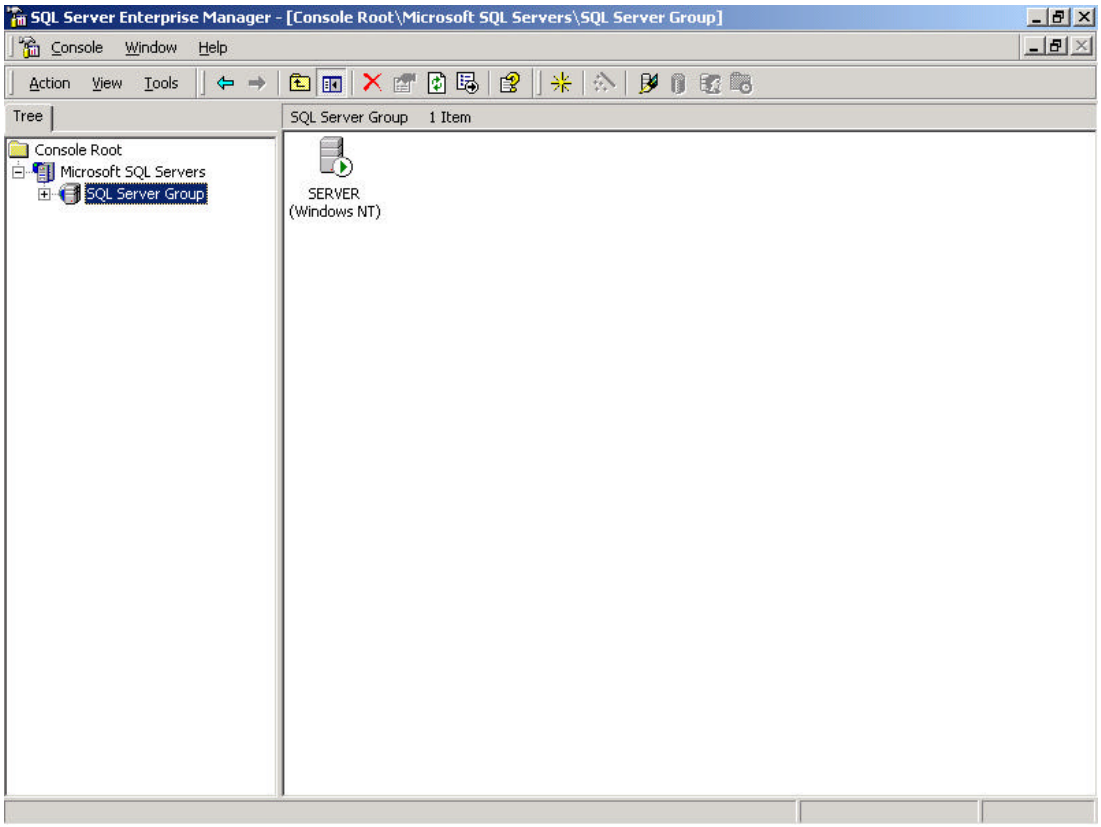
5. IMPLEMENTASI SISTEM

5.1. *Setting* Awal

Setting awal dilakukan dengan menyiapkan satu *unit* komputer sebagai *server* yaitu AMD Duron – 1,13 G Mhz dengan *memory* 128 Mb PC 133 dan IDE Hard Drive 20 GB 7200 RPM. Proses *setting* ini dilanjutkan dengan melakukan penginstalan komputer *server* dengan Microsoft SQL Server 7.0. untuk *database server*.

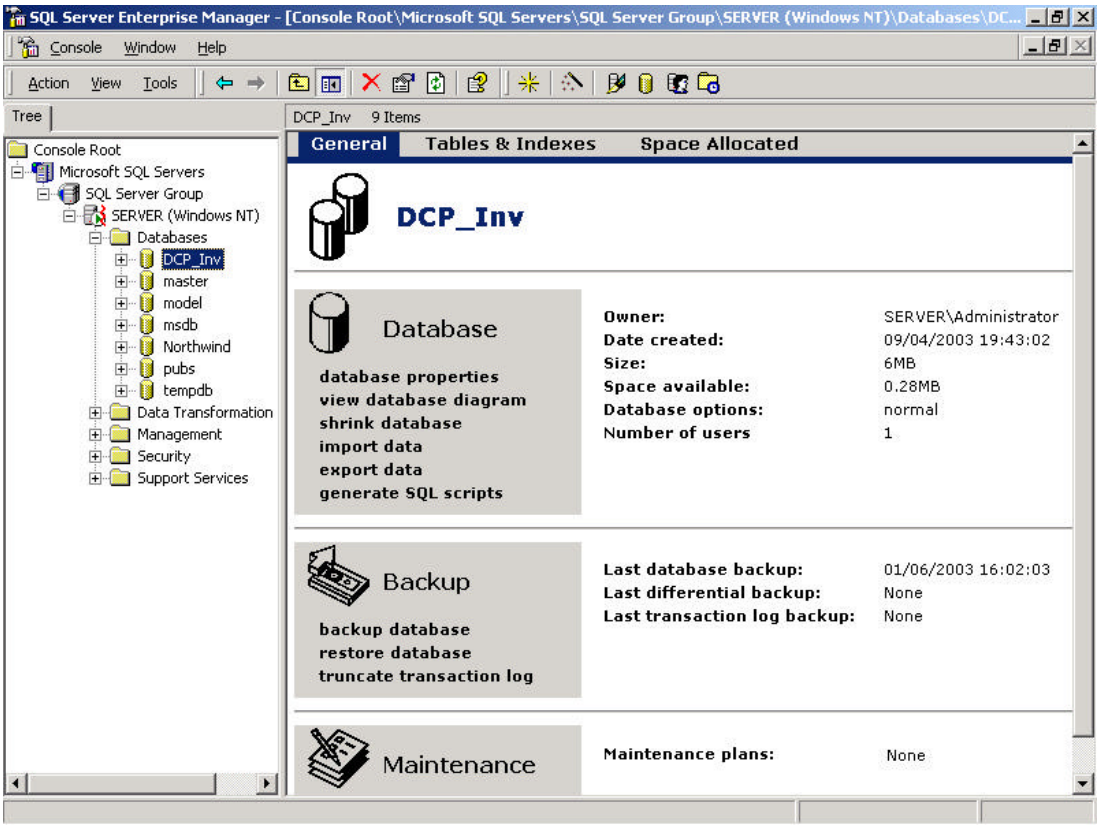
Program pembelian, penjualan dan persediaan barang ini dibuat dengan database SQL Server 7.0. dan memakai *interface* yang dibuat dengan *software* Borland Delphi 6.0. Sedangkan sistem yang digunakan server berbasis sistem operasi Windows 2000 Server Family.

Setting SQL Server dilakukan dengan pembuatan SQL *Server Group* yang bernama "SERVER" yang dapat dilihat pada Gambar 5.1.



Gambar 5.1. *Set Up Group*

Langkah selanjutnya adalah pembuatan *database* dengan nama "DCP_Inv" yang dapat dilihat pada Gambar 5.2. Langkah selanjutnya adalah menggunakan *user* dengan nama "administrator" yang kemudian diset untuk memiliki wewenang sebagai *database administrator* (DBO) dan mensetting password dengan "xyz".



Gambar 5.2. *Database* DCP_Inv

Pembuatan dialihkan kepada pembuatan interface dengan menggunakan Borland Delphi 6.0. *Setting* utama ada pada pembuatan datamodule yang kemudian menggunakan TDatabase yang disetting mengarah kepada SQL Server 7.0. di *server* yaitu dengan membuat *connection string* SQL Server dengan nama "SERVER". Proses setelah itu melakukan test koneksi, jika berhasil maka koneksi sudah terbentuk.

Pembuatan datamodule1 pada Delphi sebagai tempat tabel, yang menjadi pusat tabel dan datasource. Penggunaan TDatabase yang disetting dihubungkan dengan SQL Server. TTabel dan TDatasource masing-masing tabel yang ada di SQL

Server *database* DCP_Inv disetting dan penggunaan TSession untuk sharing data pada jaringan.

5.2. Komponen-Komponen Delphi

Proses desain *interface* dengan mempergunakan kompnen-komponen delphi seperti TDBEdit, TDBGrid, TDBLookupComboBox, dan TDBComboBox. Komponen-komponen itu merupakan komponen bersifat *visual* komponen atau yang disebut sebagai VCL (*Visual Component Libarary*) artinya komponen yang dirancang pada saat perancangan maupun saat program dijalankan.

Komponen-komponen akan diletakan pada tiap-tiap *form*, yang berfungsi untuk menginputkan data dan menampilkan data. Pada saat perancangannya peletakan dari komponen-komponen itu harus disesuaikan dengan fungsi dan juga untuk memperindah penampilan *form-form* yang ada.

5.3. DataModule Delphi

Proses selanjutnya adalah pembuatan Datamodule dengan cara memilih *menu File*, *New*, Datamodule. Datamodule adalah komponen delphi yang bersifat non *visual*, yang artinya hanya terlihat pada saat perancangan program. Fungsinya merupakan suatu wadah yang menjadi konkesi dari *database* yang dipergunakan dalam program. Di dalam datamodule ada dua komponen utama yaitu dataset dan datasource, dimana dataset merupakan penghubung antara *database* dan aplikasi sedangkan datasource merupakan penghubung *database* dengan *form-form* yang dipergunakan.

Komponen dataset yang dipergunakan dalam program ini antara lain adalah TDatabase, TSession dan TTable. TDatabase adalah merupakan komponen yang fungsinya sebagai koneksi awal dengan Microsoft SQL Server. TSession adalah komponen non *visual* delphi yang dipergunakan untuk sharing *database*. TTable adalah merupakan penghubung dengan Tabel-Tabel yang dipergunakan. Sedangkan komponen datasource adalah TDatasource.

5.4. Implementasi *Interface*

Desain *interface* yang dipergunakan dalam program adalah MDI *Form* (*Multiple Display Instrument Form*) yang terdiri dari sebuah *form* utama dengan *menu pull down* maupun *icon-icon* yang berfungsi untuk memanggil *form* sesuai dengan fungsi-fungsinya. *Form* utama ini merupakan tempat bergantungnya *form-form* anak (sesuai fungsinya) dan dapat diatur penyusunannya. Pada Tabel 5.1. ini akan digambarkan secara umum fungsi dari tiap-tiap *form*.

Tabel 5.1. Fungsi Tiap-tiap *Form*

Nama *Form*	**Fungsi**
Form Password Dialog	Untuk menampilkan *setting password* dari program.
Form Barang	Untuk menampilkan dan menyimpan data barang.
Form Supplier	Untuk menampilkan dan menyimpan data *supplier*.
Form Customer	Untuk menampilkan dan menyimpan data *customer*.
Form EOQ	Untuk menampilkan dan menyimpan data EOQ beserta dengan *ReOrder Point*.
Form Koreksi *Stock*	Untuk menampilkan dan menyimpan data koreksi *stock*.
Form Order Beli	Untuk menampilkan dan menyimpan data *order* beli.
Form Pembelian	Untuk menampilkan dan menyimpan data pembelian beserta total pembayarannya.
Form Hutang	Untuk menampilkan dan menyimpan data-data hutang.
Form PU Pembelian	Untuk menampilkan dan menyimpan data PU pembelian.
Form Bayar Hutang	Untuk menampilkan dan menyimpan data pembayaran hutang.

. (lanjutan) Fungsi Tiap-tiap *Form*

Form Retur Pembelian	Untuk menampilkan dan menyimpan data retur pembelian.
Form Penjualan	Untuk menampilkan dan menyimpan data penjualan beserta total pembayarannya.
Form Piutang	Untuk menampilkan dan menyimpan data-data piutang.
Form PU Penjualan	Untuk menampilkan dan menyimpan data PU penjualan.
Form Bayar Piutang	Untuk menampilkan dan menyimpan data pembayaran piutang.
Form Retur Penjualan	Untuk menampilkan dan menyimpan data retur penjualan.

Pada saat program dijalankan pertama kali maka *menu* yang dapat diaktifkan adalah hanya *user*.untuk melakukan proses *log in*. maka muncul *password start up dialog* yaitu sebuah *form* untuk memasukkan nama pengguna dan *password* sesuai dengan *database* yang ada. Apabila salah memasukkan maka langsung program tertutup. Apabila data yang dimasukkan benar maka *menu-menu* yang ada dalam main *form* akan dapat dipergunakan. Prosedur yang dipergunakan di dalam *form password dialog* dapat dilihat pada Tabel 5.2.

Tabel 5.2. Prosedur *Password Dialog*

Prosedur dalam *form* ini	**Fungsi**
TPasswordDlg.CancelBtnClick	Batal mempergunakan program
TPasswordDlg.OKBtnClick	Mengecek nama pengguna dan *password* sesuai dengan *database*. Apabila tidak sesuai maka program akan tertutup.
TPasswordDlg.Sort	Mensortir nama pengguna dan *password* sesuai dengan data yang masukkan.

Segmen Program 5.1. Proses *Form Password Dialog*

```
Procedure TPasswordDlg.Sort;
begin
        with datamodule1 do
        begin
                Table26.IndexNAme:= 'PK_Pass';
                Table26.SetKey;
                Table26.FieldByName('User_Name').AsString:= User_Name.Text;
                Table26.GotoNearest;
                Edit2.Text:= Table26User_Level.AsString;
        end;
end;
```

5.4.1. *Interface Main Form*

Pembuatan *main menu* terdiri dari *User*, *File*, *Stock*, Pembelian, Penjualan, Grafik, *Setup*, *Window* dan *Help*.

- *Menu* "*User*" terdiri dari "log in", yang dipergunakan untuk mengakses program.
- *Menu* "*File*" terdiri dari "Barang" untuk membuka *form* data barang. "*Supplier*" untuk membuka *form* data *supplier*. "*Customer*" untuk membuka *form* data *customer*. "Tutup" berfungsi untuk menutup *form* aktif. "Sampul" berfungsi untuk menutup main *form* dengan sampul dari program. "Keluar" untuk mematikan program.
- "*Stock*" terdiri dari "EOQ" berfungsi untuk membuka *form* perhitungan EOQ dan "Koreksi *Stock*" berfungsi untuk membuka *form* koreksi *stock*.
- "Pembelian" terdiri dari "*Order*" berfungsi untuk membuka *form order* barang, "beli" berfungsi untuk membuka *form* pembelian barang, "PU" untuk membuka *form* PU pembelian, "Hutang" terdiri dari sub *menu* "Hutang" berfungsi untuk membuka *form* data hutang dan sub *menu* "bayar" untuk membuka *form* data pembayaran hutang. "Retur" berfungsi untuk membuka *form* pengembalian pembelian barang.
- "Penjualan" terdiri dari "Penjualan" berfungsi untuk membuka *form* penjualan barang, "PU" untuk membuka *form* PU penjualan, "Piutang" terdiri dari sub *menu* "Piutang" berfungsi untuk membuka *form* data piutang dan sub *menu* "bayar" untuk membuka *form* data pembayaran hutang. "Retur" berfungsi untuk membuka *form* pengembalian penjualan barang.

- "Grafik" terdiri dari "EOQ" berfungsi untuk menampilkan grafik EOQ sesuai dengan kode barang. "Beli" berfungsi untuk menampilkan grafik pembelian sesuai tanggal pembelian. "Jual" berfungsi untuk menampilkan grafik penjualan sesuai tanggal penjualan. "Retur" terdiri dari sub *menu* "beli" berfungsi untuk menampilkan grafik retur pembelian dan sub *menu* "Jual" berfungsi untuk menampilkan grafik retur penjualan. "*Stock*" berfungsi untuk menampilkan grafik jumlah *stock* barang dalam gudang sesuai dengan kode barang.
- "Setup" terdiri dari *menu* "Periode" berfungsi untuk membuka *form* periode. "Otoritas" terdiri dari sub *menu* "Gudang" berfungsi membatasi penggunaan program untuk bagian gudang, sub *menu* "Beli" berfungsi membatasi penggunaan program untuk bagian pembelian, sub *menu* "jual" berfungsi membatasi penggunaan program untuk bagian penjualan. "*Password*" berfungsi untuk membuka *form* input data *password*.
- "*Window*" terdiri dari *menu* "*Cascade*" berfungsi untuk menyusun *form-form* yang dibuka secara bertumpuk. "Tile *Horizontally*" berfungsi untuk menyusun *form-form* yang dibuka sejajar horisontal. "Tile *Vertically*" berfungsi untuk menyusun *form-form* yang dibuka sejajar vertikal.
- "*Help*" terdiri dari *menu* "*About*" berfungsi untuk menampilkan identitas pemrogram.

5.4.2. *Interface Form* Barang

Proses pembuatan *form* barang ini menggunakan komponen-komponen TDBEdit dan TDBCombo yang beerfungsi untuk memasukkan data master barang dan menampilkan tiap recordnya sedangkan TDBGrid untuk menampilkan seluruh isi data master barang. Pada saat *form* ini diaktifan, maka tabel yang digunakan adalah tabel dari barang. Prosedur yang dipergunakan di dalam *form* barang dapat dilihat pada Tabel 5.3.

Tabel 5.3. Prosedur *Form* Barang

Prosedur dalam *form* ini	**Fungsi**
TFInMaster.Input1Click	Membuat *record* baru dengan kode barang secara unik.
TFInMaster.Lihat1Click	Menampilkan cakupan data barang yang akan diproses di *quick report*.
TFInMaster.Cetak1Click	Mencetak semua data barang.

Segmen Program 5.2. Proses *Form* Barang

```
Procedure TFInMaster.Input1Click(Sender: TObject);
var i: integer;
begin
        openedit;
        with Datamodule1 do
        begin
            Table1.Insert;
            i:= Table1.RecordCount;
            if i<10 then DBEdit1.text:= 'BG'+ '0000' + inttostr(i + 1)
            else
                if (i=10) or (i<100) then DBEdit1.text:= 'BG' + '000' +
                inttostr(i + 1)
            else DBEdit1.text:= 'BG' + '00' +  inttostr(i + 1);
        end;
end;

Procedure TFInMaster.DBComboBox2Exit(Sender: TObject);
var     s: integer; a,b: real;
begin
        with datamodule1 do
        begin
        Table1.post;
        Table1.edit;
        b:= strtofloat(DBComboBox2.text);
        a:= Table1H_Beli.value;
        Table1H_Jual.Value:= (a + (a * b/100));
        s:= DBComboBox2.itemindex;
        case s of
                0: DBEdit7.Text:= DBEdit5.text;
                1: DBEdit7.Text:= floattostr(a + ( a * 5/100));
                2: DBEdit7.Text:= floattostr(a + ( a * 10/100));
                3: DBEdit7.Text:= floattostr(a + ( a * 15/100));
                4: DBEdit7.Text:= floattostr(a + ( a * 20/100));
                5: DBEdit7.Text:= floattostr(a + ( a * 25/100));
        end;
        end;
end;
```

```
Procedure TFInMaster.Lihat1Click(Sender: TObject);
begin
        Report1Dlg:= TReport1Dlg.create(application);
        Report1Dlg.show;
end;

Procedure TFInMaster.Cetak1Click(Sender: TObject);
begin
        FQ1:= TFQ1.create(application);
        FQ1.QuickRep1.Preview;
end;
```

5.4.3. *Interface Form Supplier*

Proses pembuatan *form supplier* ini menggunakan komponen-komponen TDBEdit dan TDBCombo yang berfungsi untuk memasukkan data *supplier* dan menampilkan tiap recordnya. Pada saat *form* ini diaktifan, maka tabel yang digunakan adalah tabel dari *supplier*. Prosedur yang dipergunakan di dalam *form supplier* dapat dilihat pada Tabel 5.4.

Tabel 5.4. Prosedur *Form Supplier*

Prosedur dalam *form* ini	**Fungsi**
TFInSupp.Input1Click	Membuat *record* baru dengan kode *supplier* secara unik.
TFInSupp.Lihat1Click	Menampilkan cakupan data *supplier* yang akan diproses di *quick report*.
TFInSupp.Cetak1Click	Mencetak semua data *supplier*.

Segmen Program 5.3. Proses *Form Supplier*

```
Procedure TFInSupp.Input1Click(Sender: TObject);
var i: integer;
begin
        openedit;
        with Datamodule1 do
        begin
           i:= Table2.RecordCount;
           Table2.insert;
           if i<10 then EditKode_Supp.Text:= 'SP'+ '0000' +  inttostr(i + 1)
           else
               if (i=10) or (i<100) then EditKode_Supp.Text:= 'SP' + '000' +
               inttostr(i + 1)
           else
               if (i=100) or (i<1000) then EditKode_Supp.Text:= 'SP'+ '00' +
```

```
            inttostr(i + 1)
         else EditKode_Supp.Text:= 'SP0' + inttostr(i + 1)
      end;
      EditFax.Text:= '-';
      EditSince.Text:= datetostr(date);
end;
```

5.4.4. *Interface Form Customer*

Proses pembuatan *form customer* ini menggunakan komponen-komponen TDBEdit dan TDBCombo yang berfungsi untuk memasukkan data *customer* dan menampilkan tiap recordnya. Pada saat *form* ini diaktifan, maka tabel yang digunakan adalah tabel dari *customer*. Prosedur yang dipergunakan di dalam *form customer* dapat dilihat pada Tabel 5.5.

Tabel 5.5. Prosedur *Form Customer*

Prosedur dalam *form* ini	**Fungsi**
TFInCust.Input1Click	Membuat *record* baru dengan kode *customer* secara unik.
TFInCust.Lihat1Click	Menampilkan cakupan data *customer* yang akan diproses di *quick report*.
TFInCust.Cetak1Click	Mencetak semua data *customer*.

Segmen Program 5.4. Proses *Form Customer*

```
Procedure TFInCust.Input1Click(Sender: TObject);
var i: integer;
begin
      openedit;
      with Datamodule1 do
      begin
        i:= Table16.RecordCount;
        Table16.Insert;
        if i < 10 then EditNo_Cust.Text:= 'CS' + '0000' + inttostr(i + 1)
        else if (i=10) or (1<100) then EditNo_Cust.Text:= 'CS'+ '000' +
             inttostr(i + 1)
        else if (i= 100) or (i<1000) then EditNo_Cust.Text:= 'CS' + '00' +
             inttostr(i + 1)
        else EditNo_Cust.Text:= 'CS0' + inttostr(i + 1);
        DBEdit1.text:= '0';
        DBEdit2.Text:= '0';
      end;
      EditFax.Text:= '-';
      EditSince.Text:= datetostr(date);
end;
```

5.4.5. *Interface Form* EOQ

Proses pembuatan *form* EOQ ini menggunakan komponen-komponen TDBEdit, TDBCombo, TDBLookupComboBox, TEdit yang berfungsi untuk memasukkan dan mengolah data EOQ serta menampilkan tiap recordnya. Pada saat *form* ini diaktifan, maka tabel yang digunakan adalah tabel dari barang dan tabel dari stock_barang. Prosedur yang dipergunakan di dalam *form* EOQ dapat dilihat pada Tabel 5.6.

Tabel 5.6. Prosedur *Form* EOQ

Prosedur dalam *form* ini	**Fungsi**
TFInStock.DBEdit1Exit	Menghitung total *cost* dengan menjumlahkan biaya simpan dan biaya *order*.
TFInStock.Image1Click	Menghitung EOQ.
TFInStock.Button2Click	Menghitung ROP dan memasukannya pada data barang sesuai dengan kode barang.
TFInStock.Lihat1Click	Menampilkan cakupan data EOQ yang akan diproses di *quick report*.
TFInStock.Cetak1Click	Mencetak semua data EOQ.

Segmen Program 5.5. Proses *Form* EOQ

```
Procedure TFInStock.DBEdit1Exit(Sender: TObject);
begin
    with datamodule1 do
    begin
    Table3.Edit;
    Table3Total_Cost.Value:= Table3B_Simpan.Value + Table3Biaya_order.Value;
    Table3.Post;
    end;
end;

Procedure TFInStock.Image1Click(Sender: TObject); {hitung EOQ}
var d, k, h: real;
s:string;
begin
        with datamodule1 do
        begin
```

```
                Table3.edit;
                d:= Table3Saldo_Awal.value;
                k:= Table3Biaya_order.Value;
                h:= Table3B_Simpan.value;
                Table3EOQ.value:= Sqrt((2 * D * K)/H);
                Table3.Post;
        end;
end;

Procedure TFInStock.Button2Click(Sender: TObject);
var a: integer;
begin
        with datamodule1 do
        begin
           Table1.IndexName:= 'PK_Barang';
           Table1.SetKey;
           Table1.Fieldbyname('Kode_Brg').AsString:= DBLookupComboBox1.Text;
           Table1.GotoNearest;
           a:= strtoint(Edit3.Text);
           Edit4.Text:= floattostr(Table3Lead_Time.Value * a);
           Table1.Edit;
           Table1ROP.Value:= Table3Lead_Time.Value * a;
           Table1.post;
        end;
end;
```

5.4.6. *Interface Form* Koreksi *Stock*

Proses pembuatan *form* koreksi *stock* ini menggunakan komponen-komponen TDBEdit, TDBCombo, TDBLookupComboBox, Tedit yang berfungsi untuk memasukkan dan mengolah data koreksi *stock* serta menampilkan tiap recordnya. Pada saat *form* ini diaktifan, maka tabel yang digunakan adalah tabel dari Barang, tabel dari koreksi_stock dan tabel dari koreksi_stock_detail. Prosedur yang dipergunakan di dalam *form* koreksi *stock* dapat dilihat pada Tabel 5.7.

Tabel 5.7. Prosedur *Form* Koreksi *Stock*

Prosedur dalam *form* ini	**Fungsi**
TFInSK.DBCombo2Change	Mengurangi *stock* barang apbila dipilih “hilang”, Menambah *stock* barang apabila dipilih “penambahan”
TFInSK.Lihat1Click	Menampilkan cakupan data dari koreksi *stock* yang akan diproses di *quick report*.
TFInSK.Cetak1Click	Mencetak semua data koreksi *stock*.

Segmen Program 5.6. Proses *Form* Koreksi *Stock*

```
Procedure TFInSK.DBComboBox2Change(Sender: TObject);
var a, i: integer;
begin
        with datamodule1 do
        begin
             a:= DBCombobox2.ItemIndex;
             case a of
             0: begin
                    i:= strtoint(DBComboBox1.Text);
                    Table1.Edit;
                    Table1Stock_gudang.Value:= Table1Stock_gudang.Value - i;
                    Table1.Post;
                    if Table1Stock_gudang.Value <= Table1ROP.Value then
                        begin
                             SMIN2Dlg:= TSMIN2Dlg.create(application);
                             SMIN2Dlg.show;
                        end
                end;
             1: begin
                    i:= strtoint(DBComboBox1.Text);
                    Table1.Edit;
                    Table1Stock_gudang.Value:= Table1Stock_gudang.Value + i;
                    Table1.Post;
                end;
             end;
        end;
end;

Procedure TFInSK.Lihat1Click(Sender: TObject);
begin
        Report1Dlg:= TReport1Dlg.create(application);
        Report1Dlg.show;
end;
```

5.4.7. *Interface Form Order* Beli

Proses pembuatan *form order* beli ini menggunakan komponen-komponen TDBEdit, TDBCombo, TDBLookupComboBox, Tedit yang berfungsi untuk memasukkan dan mengolah data *order* beli serta menampilkan tiap recordnya. Pada saat *form* ini diaktifan, maka tabel yang digunakan adalah tabel dari barang, tabel dari *supplier*, tabel dari order_beli dan tabel dari order_beli_detail. Prosedur yang dipergunakan di dalam *form order* beli dapat dilihat pada Tabel 5.8.

Tabel 5.8. Prosedur *Form Order* Beli

Prosedur dalam *form* ini	**Fungsi**
TFInOrder_Beli.DBEdit3Change	Menampilkan data barang pada DBEdit1 (Nama Barang), DBEdit2 (Berat Barang) dan EditHarga (Harga Jual) sesuai dengan kode barang yang dimasukan pada DBEdit3 itu.
TFInOrder_Beli.DBEdit3Exit	Mengecek kode *supplier* yang dimiliki oleh data barang yang dimasukan. Apabila tidak sesuai maka akan dibatalkan.
TFInOrder_Beli.DBEdit4Exit	Menghitung harga jual dikalikan kuantiti *order* itu.
TFInOrder_Beli.Lihat1Click	Menampilkan cakupan data dari *order* beli yang akan diproses di *quick report*.
TFInOrder_Beli.Cetak1Click	Mencetak semua data *order* beli

Segmen Program 5.7. Proses *Form Order* Beli

```
Procedure TFInOrder_Beli.DBEdit3Change(Sender: TObject);
begin
        with datamodule1 do
        begin
                Table1.IndexName:= 'PK_Barang';
                Table1.SetKey;
                Table1.Fieldbyname('Kode_Brg').AsString:= DBEdit3.Text;
                Table1.GotoNearest;
                DBEdit1.Text:= Table1Nama_Brg.AsString;
                DBEdit2.Text:= Table1Berat.AsString;
                EditHarga.Text:= Table1H_Beli.AsString;
        end;
end;

Procedure TFInOrder_Beli.DBEdit3Exit(Sender: TObject);
begin
        with datamodule1 do
        begin
         if (Table2Kode_Supp.Value = 'SP00001') and
            (Table1Kode_Brg.value >= 'BG00025') then
                begin
                     MessageDLG('Supplier error',mtinformation,[mbOK],0);
                     Table7.Cancel;
                     DBEdit3.Setfocus;
```

```
                end;
          if (Table2Kode_Supp.Value = 'SP00002') and
             (Table1Kode_Brg.value < 'BG00025') then
                begin
                        MessageDLG('Supplier error',mtinformation,[mbOK],0);
                        Table7.Cancel;
                        DBEdit3.Setfocus;
                end;
          end;
end;

Procedure TFInOrder_Beli.Lihat1Click(Sender: TObject);
begin
        Report1Dlg:= TReport1Dlg.create(application);
        Report1Dlg.show;
end;
```

5.4.8. *Interface Form* Pembelian

Proses pembuatan *form* pembelian ini menggunakan komponen-komponen TDBEdit, TDBCombo, TDBLookupComboBox, yang berfungsi untuk memasukkan dan mengolah data *order* beli serta menampilkan tiap recordnya, TDBGrid yang berfungsi untuk menampilkan data pembelian barang. Pada saat *form* ini diaktifan, maka tabel yang digunakan adalah tabel dari barang, tabel dari pembelian, tabel dari order_beli dan tabel dari order_beli_detail.

Pada saat nomor *order* dicari (*events closeup* dari TDBLookupComboBox1) maka TDBGrid akan menampilkan data detail *order* sesuai nomor *order*.

Pembelian dapat dilakukan secara tunai maupun secara kredit, apabila dikredit maka status pembayarannya menjadi kredit, dan akan dibaca pada jatuh temponya. Prosedur yang dipergunakan di dalam *form* pembelian dapat dilihat pada Tabel 5.9.

Tabel 5.9. Prosedur *Form* Pembelian

Prosedur dalam *form* ini	**Fungsi**
TFInBeli.DBLookupComboBox1CloseUp	Menampilkan detail *oder* pada DBGrid sesuai nomor *order*, dilanjutkan dengan prosedur HitungTotal.
TFInBeli.HitungTotal	• Menjumlahkan total isi dari *field* Harga_Q pada tabel detail *order* dan memasukan hasilnya pada *field* Total_Beli tabel pembelian. • Menambahkan jumlah *stock* barang pada tabel barang sesuai dengan kode barang.
TFInBeli.DBComboBox2Change	Mengecek status pembayaran proses pembelian, apabila pembelian secara kredit, maka akan membuka *form* hutang.
TFInBeli.Lihat1Click	Menampilkan cakupan data dari pembelian yang akan diproses di *quick report*.
TFInBeli.Cetak1Click	Mencetak semua data pembelian

Segmen Program 5.8. Proses *Form* Pembelian

```
Procedure TFInBeli.DBLookupComboBox1CloseUp(Sender: TObject);
var a: integer; b: real;
begin
      with datamodule1 do
      begin
           Table7.IndexName:= 'IX_Order_Beli_Detail_1';
           Table7.SetKey;
           Table7.FieldByName('No_Order').AsString:= DBLookupComboBox1.Text;
           Table7.GotoNearest;
           if DBLookupComboBox1.Text = DBEDit3.Text then
           begin
                DBEdit4.Text:= Table6Kode_Supp.Asstring;
                EditBiaya_Order.Text:= floattostr(Table6Biaya_Order.Value);
                SpeedButton12.enabled:= false;
                HitungTotal;
                DBGrid2.visible:= true;
           end;
```

```
     end;
end;

Procedure TFInBeli.DBComboBox2Change(Sender: TObject);
var i: integer;
begin
        i:= DBComboBox2.ItemIndex;
        with datamodule1 do
        case i of
        0: begin
                FHutang:= TFHutang.create(application);
                FHutang.show;
           end;
        end;
end;
```

5.4.9. *Interface Form* Hutang

Proses pembuatan *form* hutang ini menggunakan komponen-komponen TDBEdit, TDBCombo, TDBLookupComboBox, yang berfungsi untuk memasukkan dan mengolah data hutang serta menampilkan tiap recordnya. Pada saat *form* ini diaktifan, maka tabel yang digunakan adalah tabel dari hutang, tabel dari pembelian dan tabel dari periode.

Form hutang ini dapat muncul secara otomatis apabila saat membuka *form* pembelian dan proses pembayaran yang dilakukan secara kredit dan juga bisa dibuka di sub *menu* pembelian. Prosedur yang dipergunakan di dalam *form* hutang dapat dilihat pada Tabel 5.10.

Tabel 5.10. Prosedur *Form* Hutang

Prosedur dalam *form* ini	**Fungsi**
TFHutang.DBLookupComboBox2CloseUp	Menampilkan nomor faktur pembelian yang akan dibayar secara kredit beserta total pembayarannya.
TFHutang.Lihat1Click	Menampilkan cakupan data dari hutang yang akan diproses di *quick report*.
TFHutang.Cetak1Click	Mencetak semua data hutang.

Segmen Program 5.9. Proses *Form* Hutang

```
Procedure TFHutang.DBLookupComboBox2CloseUp(Sender: TObject);
begin
        with datamodule1 do
        begin
             DBEdit2.Text:= Table8Kode_Supp.AsString;
             EditSaldo_Awal.Text:= Floattostr(Table8Total_Beli.Value);
        end;
end;
```

5.4.10. *Interface Form* PU Pembelian

Proses pembuatan *form* PU pembelian ini menggunakan komponen-komponen TDBEdit, TDBCombo, TDBLookupComboBox, yang berfungsi untuk memasukkan dan mengolah data PU pembelian serta menampilkan tiap recordnya. Pada saat *form* ini diaktifan, maka tabel yang digunakan adalah tabel dari pembelian, tabel dari PU_pembelian dan PU_pembelian_detail.

Form ini dipergunakan untuk melakukan proses pembayaran hutang dari beberapa faktur pembelian yang dibayar secara kredit. Sekaligus juga mengubah status bayar dari faktur pembelian yang telah diproses. Apabila pembelian sudah dilunasi maka status pembayarannya dan jatuh temponya tidak berlaku lagi. Prosedur yang dipergunakan di dalam *form* PU pembelian dapat dilihat pada Tabel 5.11.

Tabel 5.11. Prosedur *Form* PU Pembelian

Prosedur dalam *form* ini	**Fungsi**
TFPU1.DBLookupComboBox1CloseUp	Menampilkan data pembelian sesuai dengan faktur pembelian.
TFPU1.Lihat1Click	Menampilkan cakupan data dari PU pembelian yang akan diproses di *quick report*.
TFIPU1.Cetak1Click	Mencetak semua data PU pembelian

Segmen Program 5.10. Proses *Form* PU Pembelian

```
Procedure TFPU1.DBLookupComboBox1CloseUp(Sender: TObject);
begin
     with datamodule1 do
     begin
         EditJumlah_Bayar.Text:= Floattostr(Table13Saldo_Awal.value);
         if (DBEdit1.Text = EditNo_PU.Text) and
```

```
            (DBEdit2.text = DBEdit3.text) then
           begin
                 MessageDLG('Hutang sudah dibayar',mtinformation,[mbOK],0);
                 Table12.Cancel;
           end;
      end;
end;
```

5.4.11. *Interface Form* Bayar Hutang

Proses pembuatan *form* bayar hutang ini menggunakan komponen-komponen TDBEdit, TDBCombo, TDBLookupComboBox, yang berfungsi untuk memasukkan dan mengolah data bayar hutang dari PU pembelian serta menampilkan tiap recordnya. Pada saat *form* ini diaktifan, maka tabel yang digunakan adalah tabel dari bayar_hutang, tabel dari PU_pembelian dan tabel dari PU_pembelian_detail.

Form pembayaran hutang ini hanya dipergunakan untuk mendata hutang-hutang yang sudah dilunasi, melalui proses PU pembelian. Prosedur yang dipergunakan di dalam *form* bayar hutang dapat dilihat pada Tabel 5.12.

Tabel 5.12. Prosedur *Form* Bayar Hutang

Prosedur dalam *form* ini	**Fungsi**
TFInBayar_H.DBLookupComboBox1CloseUp	Menampilkan jumlah hutang yang telah dibayar di proses PU pembelian
TFInBayar_H.Lihat1Click	Menampilkan cakupan data dari bayar hutang yang akan diproses di *quick report*.
TFInBayar_H.Cetak1Click	Mencetak semua data bayar hutang.

Segmen Program 5.11. Proses *Form* Bayar Hutang

```
Procedure TFInBayar_H.DBLookupComboBox1CloseUp(Sender: TObject);
begin
        with datamodule1 do
        begin
             EditJumlah.Text:= floattostr(Table11T_Bayar.Value);
             EditDibayar.Text:= floattostr(Table11T_Bayar.Value);
        end;
end;
```

5.4.12. *Interface Form* Retur Pembelian

Proses pembuatan *form* retur pembelian ini menggunakan komponen-komponen TDBEdit, TDBCombo, TDBLookupComboBox, yang berfungsi untuk memasukkan dan mengolah data retur pembelian serta menampilkan tiap recordnya. Pada saat *form* ini diaktifan, maka tabel yang digunakan adalah tabel dari barang, tabel dari retur_beli, tabel dari retur_beli_detail, tabel dari pembelian dan tabel dari order_beli_detail.

Form retur beli ini akan mengurangi jumlah total pembelian dan mengurangi jumlah *stock* barang. Prosedur yang dipergunakan di dalam *form* retur pembelian dapat dilihat pada Tabel 5.13.

Tabel 5.13. Prosedur *Form* Retur Pembelian

Prosedure dalam *form* ini	**Fungsi**
TFRBeli.DBLookupComboBox1Closeup	Menampilkan harga pembelian sesuai kode barang.
TFRBeli.Lanjut	• Mengurangi jumlah *stock* dalam gudang sesuai dengan jumlah yang diretur. • Mengecek *stock minimum* dalam gudang sesuai dengan ROP, dapat dilanjutkan proses *order* beli. • Mengalikan harga pembelian dengan kuantiti yang diretur. • Mengurangi jumlah total pembelian pada tabel pembelian.
TFRBeli.Lihat1Click	Menampilkan cakupan data dari retur pembelian yang akan diproses di *quick report*.
TFRBeli.Cetak1Click	Mencetak semua data retur pembelian

Segmen Program 5.12. Proses *Form* Retur Pembelian

```
Procedure TFRBeli.DBLookupComboBox1CloseUp(Sender: TObject);
begin
     with datamodule1 do
     begin
         Table15Qty.Value:= Table9Qty.Value;
         Table15Harga.Value:= Table9Harga.value;
         if DBEdit2.Text <> DBEDit3.Text then
           begin
                MessageDLG('Kode Brg error',mtinformation,[mbOK],0);
                Table15.Cancel;
           end;
      end;
end;

Procedure TFRBeli.Lanjut;
begin
    with datamodule1 do
    begin
         Table15.post; Table1.Edit;
         Table1Stock_gudang.Value:= Table1Stock_gudang.Value -
                                    Table15Jumlah_Ganti.value;
         Table1.Post; Table15.edit;
         If Table1Stock_gudang.Value <= Table1ROP.Value then

{menampilkan unit stok minimum utk dilanjutkan dg reorder}
             begin
                SMIN2Dlg:= TSMIN2Dlg.create(application);
                SMIN2Dlg.show;
             end;
         Table15Harga.Value:= Table1H_Beli.value *
                              strtoint(DBComboBox1.Text);
         Table15.post;
         Table8.IndexName:= 'PK_Pembelian;
         Table8.SetKey;
         Table8.Fieldbyname('No_Faktur').AsString:= DBLookupComboBox2.Text;
         Table8.GotoNearest;
         Table8.Edit;
         Table8Total_Beli.Value:= Table8Total_Beli.Value –
                                  Table15Harga.Value;
         Table8.post;
    end;
end;
```

5.4.13. *Interface Form* Penjualan

Proses pembuatan *form* penjualan ini menggunakan komponen-komponen TDBEdit, TDBCombo, TDBLookupComboBox, yang berfungsi untuk memasukkan dan mengolah data penjualan serta menampilkan tiap recordnya, TDBGrid yang berfungsi untuk menampilkan data detail pembelian barang. Pada saat *form* ini diaktifan, maka tabel yang digunakan adalah tabel dari *customer*, tabel dari barang, tabel dari penjualan, tabel dari penjualan_detail dan tabel dari piutang.

Detail penjualan barang dapat dilakukan dengan memasukan kode nomor barang pada TDBEdit7 atau dengan menggunakan fasilitas searching data barang. Fasilitas searching ini dapat mencari berdasarkan nama barang ataupun kode barang.

Penjualan dapat dilakukan secara tunai maupun secara kredit, apabila dikredit maka status pembayarannya menjadi kredit, dan akan dibaca pada jatuh temponya. Apabila seorang *customer* sudah jatuh tempo masa pembayaran piutangnya maka masih dapat dilayani. Prosedur yang dipergunakan di dalam *form* penjualan dapat dilihat pada Tabel 5.14.

Tabel 5.14. Prosedur *Form* Penjualan

Prosedure dalam *form* ini	**Fungsi**
TFJ1.DBLookupComboBox1CloseUp	Mengecek status piutang dari *customer*, dengan 3 pilihan • OK : langsung dibayar piutangnya dengan *form* PU Penjualan. • Batal : transaksi penjualan dibatalkan. • Lanjutkan : transaksi penjualan dilanjutkan .
TFJ1.DBEdit7Change	Menampilkan data barang pada DBEdit2 (Nama Barang), DBEdit3 (Berat Barang), DBEdit5 (Harga Penjualan)

Tabel 5.14. (lanjutan) Prosedur *Form* Penjualan

TFJ1.DBEdit8Exit	• Mengalikan kuantiti dengan harga penjualan dimasukan ke *field* HargaQ. • Mengurangi jumlah *stock* barang pada tabel barang sesuai deengan kode barang.
TFJ1.TotalJual	Menghitung total HargaQ dan memasukan pada tabel Penjualan dengan field T_Bayar dan Total.
TFJ1.Lanjut	Mengecek status pembayaran proses penjualan, apabila penjualan secara kredit, maka akan membuka *form* piutang.
TFJ1.Lihat1Click	Menampilkan cakupan data dari penjualan yang akan diproses di *quick report*.
TFJ1.Cetak1Click	Mencetak semua data penjualan.

Segmen Program 5.13. Proses *Form* Penjualan

```
Procedure TFJ1.DBEdit7Change(Sender: TObject);
begin
        with datamodule1 do
        begin
                Table1.IndexName:= 'PK_Barang';
                Table1.SetKey;
                Table1.Fieldbyname('Kode_Brg').AsString:= DBEdit7.Text;
                Table1.GotoNearest;
                DBEdit2.Text:= Table1Nama_Brg.AsString;
                DBEdit3.text:= Table1Berat.AsString;
                DBEdit5.text:= Table1H_Satuan.AsString;
                EditHarga.TEXT:= '0';
        end;
end;

procedure TFJ1.DBEdit8Exit(Sender: TObject);
var a: integer;
begin
        with datamodule1 do
        begin
```

```
            Table1.Refresh;
            a:= strtoint(DBEdit8.Text);
            Table1.Edit;
            Table1Stock_gudang.Value:= Table1Stock_gudang.Value - a;
            Table1.Post;
            if Table1Stock_gudang.Value <= Table1ROP.Value then
            begin
               SMIN2Dlg:= TSMIN2Dlg.create(application);
               SMIN2Dlg.show;
            end;
            subtotal;
            Totaljual;
            Table1.Refresh;
        end;
        DBEDit7.Setfocus;
        Speedbutton12.Enabled:= false;
end;

Procedure TFJ1.DBLookupComboBox1CloseUp(Sender: TObject);
begin
        cekcust;
end;

Procedure TFJ1.cekcust;
begin
        with datamodule1 do
        begin
        if  Table16THutang.value >= 1 then
              begin
                 RPU2Dlg:= TRPU2Dlg.create(application);
                 RPU2Dlg.show;
                 RPU2Dlg.Edit1.text:= Table22No_Faktur.Asstring;
                 RPU2Dlg.Edit2.text:= Table16No_Cust.Asstring;
                 RPU2Dlg.Edit4.text:= Table16Status.AsString;
                 Table16THutang.Displayformat:= '#,#.00';
              end;
        end;
end;

procedure TFJ1.Totaljual;
var TempTotal: Extended;
    PrevRecord: TBookmark;
begin
     with datamodule1 do
     begin
          PrevRecord := Table18.GetBookmark;
          try
             Table18.DisableControls;
             Table18.First;
             TempTotal := 0;
             while not Table18.EOF do
             begin
                  TempTotal := TempTotal + Table18HargaQ.Value;
                  Table18.Next;
```

```
            end;
            table17.edit;
            Table17T_Bayar.Value := TempTotal;
            Table17Total.Value:= TempTotal;
            table17.post;
          finally
          Table18.EnableControls;
          if PrevRecord <> nil then
          begin
               Table18.GoToBookmark(PrevRecord);
               Table18.FreeBookmark(PrevRecord);
          end;
     end;
     end;
end;

Procedure TFJ1.Lanjut;
var i: integer;
begin
    with datamodule1 do
      begin
          FPiutang:= TFPiutang.create(application);
          FPiutang.show;
          FPiutang.OpenEdit;
          Table22.Insert;
          i:= Table22.RecordCount;
          if i<10 then Table22No_FPiutang.value:= 'NP0000' + Inttostr(i + 1)
              else if (i=10) or (i<100) then
                    Table22No_FPiutang.value:= 'NP000'+ Inttostr(i + 1)
              else if (i=100) or (i<1000) then
                    Table22No_FPiutang.value:= 'NP00' + Inttostr(i + 1)
              else Table22No_FPiutang.value:= 'NP0' + Inttostr(i + 1);
              Table16.SetKey;
              Table16.FieldByName('No_Cust').AsString:=
                      FJ1.DBLookupComboBox1.text;
              Table16.Edit;
              Table16THutang.Value:= Table17Total.value;
              Table16Status.Value:= Datetostr(Table17Tgl_Jatuh_Tempo.Value);
              Table16.post;
       end;
end;
```

5.4.14. *Interface Form* Piutang

Proses pembuatan *form* piutang ini menggunakan komponen-komponen TDBEdit, TDBCombo, TDBLookupComboBox, yang berfungsi untuk memasukkan dan mengolah data piutang serta menampilkan tiap recordnya. Pada saat *form* ini diaktifan, maka tabel yang digunakan adalah tabel dari piutang, tabel dari penjualan dan tabel dari periode.

Form piutang ini dapat muncul secara otomatis apabila saat membuka *form* penjualan dan proses pembayaran yang dilakukan secara kredit dan juga bisa dibuka di sub *menu* penjualan. Prosedur yang dipergunakan di dalam *form* piutang dapat dilihat pada Tabel 5.15.

Tabel 5.15. Prosedur *Form* Piutang

Prosedure dalam *form* ini	**Fungsi**
TFPiutang.DBLookupComboBox1	Menampilkan nomor faktur penjualan yang akan dibayar secara kredit beserta total pembayarannya.
TFPiutang.Lihat1Click	Menampilkan cakupan data dari piutang yang akan diproses di *quick report*.
TFPiutang.Cetak1Click	Mencetak semua piutang

Segmen Program 5.14. Proses *Form* Piutang

```
Procedure TFPiutang.DBLookupComboBox1CloseUp(Sender: TObject);
begin
        with datamodule1 do
        begin
             DBEdit2.Text:= Table17Kode_Cust.Asstring;
             DBEdit3.Text:= Table17Total.Asstring;
        end;
end;
```

5.4.15. *Interface Form* PU Penjualan

Proses pembuatan *form* PU penjualan ini menggunakan komponen-komponen TDBEdit, TDBCombo, TDBLookupComboBox, yang berfungsi untuk memasukkan dan mengolah data PU penjualan serta menampilkan tiap recordnya. Pada saat *form* ini diaktifan, maka tabel yang digunakan adalah tabel dari *customer*, tabel dari penjualan, tabel dari PU_penjualan dan PU_penjualan_detail.

Form ini dipergunakan untuk melakukan proses pembayaran piutang dari beberapa faktur penjualan yang dibayar secara kredit. Sekaligus juga mengubah status bayar dari faktur penjualan yang diproses itu. Apabila penjualan sudah dilunasi maka status penjualannya dan jatuh temponya tidak berlaku lagi.

Mengubah status piutang *customer* sesuai nomor faktur yang diproses pada *form* PU penjualan. Prosedur yang dipergunakan di dalam *form* PU penjualan dapat dilihat pada Tabel 5.16.

Tabel 5.16. Prosedur *Form* PU Penjualan

Prosedure dalam *form* ini	**Fungsi**
TFPU2.DBLookupComboBox1CloseUp	Menampilkan data penjualan sesuai dengan faktur penjualan.
TFPU2.Lihat1Click	Menampilkan cakupan data dari PU penjualan yang akan diproses di *quick report*.
TFIPU2.Cetak1Click	Mencetak semua data PU penjualan

Segmen Program 5.15. Proses *Form* PU Penjualan

```
Procedure TFPU2.DBLookupComboBox1CloseUp(Sender: TObject);
begin
     with datamodule1 do
     begin
         EditJUmlah_bayar.Text:= Floattostr(Table22Saldo_awal.Value);
         DBEdit1.Text:= Table22Kode_Cust.Asstring;
         if (DBEdit3.text = EditNo_PU.text) and
            (DBEdit4.text = DBEdit5.text) then
           begin
               MessageDLG('Piutang sudah dibayar',mtinformation,[mbOK],0);
                         Table21.Cancel;
           end;
      end;
end;
```

5.4.16. *Interface Form* Bayar Piutang

Proses pembuatan *form* bayar piutang ini menggunakan komponen-komponen TDBEdit, TDBCombo, TDBLookupComboBox, yang berfungsi untuk memasukkan dan mengolah data bayar piutang dari PU penjualan serta menampilkan tiap recordnya. Pada saat *form* ini diaktifan, maka tabel yang digunakan adalah tabel dari bayar_piutang, tabel dari PU_penjualan dan tabel dari PU_penjualan_detail.

Form pembayaran hutang ini hanya dipergunakan untuk mendata piutang-piutang yang sudah dilunasi melalui proses PU penjualan. Prosedur yang dipergunakan di dalam *form* bayar piutang dapat dilihat pada Tabel 5.17.

Tabel 5.17. Prosedur *Form* Bayar Piutang

Prosedure dalam *form* ini	**Fungsi**
TFBPiutang.DBLookupComboBox1	Menampilkan jumlah piutang yang telah dibayar di proses PU penjualan
TFBPiutang.Lihat1Click	Menampilkan cakupan data dari bayar piutang yang akan diproses di *quick report*.
TFBPiutang.Cetak1Click	Mencetak semua data bayar piutang

Segmen Program 5.16. Proses *Form* Bayar Piutang

```
Procedure TFBPiutang.DBLookupComboBox1CloseUp(Sender: TObject);
begin
        with datamodule1 do
        begin
             EditJumlah.Text:= floattostr(Table20T_Bayar.Value);
             EditDibayar.Text:= floattostr(Table20T_Bayar.Value);
        end;
end;

Procedure TFBPiutang.Lihat1Click(Sender: TObject);
begin
        Report1Dlg:= TReport1Dlg.create(application);
        Report1Dlg.show;
end;
```

5.4.17. *Interface Form* Retur Penjualan

Proses pembuatan *form* retur penjualan ini menggunakan komponen-komponen TDBEdit, TDBCombo, TDBLookupComboBox, yang berfungsi untuk memasukkan dan mengolah data retur penjualan serta menampilkan tiap recordnya. Pada saat *form* ini diaktifan, maka tabel yang digunakan adalah tabel dari Barang, tabel dari Retur_Jual, tabel dari Retur_Jual_Detail, tabel dari Penjualan dan tabel dari Penjualan_Detail.

Form retur jual ini akan mengurangi jumlah total penjualan dan menambah jumlah *stock* barang. Prosedur yang dipergunakan di dalam *form* retur penjualan dapat dilihat pada Tabel 5.18.

Tabel 5.18. Prosedur *Form* Retur Penjualan

Prosedure dalam *form* ini	**Fungsi**
TFRetur_Jual.DBLookupComboBox1Closeup	Menampilkan harga penjualan sesuai kode barang.
TFRetur_Jual.DBLookupComboBox1Exit	• Menambah jumlah *stock* dalam gudang sesuai dengan jumlah yang diretur. • Mengalikan harga penjualan dengan kuantiti yang diretur. • Mengurangi jumlah total penjualan pada tabel penjualan.
TFRetur_Jual.Lihat1Click	Menampilkan cakupan data dari retur penjualan yang akan diproses di *quick report*.
TFRetur_Jual.Cetak1Click	Mencetak semua data retur penjualan.

Segmen Program 5.17. Proses *Form* Retur Penjualan

```
Procedure TFRetur_jual.DBLookupComboBox1CloseUp(Sender: TObject);
begin
     with Datamodule1 do
     begin
          Table24Qty.value:= Table29Qty.Value;
          Table24Harga.value:= Table29Harga.Value;
          if DBEdit2.Text <> DBEDit3.Text then
             begin
                  MessageDLG('Kode Brg error',mtinformation,[mbOK],0);
                  Table24.Cancel;
             end;
      end;
end;

Procedure TFRetur_jual.Lanjut;
begin
   with datamodule1 do
   begin
       Table24.post;  Table1.Edit;
       Table1Stock_gudang.Value:= Table1Stock_gudang.Value +
                                  Table24Jumlah_Ganti.value;
       Table1.Post; Table24.Edit;
```

```
          Table24Harga.Value:= Table1H_Satuan.Value *
                               strtoint(DBComboBox1.Text);
          Table24.Post;
          Table17.IndexName:= 'PK_Penjualan';
          Table17.SetKey;
          Table17.Fieldbyname('No_Faktur').AsString:= DBLookupComboBox2.Text;
          Table17.GotoNearest;
          Table17.Edit;
          Table17Total.Value:= Table17Total.Value - Table24Harga.Value;
          Table17.post;
   end;
        closeedit;
        SpeedButton24.enabled:= false;
end;
```