);
    taSrvNOMER.Value    := DBGetCounter('SR', fmSrv);
    taSrvNAMAUSER.Value := UserLogin;
    deTanggal.SetFocus;
  end;
end;

procedure TfmSrv.KNavCancelClick(Sender: TObject);
begin
  with dmInventory do begin
    if dsWatch.State = dsInsert then
      taSrv.Post;

    taSrv.Cancel;
    taSrvdtl.CancelUpdates;
  end;
  deTanggal.SetFocus;
end;

procedure TfmSrv.KNavSaveClick;
begin
  KNav.SetFocus;

  with dmInventory do
  try
    taSrvdtl.DisableControls;

    { *** checking *** }
     if taSrvTANGGAL.IsNull                then DispErr('Tanggal tidak boleh kosong',
deTanggal);

    taSrvDtl.First;
    while not taSrvDtl.EOF do
    begin
      if taSrvdtlKODEBRG.IsNull                then DispErr('Kode Service tidak boleh
kosong', DBGrid1);
           if not taService.FindKey([taSrvdtlKODEBRG.Value]) then DispErr('Kode
Service tidak ditemukan', DBGrid1);

      if taSrvdtlJUMLAH.IsNull                    then DispErr('Jumlah tidak boleh
kosong', DBGrid1);
      if taSrvdtlJUMLAH.Value < 1                then DispErr('Jumlah tidak boleh < 0',
DBGrid1);
```

```
      if taSrvDtlHARGA.IsNull                    then DispErr('Harga tidak boleh kosong',
DBGrid1);
       if taSrvDtlHARGA.Value < 0                then DispErr('Harga tidak boleh < 0',
DBGrid1);

     taSrvDtl.Next;
    end;

    taSrvNAMAUSER.Value := UserLogin;
    taSrvTOTAL.Value    := subtotal;

    DBInitValue2(taSrv);
    taSrv.Post;
    taSrvdtl.CommitUpdates;
  finally
    taSrvdtl.First;
    taSrvdtl.EnableControls;
  end;
  deTanggal.SetFocus;
end;

procedure TfmSrv.DBGrid1EditButtonClick;
begin
  with dmInventory do
  begin
    fmBrowse.brwTSERVICE(DBGrid1.SelectedField.AsString, nil);
    taService.FindKey([fmBrowse.StringGrid1.Cells[1, 0]]);
    taSrvdtl.Edit;
    taSrvDtlKODEBRG.Value := fmBrowse.StringGrid1.Cells[1, 0];
  end;
end;

procedure TfmSrv.dsWatchStateChange;
begin
  DBgrid1.ReadOnly := dsWatch.State = dsBrowse;

  if dsWatch.State = dsBrowse then
    DBgrid1.Options := Dbgrid1.Options - [dgEditing]
  else
    DBgrid1.Options := Dbgrid1.Options + [dgEditing];
end;

procedure TfmSrv.dsDetailDataChange;
begin
  dsDetail.OnDataChange := nil;

  try
    with dmInventory do
    begin
      subtotal       := DBGetTotal(taSrvdtlTOTAL);
```

```
      Panel1.Caption := format('%m ', [subtotal]);
    end;
  except
  end;

  dsDetail.OnDataChange := dsDetailDataChange;
end;

procedure TfmSrv.Add1Click(Sender: TObject);
begin
  dmInventory.taSrvDtl.Insert;
end;

procedure TfmSrv.Edit1Click(Sender: TObject);
begin
  dmInventory.taSrvDtl.Edit;
end;

procedure TfmSrv.Delete1Click(Sender: TObject);
begin
  dmInventory.taSrvDtl.Delete;
end;

procedure TfmSrv.Simpan1Click(Sender: TObject);
begin
  if dsDetail.State in [dsEdit, dsInsert] then dmInventory.taSrvDtl.Post;
end;

procedure TfmSrv.Batal1Click(Sender: TObject);
begin
  dmInventory.taSrvDtl.Cancel;
end;

procedure TfmSrv.PopupMenu1Popup(Sender: TObject);
begin
  if dsWatch.State = dsBrowse then abort;
end;

procedure TfmSrv.KNavPreviewClick(Sender: TObject);
begin
  with dmInventory do
  Begin
    Crpe.Clear;
    Crpe.ReportName := PathReport + '\ntStokIn.rpt';
    Crpe.Tables.Retrieve;
    Crpe.Tables[0].Path := PathData;
    Crpe.Tables.Propagate := True;
    Crpe.WindowStyle.Title := Caption;
    Crpe.WindowSize.Height := Screen.Height;
    Crpe.WindowSize.Width := Screen.Width;
```

```
    Crpe.WindowSize.Left := 0;
    Crpe.WindowSize.Top := 0;
    Crpe.ParamFields.Retrieve;
    Crpe.ParamFields[0].Value := UserLogin;
    Crpe.ParamFields[1].Value := taSrvNOMER.Value;
    Crpe.Execute;
  end;
end;

procedure TfmSrv.KNavSearchClick(Sender: TObject);
begin
  fmSearch.LoadSearch(dmInventory.taSrv);
end;

procedure TfmSrv.DBGrid1KeyPress(Sender: TObject; var Key: Char);
begin
  if DBGrid1.SelectedField.FieldName='KODEBRG' then
    Key:=UpCase(Key);

  if DBGrid1.SelectedField.FieldName='HARGA' then
    Key := #0;
end;

end.
```