

Modul Praktek Laboratorium Komputer

Borland Delphi



Di susun oleh :

Team Penyusun Modul Delphi

Akademi Manajemen Informatika & Komputer
BINA SARANA INFORMATIKA
Jakarta
2006

Bab I

Mengenal Delphi

1.1. Pengertian Delphi

Delphi adalah suatu bahasa pemrograman (*development language*) yang digunakan untuk merancang suatu aplikasi program.

a. Kegunaan Delphi

1. untuk membuat aplikasi windows
2. Untuk merancang aplikasi program berbasis grafis
3. Untuk membuat program berbasis jaringan (client/server)
4. Untuk merancang program .Net (berbasis internet)

b. Keunggulan Delphi

1. IDE (*Integrated Development Environment*) atau lingkungan pengembangan aplikasi sendiri adalah satu dari beberapa keunggulan delphi, didalamnya terdapat menu – menu yang memudahkan kita untuk membuat suatu proyek program.
2. Proses Kompilasi cepat, pada saat aplikasi yang kita buat dijalankan pada Delphi, maka secara otomatis akan dibaca sebagai sebuah program, tanpa dijalankan terpisah.
3. Mudah digunakan, source kode delphi yang merupakan turunan dari pascal, sehingga tidak diperlukan suatu penyesuaian lagi.
4. Bersifat multi purphase, artinya bahasa pemrograman Delphi dapat digunakan untuk mengembangkan berbagai keperluan pengembangan aplikasi.

c. Sejarah Borland Delphi

1. Delphi versi 1 (berjalan pada windows 3.1 atau windows 16 bit)
2. Delphi versi 2 (Berjalan pada windows 95 atau delphi 32 bit)
3. Delphi versi 3 (berjalan pada windows 95 keatas dengan tambahan fitur internet atau web)
4. Perkembangan selanjutnya diikuti dengan Delphi versi 4, 5 dan 6.
5. Versi terkini dari delphi adalah versi 7 dengan tambahan fitur .net dengan tambahan file XML

1.2. OOP (Object Oriented Programming)

OOP adalah metode pemrograman dengan membantu sebuah aplikasi yang mendekati keadaan dunia yang sesungguhnya. Hal itu bisa dilakukan dengan cara mendisain object untuk menyelesaikan masalah.

a. Tiga unsur OOP

1. Encapsulation atau pemodelan

Encapsulation adalah konsep penggabungan data dengan operator. Dalam konsep pemodelan data dan operasi menjadi satu kesatuan yang disebut object. *Encapsulation* juga disebut dengan penyembunyian informasi (*information hiding*)

Contoh = ketika kita menyalakan sebuah TV kita tidak tahu apa yang terjadi dengan proses dan percakapan antar alat yang berhubungan dengan TV tersebut sehingga mampu menampilkan sebuah gambar.

Atau = ketika melakukan klik pada sebuah object dalam suatu aplikasi program kita tidak tahu apa yang dilakukan program sehingga ditampilkan hasil yang sesuai.

Catatan = dari dua contoh kasus tersebut terdapat kesamaan proses mengenai *information hiding* yang tidak diketahui oleh user sampai hasil ditampilkan.

b. Inheritance atau penurunan

Inheritance adalah sebuah object yang dapat diturunkan menjadi object yang baru dengan tidak menghilangkan sifat asli dari object tersebut.

Contoh = TV merupakan salah satu media elektronik yang digunakan untuk menampilkan gambar dengan tujuan untuk memberikan informasi kepada konsumen. Secara umum TV mempunyai cara kerja yang sama dengan media elektronik yang lain dalam proses penyampaian informasi, tetapi mempunyai sifat yang unik yang dapat membedakan dengan media elektronik yang ada.

Atau = Dalam aplikasi program kita sering menggunakan command button, untuk beberapa perintah yang berbeda.

c. Polymorphism atau Polimorfisme

Polymorphism merupakan penggunaan berbagai macam object yang berbeda tetapi secara fungsi bergantung pada satu object sebagai induk, dengan cara pelaksanaan yang berbeda – beda.

Contoh = TV dan radio adalah media elektronik yang mempunyai sistem yang sama tentang bagaimana menyebarkan suatu informasi, tetapi cara kerja dari masing – masing sistem pasti berbeda.

Atau

Object Simpan dan Update adalah icon yang berasal dari induk yang sama yaitu , command button tetapi cara kerja tersebut berbeda – beda.

1.3. Delphi dan OOP (Object Oriented Programming)

Secara default ketika kita merancang suatu aplikasi program, mau tidak mau dan tanpa kita sadari bahwa kita telah menerapkan OOP, walaupun secara teori kita kurang memahami OOP dalam arti yang sebenarnya.

Contoh sederhana adalah ketika kita merancang suatu form (Tform1) baru, sadar atau tidak sebenarnya form yang kita aktifkan merupakan turunan dari Tform sebagai induknya atau ketika kita mengaktifkan button pada form merupakan turunan dari tbutton.

Atau

Contoh dalam bahasa program adalah sebagai berikut = ketika merancang suatu label di form secara otomatis delphi akan menuliskan label tersebut dalam jendela code editor tentang turunan dari label tersebut.

Type

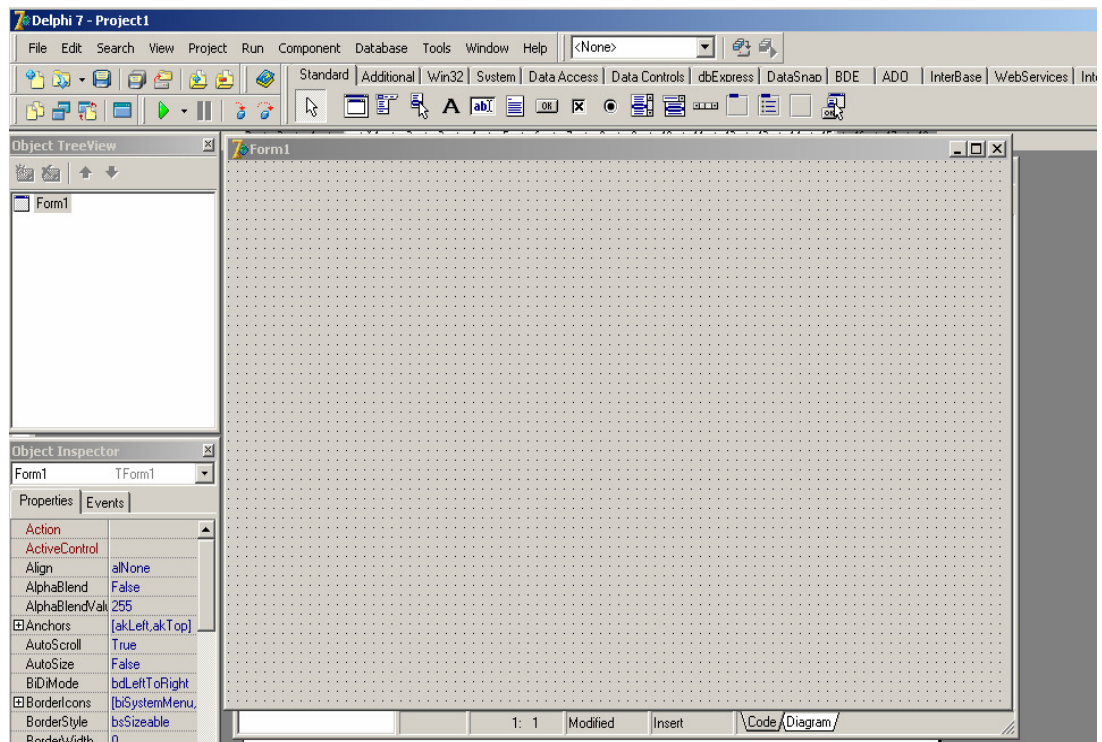
```
Tform = class(tform)
    Label1 = TLabel
End;
```

1.4. IDE DELPHI

a. langkah – langkah mengaktifkan Delphi

- a. Klik start
- b. pilih program Files
- c. pilih Borland Delphi
- d. pilih dan klik Delphi 7

b. Jendela Utama Delphi

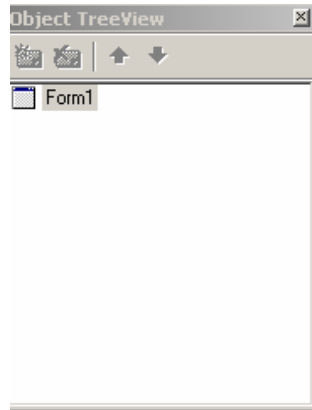


Gambar 1.1 Jendela Utama Delphi

c. Bagian – bagian dari Jendela Delphi

1. Object Tree View

Merupakan sebuah diagram pohon yang menggambarkan hubungan logis menghubungkan semua komponen yang terdapat dalam suatu proyek program. Komponen tersebut meliputi form, modul atau frame. Fungsinya digunakan untuk menampilkan seluruh daftar komponen program dalam sebuah aplikasi program sesuai dengan penempatannya.



Gambar 1.2 Jendela Object Tree View

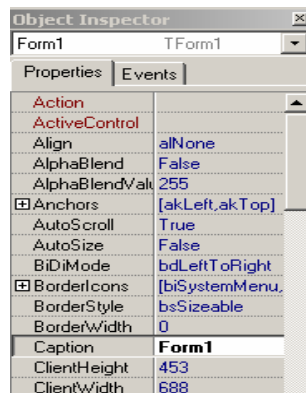
2. Object Inspector

Merupakan jendela yang digunakan untuk mengatur tampilan komponen pada form, misal bagaimana mengubah tulisan button pada command button menjadi Simpan, atau menghapus tulisan pada label dan mengganti nama menjadi Nama Mahasiswa atau memberikan perintah tertentu pada sebuah komponen sehingga ada interaksi ketika program dijalankan..

Secara Umum Object Inspector terbagi menjadi 2, yaitu =

a. Properties

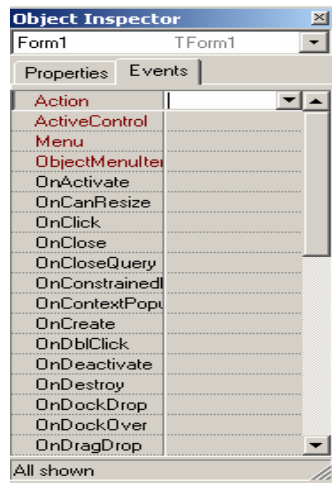
Digunakan untuk mengatur tampilan pada sebuah komponen baik itu meliputi penggantian nama, warna, jenis huruf, border dan lain –lain.



Gambar 1.3 Jendela Inspector
(properties)

b. Events

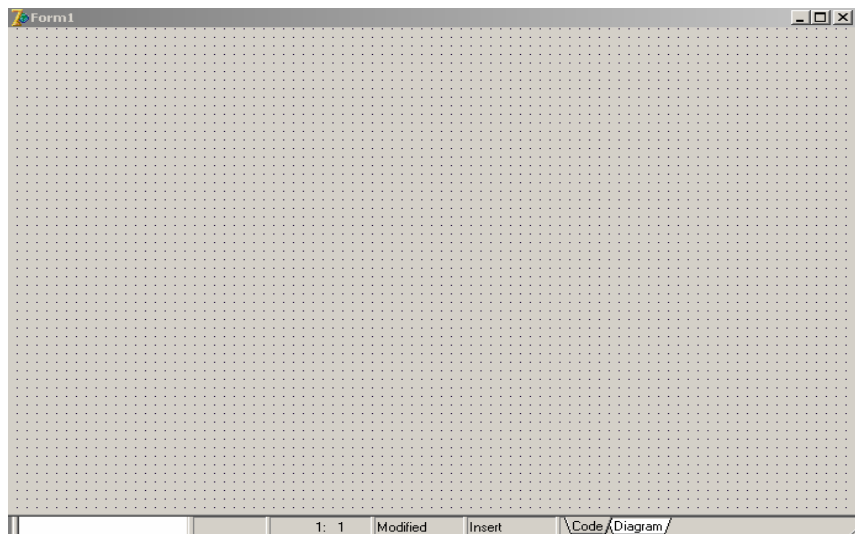
Merupakan jendela properties yang digunakan untuk memberikan fungsi yang lebih detail dari fungsi sebenarnya. Misalnya ketika tombol Simpan di klik maka program akan menjalankan perintah penyimpanan data. Dari kalimat tersebut ada event klik untuk mengeksekusi sebuah tombol simpan. Perintah event klik tersebut dapat diberikan melalui jendela events.



Gambar 1.4. Jendela Events pada Object Inspector

3. Form Designer

Merupakan tempat yang digunakan untuk merancang semua aplikasi program yang diambil dari komponen pallete.



Gambar 1.5. Jendela Form Designer

4. Component Palette

Merupakan kumpulan icon yang digunakan untuk merancang suatu aplikasi pada untuk membentuk sebuah aplikasi user interface.

Dalam komponen palette semua icon dikelompokkan dalam berbagai komponen sesuai dengan fungsi dan kegunaannya.

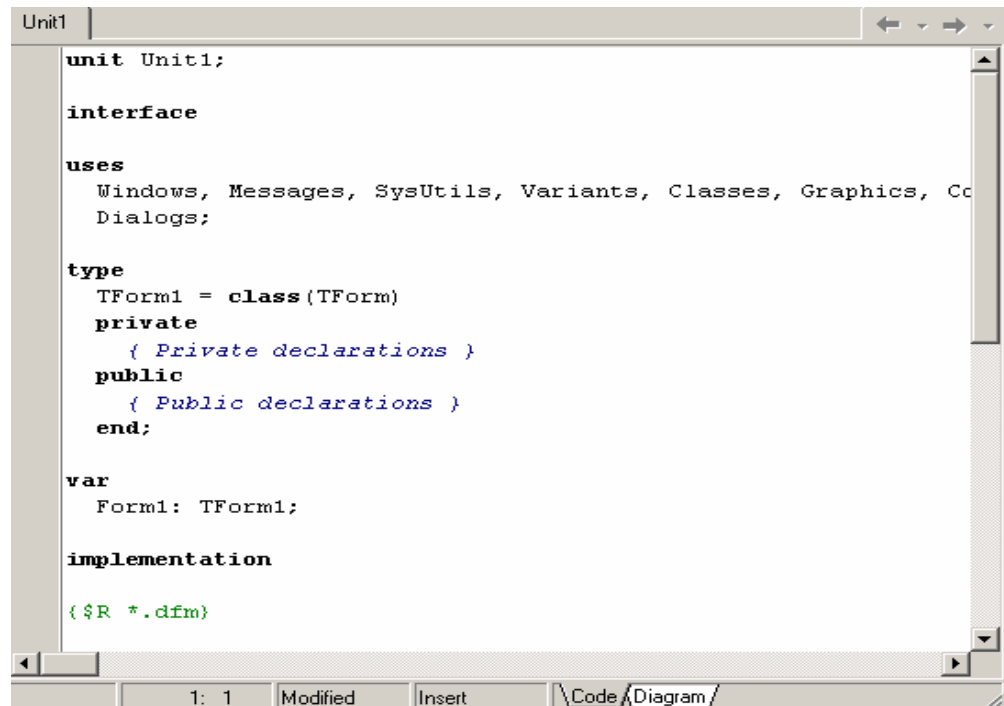


Gambar 1.6. Jendela Komponen Palette

No	Icon	Name	Fungsi
1		Pointer	Mengembalikan fungsi mouse ke defaultnya
2		Frame	Membentuk suatu frame terhadap obyek yang ada didalamnya
3		Main menu	Membuat menu Utama
4		Popup Menus	
5		label	Hanya untuk menampilkan Teks
6		Edit	Untuk menampilkan dan input data (1 baris)
7		Memo	Sama seperti edit tetapi mempunyai kapasitas lebih besar (lebih dari 1 baris)
8		Button	Digunakan untuk melakukan eksekusi terhadap suatu proses
9		Checkbox	Digunakan untuk menentukan pilihan lebih dari satu
10		Radio Button	Digunakan untuk menentukan pilhan, tetapi hanya satu pilhan yang bisa digunakan
11		List Box	Menmpilkan pilihan dalam bentuk list
12		Combo Box	Menampilkan pilihan dalam bentuk popup
13		Scroll Bar	Merupakan icon yang berupa baris status
14		Group Box	Digunakan untuk mengelompokkan suatu icon
15		Radio Group	Digunakan untuk mengelompokkan pilihan

5. Code Editor

Bagian dari delphi yang digunakan untuk menuliskan kode program. Pada bagian code editor terdapat 3 bagian utama yaitu = bagian *paling kiri* yang berisi berupa *angka* menunjukkan baris dan kolom. Keterangan *modified* menunjukkan bahwa telah terjadi modifikasi terhadap baris program. Dan paling kanan menunjukkan status keyboard tentang tombol *insert* atau *over write*.



Gambar 1.7 Jendela Code Editor

6. Code Explorer

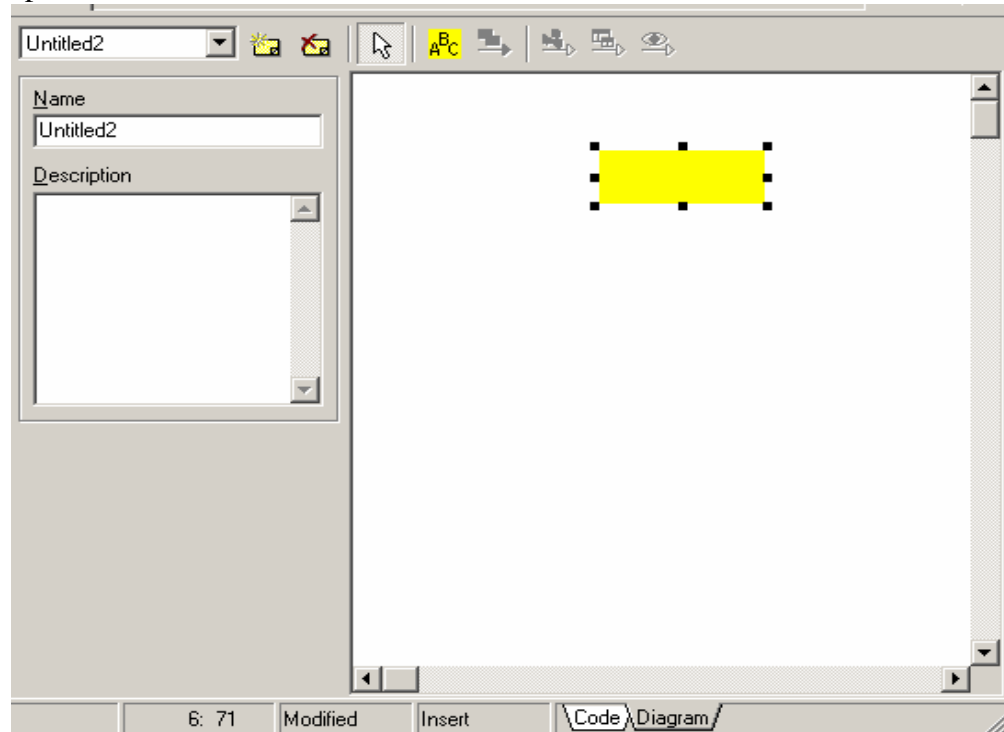
Jendela yang digunakan untuk menampilkan seluruh *variabel*, *type*, dan *routine* yang didefinisikan pada sebuah unit.



Gambar 1.8. Jendela Code Explorer

7. Code Diagram

Merupakan fasilitas pada delphi yang digunakan untuk mendesain sebuah diagram atas komponen – komponen yang digunakan dalam suatu rancangan aplikasi.



Gambar 1.9. Jendela Code Diagram

d. Proyek Delphi

1. File Proyek

File ini disimpan dengan ber-ekstensi **.dpr**. File ini berisi informasi mengenai seluruh proyek program

2. File Unit

File ini merupakan kumpulan dari barisan kode program yang terdapat di jendela code editor, baik itu yang dituliskan oleh programmer maupun oleh system. Extension file ini adalah **.pas**

File Unit dibagi menjadi 2

a. Bagian Interface

Barisan ini dimulai dari kata Interface (setelah nama unit), berisi seluruh deklarasi variabel, tipe data object maupun deklarasi tambahan.

b. Bagian Implementation

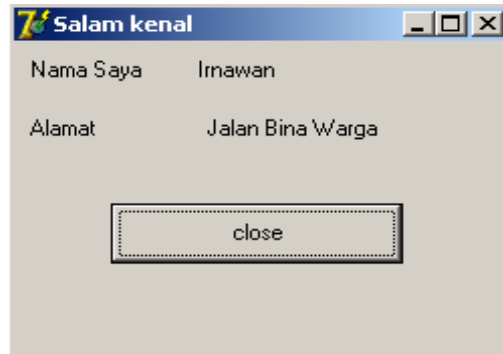
Dimulai dari kata kunci *implementation* dan diakhiri dengan kata *end*. Fungsi digunakan untuk menuliskan kode program sebagai bagian dari interaksi antar komponen ataupun dengan user.

3. File Form

Berisi tentang seluruh informasi yang ada kaitannya dengan form yang dibuat, meliputi tinggi, lebar, posisi form atau tentang komponen didalamnya. Penggunaan file ini tidak dianjurkan karena untuk pengaturan sudah disediakan *object inspector* sebagai media pengaturan semua komponen.

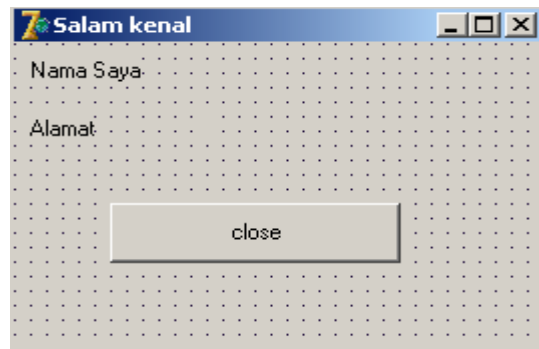
1.5. Aplikasi Salam Kenal (LATIHAN 1)

a. Hasil Program



Gambar 1.10 Gambar Form Salam kenal

b. Desain Form



Gambar 1.20 Gambar Desain Form Salam Kenal

c. Desain Properties

Object	Name	Caption
Label 1	Label 1	Nama Saya
Label 2	Label 2	Alamat
Label 3	Lnama	-
Label 4	Lalamat	-
Button1	Bclose	Close
Form 1	Form1	Salam Kenal

d. Listing program

1. Ketika form dalam keadaan aktif maka form akan menampilkan nama dan alamat pembuat program

```
procedure TForm1.FormCreate(Sender: TObject);  
begin  
    nama.Caption := 'Irnawan';  
    alamat.Caption := 'Jalan Bina Warga';  
end;
```

2. Untuk keluar dari program user mengklik buton close

```
procedure TForm1.TcloseClick(Sender: TObject);  
begin  
    close;  
end;  
end.
```

Bab II

Operator, Deklarasi, Tipe Data dan Mengenal Object I

2.1 Deklarasi

Dalam setiap penulisan bahasa pemrograman deklarasi sangat digunakan apabila dalam penulisan program dibutuhkan indentifier atau tanda pengenal. Indentifier pada umumnya di buat oleh programmer yang digunakan untuk mewakili nilai dari suatu object.

Indentifier yang dikenal dalam Delphi adalah label, konstanta, tipe, fungsi, procedure maupun variabel.

2.1.1. Deklarasi Konstanta

Deklarasi konstanta adalah tanda pengenal dalam Delphi yang mempunyai nilai yang sudah tetap. Definisi konstanta diawali dengan kata baku **Const** diikuti dengan kumpulan indentifier yang diberi sebuah nilai.

Contoh

```
procedure TForm2.etertulisChange(Sender: TObject);
const
  nil1:='30000';
begin
end;
```

2.1.2. Deklarasi Variabel

Deklarasi variabel adalah tanda pengenal dalam Delphi yang mempunyai nilai yang mana nilai tersebut akan terus berubah selama proses berjalan. Definisi variabel diawali dengan kata baku **Var** diikuti dengan kumpulan indentifier yang diikuti dengan tipe data yang dibutuhkan.

Contoh

```
procedure TForm2.EpraktekKeyPress(Sender: TObject; var Key: Char);
var
  praktek,nil2,nil1 :real;
begin
  if (key = #13) then
  begin
    nil1 := strtoint(ehtulis.Text);
```

```

praktek:= strtofloat(epraktek.Text);
nil2:= 0.4 * praktek;
ehpraktek.Text := floattostr(nil2);
form2.ActiveControl := cmi;
emurni.Text := floattostr(nil1 + nil2);
if nil1 > 60 then
    egrade.Text := 'Lulus'
else
    egrade.Text := 'Gagal'
end;
end;

```

2.2 Tipe Data

Secara sederhana tipe data dapat didefinisikan dengan istilah tempat untuk menentukan pemberian nilai terhadap suatu variabel sesuai atau tidak dengan nilai yang diberikan oleh user. Dalam versi lain tipe data juga diartikan sebagai batasan terhadap fungsi tanda pengenal terhadap semua nilai yang diterima. logika yang dapat kita berikan adalah ketika kita menempatkan tanda pengenal **harga** hanya mengenal angka, maka ketika kita memberikan nilai berupa string maka secara otomatis data tersebut akan ditolak karena nilai tersebut tidak dikenali oleh **tipe data** yang diberikan.

2.2.1 Tipe Data Numeric Integer

Tipe data integer merupakan tipe data bilangan bulat yang hanya mengenal bilangan decimal. Dimana tipe data **Integer** tidak mengenal pecahan

Bentuk Umum

```

Var
    Nil1:integer;
Begin
    Nil1:=5000;

```

2.2.2 Tipe Data Real

Tipe data numeric real adalah tipe data dari suatu tanda pengenal selain mengenal bilangan bulat utuh tipe data ini juga mengenal nilai angka yang mengenal pecahan.

Bentuk Umum

Var

Nil:real;

Begin

Nil1:=20,5;

2.2.3 Tipe Data String

Tipe data string merupakan salah satu jenis tipe data selain mengenal angka disini tipe data dapat juga mengenal data berupa huruf maupun tanda baca.

Bentuk umum

Var

Nama:string;

Begin

Nama:='Anton';




2.2.4 Tipe Data Char

Secara fungsi tipe data char sama dengan tipe data string tetapi dari segi kapasitas ruang diperoleh tipe data char jauh lebih sedikit karena hanya mengenal 1 karakter.

2.3. Dasar umum merancang Program aplikasi berbasis visual

- Merancang tampilan program (user interface) hal ini meliputi = Form dan toolbox
- Desain properties. Hal ini digunakan untuk merubah tampilan icon yang asli toolbox agar sesuai dengan tampilan yang diinginkan.
- Jendela Code Editor , digunakan sebagai media komunikasi antar object pada form dengan system yang ada.

2.4. Mengenal Komponen label, edit & Button

	Label	Hanya untuk menampilkan Teks
	Edit	Untuk menampilkan dan input data (1 baris)
	Button	Digunakan untuk melakukan eksekusi terhadap suatu proses

2.5. Mengetahui Operator

Delphi mengenal banyak operator, sama seperti bahasa pemrograman yang lain, operator menjadi satu hal penting yang harus ada untuk perancangan program.

2.5.1. Operator Penugasan (assignment Operator)

Symbol operator digunakan untuk melakukan suatu proses atas suatu nilai dengan memberikan nilai baru pada suatu variabel

Lambang operator Penugasan “ := ”

Bentuk Umum penulisan

Var := perintah;

Contoh =

A:=”B”;

Label1.caption := “Irnawan”

C:= A + B;

2.5.2. Operator Aritmatika

Operator aritmatika berfungsi untuk melakukan suatu proses aritmatika yang meliputi perkalian, pembagian, penjumlahan, pengurangan maupun pengurangan terhadap suatu nilai variabel yang tersimpan dalam suatu object, dengan memberikan nilai baru.

Symbol yang dikenal dalam delphi

Symbol	Keterangan
+	Penjumlahan
-	Pengurangan
*	Perkalian
/	Pembagian
Div	Pembagian Integer
Mod	Sisa Pembagian

Contoh

B := 15 + 2;	hasil	B = 17
B := 15 div 2		B = 7
B := 15/2		B = 7,5
B := 15 * 2		B = 30
B := 15 mod 2		B = 1

2.5.3. Operator String

Digunakan untuk menggabungkan dua teks/string atau lebih.

Symbol yang digunakan = +

Contoh =

Bentuk Umum

A := teks1 + teks2

Contoh

A := "Aku";

B := "Bisa";

C := A + B;

Hasil C = Aku Bisa

2.5.4. Merancang Aplikasi Dengan Delphi

a. Hasil ketika form dijalankan

Luas Persegi panjang

Program pengenalan Dasar operator Delphi

Nilai Praktek I: 5

Nilai Praktek II: 3

Nilai Perhitungan

Tambah	hasil	8
Kali	hasil	15
Bagi	hasil	1,6666666666666666
Kurang	Hasil	2

Close

Gambar 2.1 Form Operator

b. Desain Form

Gambar 2.2 Desain Form Operator

c. Desain Properties

Object	Name	Caption
Label 1	Label 1	Program pengenalan dasar operator delphi
Label 2	Label 2	Nilai Praktek I
Label 3	Label3	Nilai Praktek II
Label 4	Label4	Nilai Perhitungan
Label5	Label5	Hasil
Label6	Label6	Hasil
Label7	Label7	Hasil
Label8	Label8	Hasil

Object	Name	Text
Edit1	Nil1	-
Edit2	Nil2	-
Edit3	Ehtambah	-
Edit4	ehkali	-
Edit5	Ehbagi	-
Edit6	ehkurang	-

Object	Name	Caption
Button1	Btambah	Tambah
Button2	Bkali	Kali
Button3	Bbagi	Bagi
Button4	Bkurang	Kurang
Button5	Bclose	Close

d. Listing Program

Catatan =

Dalam bahasa pemrograman Delphi semua data yang diinput melalui edit box dideklarasikan dengan tipe data string, sehingga ketika kita akan melakukan proses perkalian atau pengurangan kita tidak bisa mengalikan secara langsung edit box yang dimaksud tetap kita perlu merubah menjadi nilai dengan tipe data yang dapat dikalikan.

1. Program Tambah

Digunakan untuk menampilkan hasil penjumlahan antara nilai praktek 1 dengan nilai praktek II, cara double klik pada buton tambah dan tuliskan kode berikut ini =

```
procedure TForm2.BtambahClick(Sender: TObject);
var
    nil1, nil2 : real;
    tambah: real;
begin
    nil1 := strtofloat(enil1.text);
    nil2 := strtofloat(enil2.text);
    tambah := nil1 + nil2;
    ehtambah.text := floattostr(tambah);
end;
```

Catatan

Var

```
    Nil1,nil2 : real;
    Tambah ; real;
```

Var digunakan untuk mendeklarasikan suatu nama variabel yang dibuat oleh programmer yang berfungsi untuk menyimpan nilai atau data selama proses program berjalan.

Nil1,nil2 dan **tambah** adalah nama variabel baru yang dibentuk oleh programmer

Real adalah nama dari sekian banyak tipe data yang dikenal dalam bahasa pemrograman Delphi. Cakupan tipe data real adalah membaca bilangan angka dalam bentuk decimal.

Strtfloat adalah fungsi yang digunakan merubah nilai dari tipe data string menjadi tipe data numeric.

Floattostr adalah fungsi yang digunakan merubah nilai pecahan menjadi data string.

2. Program untuk kali

```
procedure TForm2.BkaliClick(Sender: TObject);
var
    bil1,bil2:real;
    kali : real;
begin
    bil1 := strtfloat(enil1.text);
    bil2 := strtfloat(enil2.text);
    kali := bil1*bil2;
    ehkali.text := floattostr(kali);
    {ehkali.text      :=      inttostr(strtoint(enil1.text)      *
    strtoint(enil2.text));}
end;
```

3 Program untuk bagi

```
procedure TForm2.BbagiClick(Sender: TObject);
var
    bil1,bil2 : real;
    bagi : real;
begin
    bil1 := strtfloat(enil1.Text);
    bil2 := strtfloat(enil2.Text);
    bagi := bil1/bil2;
    ehbagi.Text := floattostr(bagi);
end;
```

4. Program untuk kurang

```
procedure TForm2.BkurangClick(Sender: TObject);
var
    bil1,bil2,kurang : real;
begin
    bil1 := strtfloat(enil1.Text);
```

```

bil2 := strtofloat(enil2.Text);
kurang := bil1 - bil2;
ekurang.Text := floattostr(kurang);
end;

```

5. Program untuk keluar

```

procedure TForm2.bcloseClick(Sender: TObject);
begin
messageDlg('Ingin Menutup form',mterror,[mbok],0);
close;
end;

```

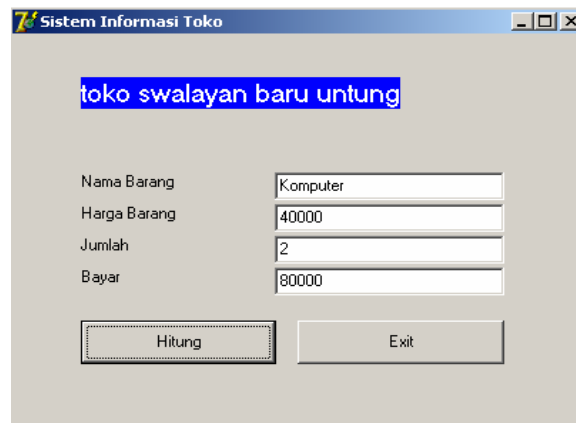
Catatan

MessageDlg adalah penggalan program yang digunakan untuk menampilkan kotak pesan.

Close digunakan untuk menutup form.

2.5.5. Latihan Program Pembayaran

a. Form setelah dijalankan






Gambar 2.3 Gambar Toko

b. Ketentuan yang diinginkan pemilik toko

1. Nama barang, harga barang dan jumlah merupakan media input bagi user mengenai detail nama barang, harga barang serta jumlah yang dibeli.
2. Bayar digunakan untuk menampilkan perkalian dari jumlah yang dibeli dengan harga barang, dengan cara mengklik hitung.
3. Untuk menutup form user tinggal mengklik exit, dengan menampilkan pesan untuk user.

Catatan :

1. Untuk Menambah Form Baru pilih dan klik icon **New Form** 
2. Untuk Mengaktifkan Form Yang Pernah Dibuat adalah pilih dan klik **View Form** , pada jendela View Form pilih form yang ingin diaktifkan dan klik **OK**
3. Untuk Menjalankan Form yang diinginkan dari beberapa form yang telah dibuat adalah dengan cara pilih **Projectl option**. Pada combo **Project Option**, pilih dan klik form yang diinginkan dengan mengaktifkan Combo Main form. Setelah selesai klik **OK**.
4. Untuk menyimpan semua project pilih dan klik **Save All**. 

BAB III

Percabangan dan Mengenal Object II (combo box, radio button)

3.1. Operator Percabangan

percabangan adalah merupakan operator yang digunakan untuk menentukan pilihan terhadap beberapa pilihan yang ada.

Dalam bahasa pemograman Delphi mengenal dua operator per cabangan

3.1.1. Percabangan If

merupakan operator percabangan yang digunakan untuk menentukan pilihan atas beberapa kondisi yang merupakan syarat terhadap pilihan yang sudah ditentukan.

Ada dua model percabangan if

a. Percabangan untuk kondisi pilihan tunggal

Merupakan operator percabangan yang digunakan untuk menentukan sebuah pilihan dengan kondisi tunggal

Bentuk Umum

If Syarat **then** hasil;

Contoh

If Nilai > 80 **then** keterangan = 'Lulus';

b. Percabangan untuk kondisi majemuk

Merupakan operator percabangan yang digunakan untuk menentukan pilihan dengan kondisi yang harus dipeuhi lebih dari satu.

Bentuk Umum

If Syarat1 **then**

 Hasil1

Else

If syarat2 **then**

 Hasil2

Else

.....

.....

end;

Contoh

If nilai > 80 **then**

 Grade = "A"

Else

if nilai > 70 **then**

 grade = "B"

else

if nilai > 60 **then**

 grade = "C"

else

 grade = "E";

3.1.3. Percabangan case

case of adalah merupakan metode lain dari sebuah percabangan, berfungsi sama seperti fungsi **if** yaitu untuk melakukan seleksi atas beberapa pilihan dengan kondisi sebagai syarat yang harus terpenuhi. Secara fungsi **case** dan **if** tidak ada perbedaan tetapi untuk penulisan fungsi **case** lebih mudah diterapkn untuk pilihan atau kondisi lebih dari satu.

Bentuk Umum fungsi Sace OF

Case <variabel> **of**
 <pilihan ke 1> ; hasil1;
 <pilihan ke 2>; hasil2;
 <pilihan ke 3>;hasil3;

 <pilihan ke n>; hasiln;
end;

atau fungsi **case of** bisa juga diberikan **else** untuk piliahn terakhir.

Bentuk Umum

Case <variabel> **of**
 <pilihan ke 1> ; hasil1;
 <pilihan ke 2>; hasil2;
 <pilihan ke 3>;hasil3;





 else
 hasiln;
end;

Contoh

Case Bilangan of

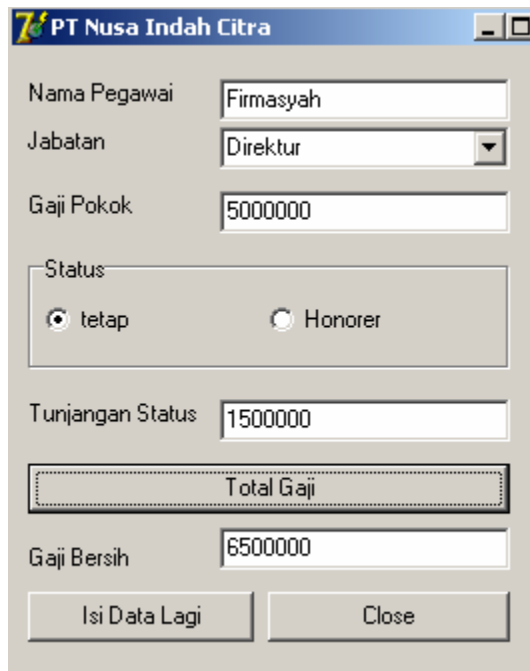
```
1:showmessage('Angka 1');  
2:showmessage('Angka 2');  
3:showmessage('Angka 3');  
end;
```

3.2. Mengenal Object II (combo box, list box dan radio button)

1		Checkbox	Digunakan untuk menentukan pilihan lebih dari satu
2		Radio Button	Digunakan untuk menentukan pilhan, tetapi hanya satu pilhan yang bisa digunakan
3		List Box	Menmpilkan pilihan dalam bentuk list
4		Combo Box	Menampilkan pilihan dalam bentuk popup

3.3. Merancang Program dengan fungsi IF

a. Hasil Yang Diperoleh



PT Nusa Indah Citra

Nama Pegawai: Firmasyah

Jabatan: Direktur

Gaji Pokok: 5000000

Status: ☒ tetap ☐ Honorar

Tunjangan Status: 1500000

Total Gaji: 6500000

Gaji Bersih: 6500000

Isi Data Lagi Close

Gambar 3.1 Gambar Perhitungan Gaji

Ketentuan Perhitungan Gaji

1. Dalam Struktur organisasi perusahaan ketentuan gaji perusahaan terbagi menjadi 3 struktur utama

Jabatan	Gaji Pokok
Direktur	5000000
Manager	3000000
Karyawan	1000000

2. Besarnya tunjangan ditentukan oleh status kep

b. Desain Form



Gambar 3.2. Desain Form Gaji

c. Desain Properties

Object	Name	Caption
Label 1	Label1	Nama Pegawai
Label 2	Label2	Jabatan
Label 3	Label3	Gaji Pokok
Label 4	Label4	Tunjangan Status
Label5	Label5	Gaji Bersih
GroupBox	GroupBox1	Status
RadioButton1	Rtetap	Tetap
RadioButton2	Rhonorar	Honorar
Object	Name	Text

Edit1	enama	-
Edit2	epokok	-
Edit3	etunjangan	-
Edit4	ebersih	-
Combo Box	Cjabatan	-

Object	Name	Caption
Button1	Btotal	Total Gaji
Button2	Blagi	Isi Data Lagi
Button3	Bclose	Close

d. Listing Program

- Memberikan pilihan pada combo box sesuai dengan pilihan pada jabatan

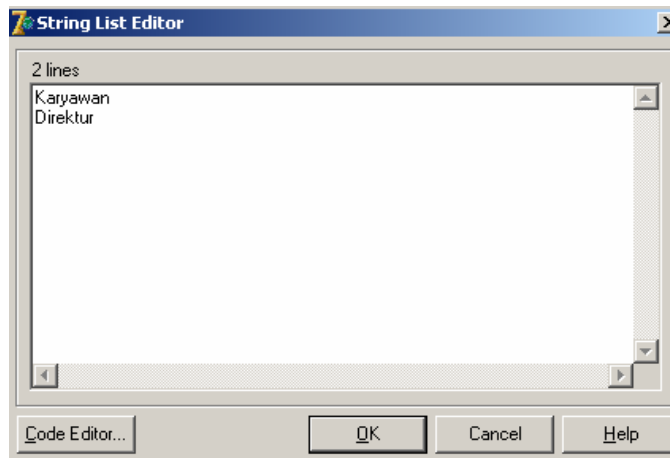
Dengan Menggunakan Object Properties

Langkah – langkahnya

1. aktifkan combo box yang akan dipilih
2. pada properties pilih dan klik item, lalu klik command (...)



3. Pada jendela **String list Editor**, Seperti terlihat pada jendela berikut



4. Ketikan kata sebagai kata pilihan pada jendela seperti contoh tersebut diatas.
5. Setelah selesai klik OK

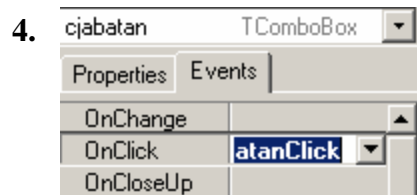
Melalui jendela code editor

Double click pada form, sebarang tempat dan ketikan program berikut ini.

```
procedure TForm1.FormCreate(Sender: TObject);
begin
  cjabatan.Items.Add('Direktur');
  cjabatan.Items.Add('Manager');
  cjabatan.Items.Add('Karyawan');
end;
```

➤ Program untuk mencari gaji pokok

1. Aktifkan combo box untuk jabatan
2. Pada pada jendela properties pilih **event**,
3. pilih dan aktifkan **onclick**, Seperti yang terlihat seperti gambar berikut ini.



5. Double click pada combo **onclick**.
6. Ketikan program berikut ini

```
procedure TForm3.cjabatanClick(Sender: TObject);
begin
  if cjabatan.Text = 'Direktur' then
    epokok.Text := '5000000'
  else
    if cjabatan.Text = 'Manager' then
      epokok.Text := '2000000'
    else
      if cjabatan.Text = 'Karyawan' then
        epokok.Text := '1000000'
      else
        epokok.Text := '0'
  end;
```

- **Program menghitung Tunjangan Karyawan Tetap**
Double click pada option tetap dan ketikan program berikut ini

```
procedure TForm3.etetapClick(Sender: TObject);
begin
if etetap.Checked = true then
    a := strtofloat(epokok.Text);
    hasil :=a*0.3;
    etunjangan.Text := floattostr(hasil)
end;
```

- **Program menghitung Tunjangan Karyawan Honorer**
Double click pada option honorer dan ketikan program berikut ini

```
procedure TForm3.rhonorerClick(Sender: TObject);
begin
if rhonorer.Checked = true then
    a := strtofloat(epokok.Text);
    hasil :=a*0.1;
    etunjangan.Text := floattostr(hasil)

end;
```

- **Program menghitung Gaji Bersih**
Double click pada command **total gaji**, dan ketikan program berikut ini :

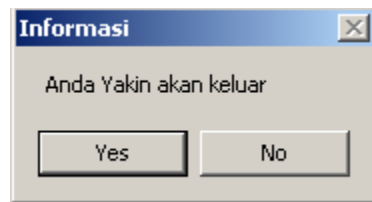
```
procedure TForm3.bttotalClick(Sender: TObject);
begin
a := strtofloat(epokok.Text);
b := strtofloat(etunjangan.Text);
c := a+b;
ebersih.Text := floattostr(c);
end;
```

➤ **Program bersih**

Double click pada **Isi Data Lagi**, dan ketikkan program berikut ini

```
procedure TForm3.blagiClick(Sender: TObject);
begin
  cjabatan.Text := 'Jabatan';
  enama.Text := '';
  ebersih.Text := '0';
  etunjangan.Text := '0';
  epokok.Text := '0';
  etetap.Checked := false;
  rhonorer.Checked := false;
end;
```

➤ **Program Close dengan pesan, seperti tampilan sebagai berikut.**



Gambar 3.3 Form Pesan

Programnya

```
procedure TForm3.bcloseClick(Sender: TObject);
begin
  if (application.MessageBox('Anda Yakin akan
keluar','Informasi',MB_YESNO)= IDYES)then
    close
end;
```

3.4. Program Biaya Ujian

Buatlah program untuk mencari biaya ujian untuk calon siswa pada suatu perguruan tinggi.

Tampilan yang diinginkan adalah sebagai berikut

lembar Penilaian Ujian

**PENILAIAN UJIAN SARINGAN MASUK
AKADEMI BINTANG TERANG**

UJIAN TERTULIS	89	× 40 %	35,6
UJIAN PRAKTEK	56	× 60 %	22,4

Nilai murni Siswa: 58 Grade: Gagal

Pilihan Jurusan:

<input checked="" type="checkbox"/> MI	Biaya Jurusan MI	250000
<input checked="" type="checkbox"/> TK	Biaya Jurusan TK	300000

Pilihlah Kuliah:

☒ Pagi / Siang Biaya waktu: 500000

☐ Sore / Malam

Total Biaya: 1050000

ISI DATA LAGI CLOSE

Gambar 3.4 Form Penilaian Siswa

Ketentuan Pencarian Biaya

1. Nilai murni didapat dari penjumlahan (nilai Prkatek * 60%) ditambah (Nilai Teori * 40 %).
2. Keterangan = Gagal jika nilai murni < 60 dan dinyatakan lulus jika nilai murni >= 60
3. Biaya perjurusan diperoleh dari dua pilihan jurusan yang ada:
 1. jika pilhan MI maka biayanya = 250000
 2. jika pilihan TK maka biaya = 300000
4. Pilihan Biaya Waktu Ujian diperoleh dari 2 waktu kuliah yang ada.
Jika pilihan pagi maka biaya kuliah = 50000, 1000000 untuk biaya kuliah malam.
5. Total biaya diperoleh dari penjumlahan Semua jurusan ditambah biaya waktu kuliah.
6. Isi data lagi untuk mengulang input data
7. close untuk menutup program.

BAB IV

Prosedur , Perulangan dan Megenal Input Box

4.1. Prosedur

Prosedur adalah suatu program terpisah dan berdiri dalam suatu blok program dan berfungsi sebagai sebuah sub program (program bagian). Penulisan prosedur diawali dengan kata ***Procedure*** pada bagian deklarasi program dan cukup menuliskan nama prosedur yang dibuat pada bagian ***Implementasi***.

Alasan penggunaan prosedur

1. Digunakan untuk penggalan program yang akan digunakan secara berulang – ulang dalam suatu proses program.
2. Digunakan untuk memmeca – mecah program menjadi sebuah modul program, sehingga listing program menjadi lebih sederhana.

Syarat penulisan nama procedure

1. harus diawali dengan karakter.
2. untuk nama prosedur dengan menggunakan dua kata atau lebih penulisannya tidak boleh menggunakan spasi, harus digabung atau dihubungkan dengan underscore (_)
3. tidak mengenal tanda baca.

Contoh

a. Pendeklarasian Prosedur

Penulisan pendeklarian sebuah prosedur dalam Delphi ditempatkan setelah kata ***public*** agar dapat dikenali oleh semua object yang ada dalam proses program tersebut. Penulisannya diawali dengan kata ***Procedure*** diikuti nama procedure.

Public

Procedure bersih;

b. Penulisan prosedur

Penulisan pogram prosedur terdapat dalam bagian ***implementation***,

Procedure TForm1.Bersih;

Begin

Enama.text := ";

Ealamat.text := ";

Ekota := ";

End;

c. Pemanggilan Prosedur

Pemanggilan prosedur terdapat dalam bagian implementation, pada umumnya pemanggilan prosedur terdapat dalam suatu ruang lingkup object yang mempunyai sebuah event. Cara pemanggilannya cukup hanya dituliskan nama prosedurnya saja.

```
Procedure TForm1.button1click(sender: TObject);  
Begin  
    Bersih;  
End;
```

atau ketika program prosedur digunakan pada object yang lain untuk memanggil program yang sama kita hanya cukup menuliskan nama prosedurnya tanpa dibutuhkan pendeklarasian prosedur baru.

```
Procedure TForm1.enamakeypress(sender: TObject; var key:char);  
Begin  
    If key = #13 then  
        Bersih;  
End;
```

4.2. Perulangan

Merupakan control program yang digunakan untuk suatu proses yang akan berjalan terus menerus. Kondisi perulangan merupakan proses berjalannya program secara terus menerus dan akan berhenti ketika proses mendapatkan kondisi yang sudah ditentukan.

Di dalam Delphi mengenal 3 jenis perulangan

a. Perulangan For – To - Do

Perulangan dengan statement for adalah perulangan yang digunakan untuk melakukan suatu proses dalam sebuah blok program. Proses perulangan For – To – Do dimulai dengan nilai terkecil ke besar.

Bentuk Umum

For variabel := nilai awal to nilaiakhir statement

Catatan : semua variabel yang berhubungan dengan perulangan harus mempunyai tipe data sama.

Contoh

```
procedure TForm4.Button1Click(Sender: TObject);
var
  i:integer;
begin
  for i := 1 to 5 do
    edit1.SelText := inttostr(i);
  end;
end.
```

Bila Program tersebut dijalankan maka nilai i akan ditampilkan pada edit1 dengan hasil sebagai berikut =



12345

b. Perulangan For – DownTo-Do

Perulangan For-DownTo-Do adalah perulangan yang menghitung suatu proses dengan nilai awal besar dan nilai akhirnya lebih kecil, maka variabel sebagai control program yang diperoleh adalah dari besar ke kecil.

Bentuk Umum

For Variabel := nilai-awal Downto Nilai-akhir Do Pernyataan

Contoh

```
procedure TForm4.Button2Click(Sender: TObject);
var
  i : integer;
begin
  for i := 5 Downto 1 do
    edit2.SelText := inttostr(i);
  end;
```

Bila program tersebut dijalankan maka hasil yang diperoleh adalah sebagai berikut ;



54321

c. Perulangan While Do

Perulangan **While Do** adalah statement perulangan akan terus melakukan suatu proses selama kondisi/syarat yang ditentukan bernilai benar.

Bentuk Umum

Statement **While – Variabel Syarat – Do Pernyataan**

Contoh

```
procedure TForm4.Button3Click(Sender: TObject);
var
  i : integer;
begin
  i := 0;
  while i < 5 do
  begin
    edit3.SelText := inttostr(i);
    i := i + 1;
  end;
end;
```

Bila program tersebut dijalankan maka hasil yang akan diperoleh adalah sebagai berikut :

A screenshot of a text box with a black border and a light gray background. The text '01234' is displayed in a black, monospaced font.

d. Perulangan Repeat Until

Perulangan **repeat until** digunakan untuk mengulang suatu kondisi sampai (until) kondisi bernilai salah.

Bentuk Umum

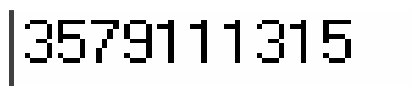
Repeat – Statement/penyataan — Until --- ungkapan logika

Contoh

```
procedure TForm4.Button4Click(Sender: TObject);
var
  i : integer;
begin
  i := 1;
  repeat
    i := i + 2;
    edit4.SelText := inttostr(i);
  until i = 15;

end;
```

Bila program tersebut dijalankan maka akan diperoleh hasil sebagai berikut :



4.3. Mengenal Input Box

Input Box adalah sebuah kotak pesan bagi user. Selain itu juga bisa digunakan untuk menginput data.

Bentuk umum

```
Var
Identifier : typedata;
begin
Identifier := inputbox('string_title','string_subtitle','');
end;
```

Contoh

```
Var
Inputnama : string;
Begin
Inputnama := inputbox('Info','Ketikan Nama','');
End;
```

4.4. Study Kasus Perulangan

a. Hasil Setelah Dijalankan

The screenshot shows a Java Swing window titled "Penjualan". It contains several input fields and buttons. At the top left, there is a section "Daftar Penjualan" with a label "Total Item yang Dibeli" and a text box containing the number "2", followed by an "OK" button. To the right of this is a "Tanggal" field with the text "Hari ini06-09-2005" and a "No Faktur" field. Below the "Daftar Penjualan" section is a list box titled "Daftar Barang Yang Dibeli" containing the items "Gatsby" and "Biore Body Form". To the right of this list box is another list box titled "Daftar Harga Barang" containing the prices "4500" and "5000". At the bottom left, there is a "Total Penjualan" label and a text box showing "9500". At the bottom of the window are two buttons: "ISI DATA LAGI" and "CLOSE".

Gambar 4.1 Form Penjualan

Ketentuan Dan Alur Logika

1. Total item digunakan sebagai pembatas berapa kali, pelanggan akan membeli barang.
2. Ketika jumlah item diberikan nilai maka ketika kita klik OK akan ditampilkan kotak pesan (input Box) untuk menginput nama barang dan harga barang yang dibeli. Begitu seterusnya sampai kondisi dari total item terpenuhi.
3. Total Penjualan diperoleh dari penjumlahan seluruh harga barang yang dibeli.
4. isi data lagi diberikan untuk mengulang pembelian.
5. Close untuk menutup form.
6. Untuk tanggal ditampilkan secara otomatis.

b. Desain Form

Gambar 4.2 Desain Form Penjualan

c. Desain Properties

Object	Caption	Name
Label 1	Total Item yang dibeli	Label1
Label 2	Tanggal	Label2
Label 3	NO Faktur	Label3
Label 4	Total Penjualan	Label4
GroupBox1	Data Penjualan	Groupbox 1
GroupBox2	Daftar Nama Barang Yang Debeli	Groupbox2
Groupbox3	Daftar Harga Barang	Groupbox3
Command1	OK	Cmdok
Command2	Isi Data Lagi	Cmdlagi
Command3	Close	cmdclose

Object	Text	Name
Edit1	-	eitem
Edit2	-	enofaktur
Edit3	-	etgl
Edit4	-	Ettotal
ListBox1	-	lnama
Listbox2	-	lharga

d. Listing Program

Program untuk menampilkan tanggal secara otomatis ketika form dijalankan

```
procedure TForm5.FormCreate(Sender: TObject);
begin
  DateSeparator := '-';
  ShortDateFormat := 'mm/dd/yyyy';
  etanggal.Text := 'Hari ini' + DateToStr(Date);
end;
```

Program command OK

```
procedure TForm5.bokClick(Sender: TObject);
var
  inputnama : string;
  inputharga : string;
  i : integer;
  j : integer;
  harga : real;
begin
  harga := 0;
  j := 0;
  j:= strtoint(eitem.Text);
  if j <= 0 then
  begin
    showmessage('Data Tidak boleh lebih kecil dari Nol');
    exit;
  end
  else
  begin
    for i := 1 to j do
    begin
      inputnama := inputbox('INput', 'Ketikan Nama Barang', '');
      inputharga := inputbox('INput', 'Ketikan Harga Barang', '');
      lnama.Items.Add(inputnama);
      lharga.Items.Add(inputharga);
      harga := harga + strttoFloat(inputharga);
    end;
  end;
end;
```

```
etotal.Text := floattostr(harga);
end;
```

Program Untuk Isi Data Lagi

```
procedure TForm5.clagiClick(Sender: TObject);
begin
  eitem.Text := '';
  enofaktur.Text := '';
  Inama.Clear;
  lharga.Clear;
  etotal.Text := '';
end;
```

Program Untuk Menutup Form

Close;

4.5. Aplikasi Konversi Suhu

a. hasil yang Diinginkan

Gambar 4.4 Hasil Form Penjualan

b. Ketentuan Program

1. Batas Awal dan Batas akhir diberikan untuk mengetahui nilai awal dan akhir derajat celcius yang ingin cetak.

2. Penambahan diberikan untuk memberikan penambahan dari nilai awal untuk mencapai nilai akhir.
3. Close adalah untuk keluar dari program
4. Rumus Mencari Fahrenheit adalah $= 1.8 * \text{Celcius} + 32$

BAB V

FUNGSI

5.1 Pengertian Fungsi

Pengertian secara umum mengenai fungsi adalah merupakan sebuah penggalan program yang terpisah dari program utama dan berfungsi sebagai sebuah program bagian dari program utama. Penulisan fungsi diawali dengan kata cadangan *function* dan dideklarasikan dalam bagian deklarasi fungsi. Dan penulisan program fungsi ditempatkan pada program utama. Satu hal yang perlu diperhatikan dalam penulisan fungsi adalah harus diikuti dengan tipe datanya.

Bentuk Umum penulisan Fungsi

Function indentifier(daftar-parameter) : type;

Contoh

Function hitung(var a,b : real) :real;

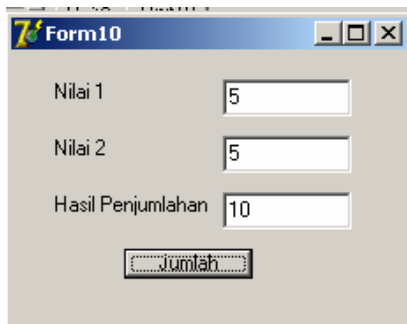
Penulisan blok fungsi diawali dengan kata cadangan *begin* dan diakhiri dengan *end*;

5.2. Fungsi Tunggal

fungsi tunggal merupakan suatu fungsi dimana proses pemanggilan dirinya sendiri tanpa melalui fungsi yang lain atau fungsi yang tidak terdapat dalam fungsi yang lain.

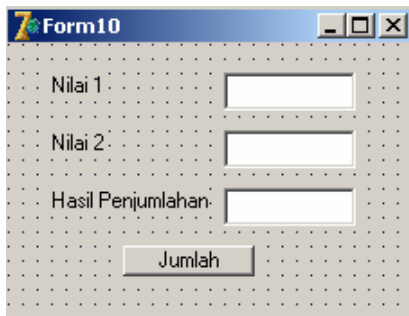
Contoh

a. Hasil yang diinginkan



The image shows a screenshot of a Windows application window titled "Form10". Inside the window, there are three text input fields arranged vertically. The first field is labeled "Nilai 1" and contains the number "5". The second field is labeled "Nilai 2" and also contains the number "5". The third field is labeled "Hasil Penjumlahan" and contains the number "10". Below these three fields, there is a button with the text "Jumlah" on it.

b. Desain Form

The image shows a screenshot of a Windows application window titled "Form10". The window has a standard Windows XP-style title bar with minimize, maximize, and close buttons. The main area of the form is a light gray grid. There are three text boxes arranged vertically. The first is labeled "Nilai 1", the second "Nilai 2", and the third "Hasil Penjumlahan". Below these text boxes is a button labeled "Jumlah".

c. Listing Program Dengan Fungsi

```
function hitung(var a,b : integer): integer;  
begin  
    hitung := a+b;  
end;
```

```
procedure TForm10.bjumlahClick(Sender: TObject);  
var  
    x,y : integer;  
    c : integer;  
begin  
    x := strtoint(enil1.Text);  
    y := strtoint(enil2.Text);  
    c:= hitung(x,y);  
    ehasil.Text := inttostr(c);  
end;
```

Didalam program utama tidak ditempatkan rumus untuk menghitung penjumlahan tetapi ketika dilakukan eksekusi terhadap program dan kita melakukan input nilai 1 dan 2 dan kita klik jumlah maka akan ditampilkan hasil yang diinginkan. Hasil itu diperoleh dari pengiriman nilai dari teks ke nilai pada variabel deprogram utama ($x := \text{strtoint}(\text{enil1.Text}); y := \text{strtoint}(\text{enil2.Text});$). Selanjutnya nilai pada variabel deprogram utama dikirim ke fungsi hitung untuk mendapat hasil penjumlahan ($c := \text{hitung}(x,y);$).

5.3. Fungsi Untuk Memanggil Fungsi yang Lain

Merupakan pembuatan program fungsi yang digunakan untuk memanggil program fungsi yang lain didalam satu listing program.

Contoh

a. Hasil Yang Diinginkan

The screenshot shows a Windows application window titled "Form15". The form is titled "Koperasi Agung Bakti". It contains several input fields and buttons. The "Rincian Anggota" section has a "Nama Peminjam" field with the value "Agung S". Below this, there are three input fields: "Jumlah Uang Pinjam" with the value "2000000", "Tingkat Bunga (dalam %)" with the value "2", and "Lama Pinjam" with the value "12". Further down, there are two more input fields: "Besarnya Uang yang Harus Dikembalikan" with the value "2.536.483,5891" and "Cicilan Uang Setiap Bulan" with the value "211.373,6324". At the bottom, there are three buttons: "Total", "Bersih", and "Close".

b. Desain Form

The screenshot shows the same Windows application window titled "Form15", but in design view. The form is titled "Koperasi Agung Bakti". The "Rincian Anggota" section has a "Nama Peminjam" field. A tooltip is visible over the "Nama Peminjam" field, displaying the text "Label2: TLabel" and "Origin: 16, 8; Size: 231 x 29". Below this, there are three input fields: "Jumlah Uang Pinjam", "Tingkat Bunga (dalam %)", and "Lama Pinjam". Further down, there are two more input fields: "Besarnya Uang yang Harus Dikembalikan" and "Cicilan Uang Setiap Bulan". At the bottom, there are three buttons: "Total", "Bersih", and "Close".

c. Desain Properties

Object	Caption/Text	Name
Label 1	Nama Peminjam	Label1
Label 2	Jumlah Uang Pinjam	Label2
Label 3	Tingkat Bunga (dalam %)	Label3
Label 4	Lama Pinjam	Label4
Label5	Besarnya Uang Yang Harus Dikembalikan	Label5
Label6	Cicilan Uang Setiap Bulan	Label6
GroupBox1	Rincian Nama Anggota	Groupbox 1
Edit1	-	Epeminjam
Edit2	-	Epinjam
Edit3	-	Ebunga
Edit4	-	Elama
Edit5	-	Ekembalian
Edit6	-	Ecicil
Button1	Total	Btotal
Button2	Bersih	Bbersih
Button3	Close	bclose

d. Listing Program

➤ Fungsi Perhitungan Bunga

```
function pangkat(x,y : real):real;  
begin  
    pangkat := exp(ln(x)*y);  
end;  
function bunga(e,f,g : real):real;  
begin  
    bunga := e*pangkat((1+f/100),g);  
end;
```

➤ Program Total

```
procedure TForm15.bttotalClick(Sender: TObject);  
var  
    a,b,c,d,e : real;  
begin  
    a := strtofloat(ejumlah.Text);  
    b := strtofloat(ebunga.Text);  
    c := strtofloat(elama.Text);
```

```

d := bunga(a,b,c);
ekembalian.Text := floattostr(d);
ekembalian.Text := formatfloat('#.###,0',d);
e := d/c;
ecicil.Text := floattostr(e);
ecicil.Text := formatfloat('#.###,0',e);
end;

```

➤ **Program Bersih**

```

procedure TForm15.bbbersihClick(Sender: TObject);
begin
ejumlah.Text := '';
elama.Text := '';
epeminjam.Text := '';
ebunga.Text := '';
ekembalian.Text := '';
ecicil.Text := '';
end;

```

➤ **Program Menutup Form**

```

procedure TForm15.bcloseClick(Sender: TObject);
begin
close;
end;

```

BAB VI ARRAY

6.1. Pengertian Array

Array (larik) merupakan tipe data terstruktur dimana didalamnya terdiri dari komponen – komponen yang mempunyai tipe data yang sama. Didalam suatu array jumlah komponen banyaknya adalah tetap. Didalam suatu larik atau array setiap kompoenen ditunjukkan oleh suatu index yang unik. Index dari setiap komponen array menunjukan urutan data atau identitas yang mewakili data yang ada didalamnya.

Logika sederhananya array itu bisa disamakan dengan dua orang dengan nama yang sama didalam suatu komunitas, untuk membedakan antara nama yang satu atau dengan nama yang lain maka diberikan initial tambahan untuk setiap nama.

6.2. Deklarasi Array

Didalam penulisan bahasa pemograman setiap penggunaan array harus dideklarsikan terlebih dahulu. Pendeklarasian array diawali dengan nama variabel array diikuti dengan indeks array yang dituliskan didalam tanda “[]” , diikuti dengan kata cadangan of dan tipe data yang dibutuhkan.

Bentuk Umum Penulisan

Tanda_pengenal : array [..tipe index ..] of tipe data;

Contoh :

Var

A : array[1..4] of integer;

B : array[1..5] of string;

C: array[1..10] of real;

Keterangan :

A,B,C merupakan tanda pengenal/ nama variabel dari array;

1..4 : merupakan tipe indek dari array, yang menunjukkan banyaknya data yang mampu disimpan.

Integer : menunjukan bahwa data yang diinput berupa bilangan bulat.

6.3. Alokasi Penggunaan Array

a. Array Static (*Static Array*)

array static adalah model pendeklarasian array dimana tipe data yang digunakan mempunyai nilai yang tetap. Nilai yang digunakan untuk menentukan jangkauan pada umumnya bernilai integer. Array Static juga bisa disebut Array dengan deklarasi tipe indeks subrange integer.

Bentuk Umum

array[indexType1, ..., indexTypen] of baseType

Keterangan = index type menunjukkan tipe data ordinal yang menunjukkan batasan atau elemen maksimal terhadap seberapa besar variabel tersebut menyimpan komponen.

Contoh

Var arrayku : array[1..5] of char

Atau juga

type

jangkauan = 1..5;

var

nilai : array[jangkauan] of integer;

b. Array Dinamis (*Dynamic arrays*)

Larik atau array dinamis merupakan array yang tidak mempunyai suatu jangkauan atau ukuran yang tetap. Tetapi ketika program dijalankan maka memori untuk suatu array dinamis direalokasikan ketika kita menugaskan suatu nilai kepada array. Dynamic-Array jenis ditandai oleh konstruksi (menyangkut) format

Bentuk Umum

array of baseType

Contoh

var nilai: array of Real;

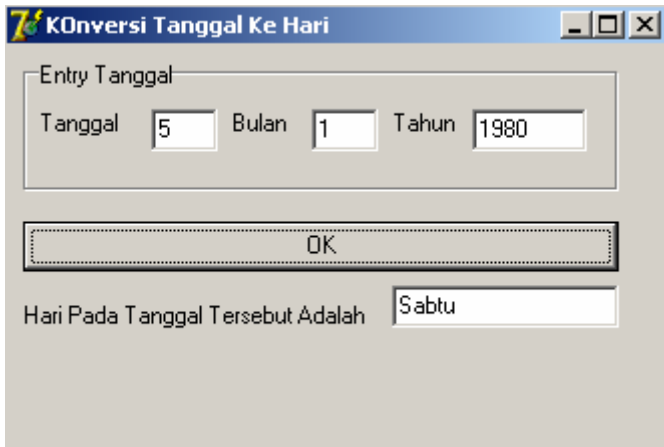
Dari deklarasi tersebut nilai yang merupakan deklarasi array belum memperoleh nilai yang tetap, tetapi hanya diberikan batasan sebagai tipe data real. Untuk mendeklarasikan array tersebut kita harus menempatkan array didalam suatu memori, caranya adalah dengan memanfaatkan fungsi dari perintah *sellength*.

Selllength(nilai,20)

Dari penggalan program tersebut nilai untuk array nilai tersebut mempunyai range sebanyak atau cakupan 20 untuk tipe data real, dengan indeex dimulai dari 0 sampai dengan 20.

6.4. Studi Kasus program dengan Array

a. Hasil ketika form Dijalankan



Gambar 5.1 Form Konversi Tanggal

b. Desain Properties

Object	Caption/Text	Name
Label 1	Tanggal	Label1
Label 2	Bulan	Label2
Label 3	Tahun	Label3
Label 4	Hari Pada Tanggal tersebut Adalah	Label4
GroupBox1	Entry Tanggal	Groupbox1
Edit1	-	Etgl
Edit2	-	Ebln
Edit3	-	Etahun
Edit4	-	Ehari
Command1	OK	bok

c. Listing Program

Program untuk Command OK

```
procedure TForm12.bhariClick(Sender: TObject);
type
  x = string[7];
const
  faktorbln : array[1..12] of byte = (0,3,3,6,1,4,6,2,5,0,3,5);
  hari : array[0..8] of
```



```

x=('Minggu','Senin','Selasa','','Rabu','Kamis','','Jum''at','Sabtu');
var
  hr : string;
  nama : string[255];
  j1,j2,j3,j4 : integer;
  tanggal,bulan,tahun : integer;
begin
  tanggal := strtoint(etgl.Text);
  bulan := strtoint(ebln.Text);
  tahun := strtoint(ethn.Text);
  if tahun > 1900 then tahun := tahun - 1900;
  j1 := trunc(tahun * 365.25);
  j2 := j1 + faktorbln[bulan];
  if (tahun/4 = int(tahun/4)) and (bulan < 3) then j2 := j2-1;
  j3 := j2 + tanggal;
  j4 := trunc(frac(j3/7) * 10);
  hr := Hari[j4];
  ehari.Text := hr;
end;

```

BAB VII

Operator String

7.1. Menggabungkan String

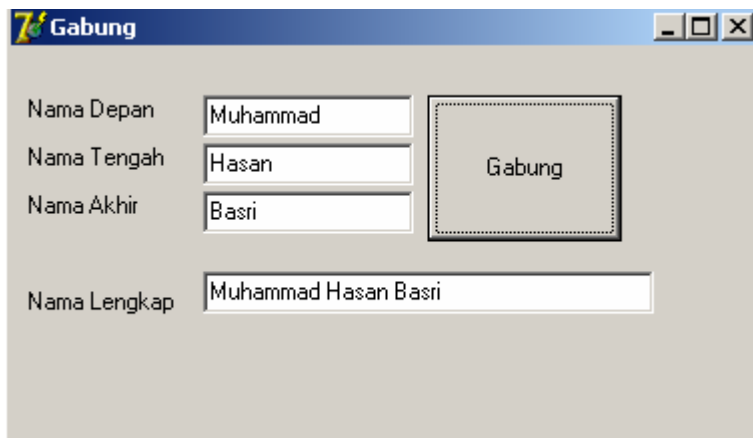
menggabungkan string adalah proses merangkai dua karakter atau lebih menjadi satu kalimat atau kata yang baru. Operator string yang dikenal dalam Bahasa Delphi adalah operator dengan symbol '+'.

a. Model Penulisan

```
var
    nama : string[20];
    saya : string[15];
    namasaya : string[50];
Begin
    Nama := 'Nama Saya adalah = ';
    Saya := 'Irnawan';
    Namasaya := 'nama+' '+saya;
    Ehasil.text := namasaya;
End.
```

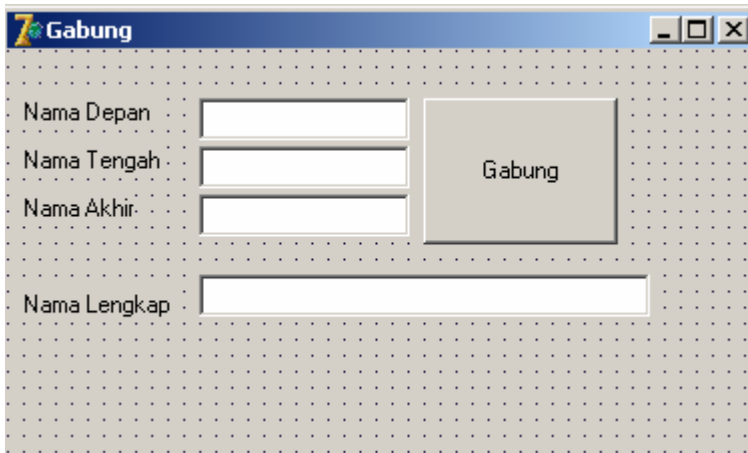
b. Contoh Program

Hasil yang Diperoleh



Gambar 7.1 Form Penggabungan String

Desain Form



Gambar 7.2 Desain Form Penggabungan String

Desain Properties

Object	Caption/Text	Name
Label 1	Nama Depan	Label1
Label 2	Nama Tengah	Label2
Label 3	Nama Akhir	Label3
Label 4	Nama Lengkap	Label4
Edit1	-	Edepan
Edit2	-	Etengah
Edit3	-	Eakhir
Edit4	-	Egabung
Button1	Gabung	bgabung

Listing Program

```
procedure TForm13.bgabungClick(Sender: TObject);
var
  depan : string[20];
  tengah : string[20];
  akhir : string[20];
  gabung : string[80];
begin
  depan := edepan.Text ;
  tengah := etengah.Text;
  akhir := eakhir.Text;
```

```
gabung := depan+' '+tengah+' '+akhir;  
elengkap.Text := gabung;  
end;
```

7.2. Menghapus Teks

Prosedur standart ini digunakan untuk menghapus atau mengurangi sebagian atau seluruh karakter terhadap string atau teks.

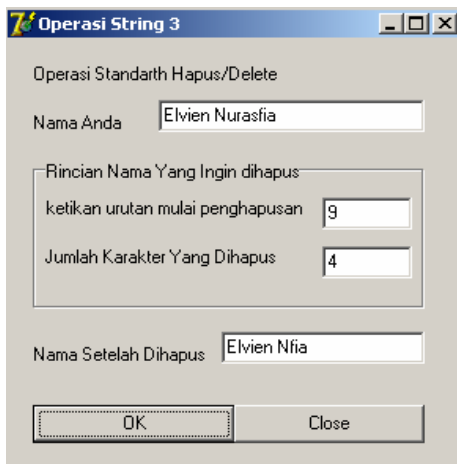
Bentuk umum penulisan

Delete(teks, index, jumlah)

Delete adalah prosedur standart yang digunakan untuk menghapus teks. *Teks* merupakan kalimat atau string yang akan dihapau, *Index* menunjukan posisi awal yang akan dihapus. *Jumlah* menunjukan jumlah karakter yang akan dihapus.

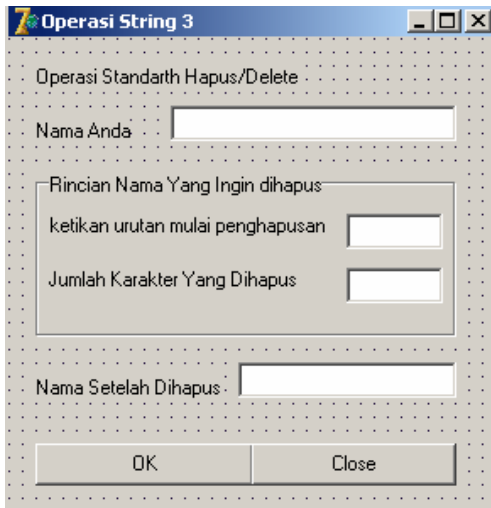
Contoh

a. Hasil Setelah Form Dijalankan



Gambar 7.3 Form Hapus teks

b. Desain Form



Gambar 7.4 Desain Form Hapus teks

c. Desain Properties

Object	Caption/Text	Name
Label 1	Nama Anda	Label1
Label 2	Ketikan urutan mulai penghapusan	Label2
Label 3	Jumlah Karakter yang dihapus	Label3
Label 4	Nama Sesudah dihapus	Label4
Edit1	-	Enama
Edit2	-	Eurut
Edit3	-	Ejumlah
Edit4	-	Ehasil
Button1	OK	bOK
Button2	Close	Bclose
Group Box 1	Rincian Nama Yang Ingin Dihapus	Group box 1

d. Listing Program

Program untuk eksekusi perintah penghapusan teks.

```
procedure TForm9.bokClick(Sender: TObject);  
var  
karakter:string;  
pos,jumlah:integer;
```

```

begin
karakter:= enama.Text;
pos := strtoint(eurut.Text);
jumlah := strtoint(ejumlah.Text);
Delete(karakter,pos,jumlah);
ehasil.Text := karakter;
end;

```

Program untuk keluar dari form

```

procedure TForm9.bcloseClick(Sender: TObject);
begin
close;
end;

```

7.3. Mencari Panjang teks (length)

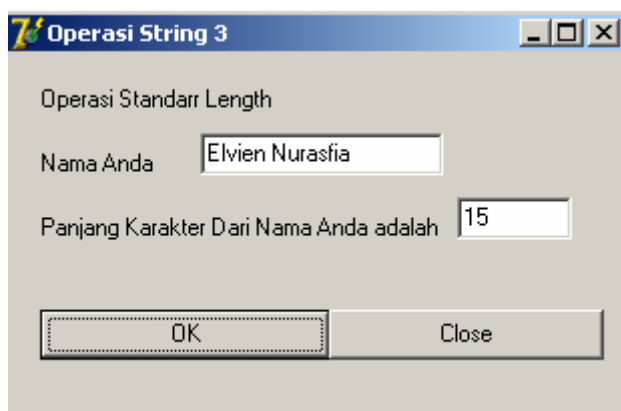
fungsi ini digunakan untuk menghitung panjang atau jumlah karakter dari suatu teks atau kalimat. Dalam menghitung jumlah karakter dalam suatu teks spasi akan dibaca sebagai satu karakter. Sebagai catatan hasil yang diperoleh dari perhitungan adalah bilangan bulat positif.

Bentuk Umum

Length(teks)

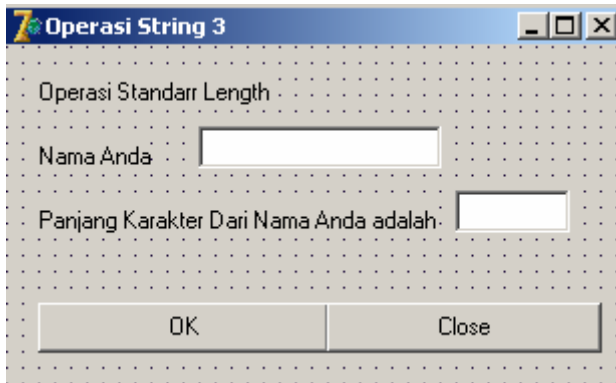
Contoh

a. Hasil setelah form dijalankan



Gambar 7.5 Hasil Form Panjang String

c. Desain Form



Gambar 7.6 Desain Form Panjang String

d. Desain properties

Object	Caption/Text	Name
Label 1	Operasi Standart Length	Label1
Label 2	Nama Anda	Label2
Label 3	Panjang Karakter dari Nama Anda	Label3
Edit1	-	Enama
Edit2	-	Ehasil
Button1	OK	bOK
Button2	Close	Bclose

e. Listing Program

Program untuk meneksekusi pencarian panjang karakter

```
procedure TForm8.bokClick(Sender: TObject);
var
  nama:string;
begin
  nama := enama.Text ;
  ehasil.Text := inttostr(length(nama));
end;
```

Program Untuk menutup Form

```
procedure TForm8.bcloseClick(Sender: TObject);
begin
```

```
close;  
end;
```

7.4. Mencari Karakter Pada Teks (POS)

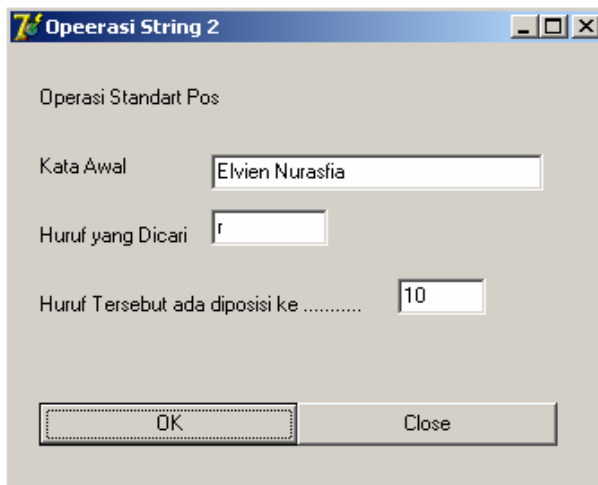
Fungsi ini digunakan untuk mencari letak atau posisi dari suatu karakter dari suatu nilai string. Nilai yang dihasilkan adalah berupa byte.

Bentuk umum

Pos(sustr, string);

Contoh

a. Hasil setelah dijalankan



The screenshot shows a window titled "Opeerasi String 2". Inside, the text "Operasi Standart Pos" is displayed. Below this, there are three input fields: "Kata Awal" containing "Elvien Nurasfia", "Huruf yang Dicari" containing "r", and "Huruf Tersebut ada diposisi ke" containing "10". At the bottom, there are two buttons: "OK" and "Close".

Gambar 7.7 Hasil Form Pencarian Huruf

b. Desain Form



The screenshot shows the same window titled "Opeerasi String 2", but it is in a design mode. The background is a grid. The layout of the form is visible, including the labels "Operasi Standart Pos", "Kata Awal", "Huruf yang Dicari", and "Huruf Tersebut ada diposisi ke", along with the "OK" and "Close" buttons at the bottom.

Gambar 7.8 Desain Form Pencarian Huruf

c. Desain Properties

Object	Caption/Text	Name
Label 1	Operasi Standart Pos	Label1
Label 2	Kata Awal	Label2
Label 3	Huruf yang Dicari	Label3
Label4	Huruf tersebut ada diposisi ke	
Edit1	-	eawal
Edit2	-	ecar
Edit3	-	ehasil
Button1	OK	bOK
Button2	Close	Bclose

d. Listing Program

Program untuk eksekusi perintah pencarian

```
procedure TForm7.bokClick(Sender: TObject);
var karakter : string;
kar : string;
hasil : integer;
begin
karakter:= eawal.Text;
kar:= ecari.Text;
hasil := pos(kar,karakter);
ehasil.Text := inttostr(hasil);
end;
```

Program Untuk menutup Form

```
procedure TForm7.bcloseClick(Sender: TObject);
begin
close;
end;
end.
```

7.5. Mengkopi String (Copy)

Fungsi ini digunakan untuk mencetak ulang string atau karakter sebanyak huruf yang dipilih dengan posisi karakter awal yang sudah ditentukan.

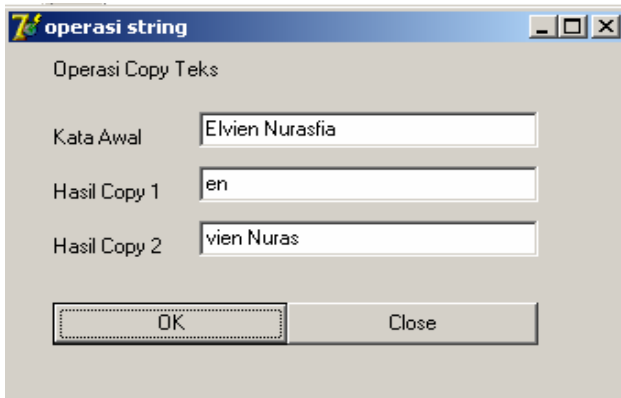
Bentuk Umum

Copy(teks, index, jumlah);

Penulisan awal diawali dengan perintah copy, dengan diikuti teks yang ditunjukkan teks. Awal pengcopian ditunjukkan dengan index, dan jumlah karakter yang akan di tulis ulang ditunjukkan deengan perintah jumlah.

Contoh

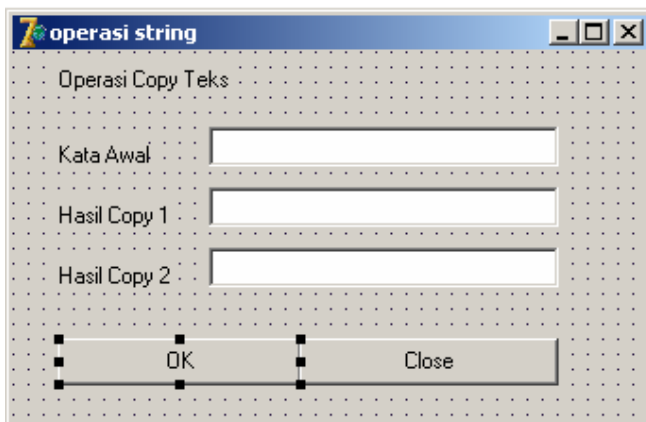
a. Hasil setelah form dijalankan



The screenshot shows a window titled "operasi string" with a standard Windows XP-style title bar. The window contains the text "Operasi Copy Teks". Below this, there are three text input fields. The first field, labeled "Kata Awal", contains the text "Elvien Nurasfia". The second field, labeled "Hasil Copy 1", contains the text "en". The third field, labeled "Hasil Copy 2", contains the text "vien Nuras". At the bottom of the window, there are two buttons: "OK" and "Close".

Gambar 7.9 Hasil Form Copy Teks

b. Desain Form



The screenshot shows the same window titled "operasi string", but it is in a design mode. The background of the window is a light gray grid. The text "Operasi Copy Teks" is at the top. Below it, there are three empty text input fields, each preceded by a label: "Kata Awal", "Hasil Copy 1", and "Hasil Copy 2". At the bottom, there are two buttons labeled "OK" and "Close". The window has a standard Windows XP-style title bar with minimize, maximize, and close buttons.

Gambar 7.10 Desain Form Copy Teks

c. Desain Properties

Object	Caption/Text	Name
Label 1	Operasi Copy Teks	Label1
Label 2	Kata Awal	Label2
Label 3	Hasil Copy 1	Label3
Label4	Hasil Copy 2	Label4
Edit1	-	eawal
Edit2	-	Ehasil1
Edit3	-	Ehasil2
Button1	OK	bOK
Button2	Close	Bclose

d. Listing Program

Program untuk mengeksekusi program

```
procedure TForm6.bokClick(Sender: TObject);
var
    karakter:string;
begin
    karakter := eawal.Text;
    ehasil1.Text := copy(karakter,5,3);
    ehasil2.Text := copy(karakter,3,10)
end;
```

Program untuk menutup Form

```
procedure TForm6.bcloseClick(Sender: TObject);
begin
    close;
end;
```

7.6. Konversi Karakter ke Ascii (Chr)

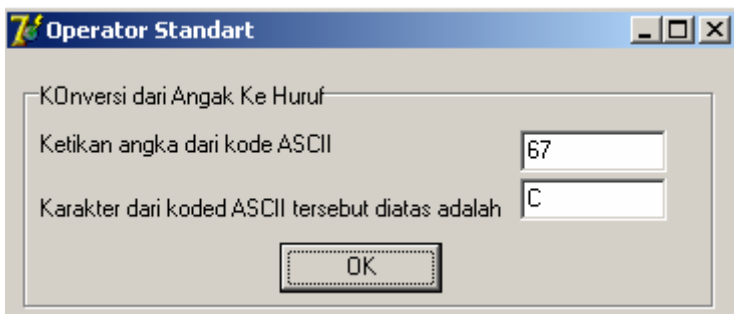
Fungsi ini digunakan untuk mengkonversi Kode Acsii menjadi nilai karakter atau huruf.

Bentuk Umum

```
Chr(karakter);
```

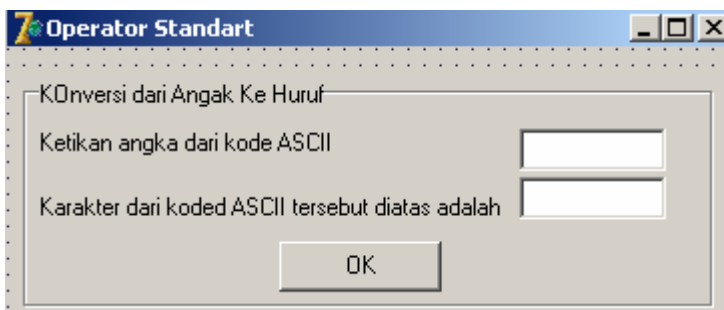
Contoh

a. Hasil setelah dijalankan



Gambar 7.11 Hasil Form Konvesi Ascii

b. Desain Form



Gambar 7.12 Desain Form Konversi ASCII

c. Desain properties

Object	Caption/Text	Name
Label 1	Ketikan angka dari kode ASCII	Label1
Label 2	Karakter dari kode Ascii tersebut adalah	Label2
Edit1	-	eanangka
Edit2	-	ehuruf
Button1	OK	bOK
Groupbox1	Konversi dari Angka ke huruf	Groupbox1

d. Listing Program

```
procedure TForm11.bokClick(Sender: TObject);  
var  
angka : integer;  
begin  
angka:= strtoint(eangka.Text);  
ehuruf.Text := chr(angka);  
end;
```

Bab VIII

Data Base Dekstop

8.1. Pengertian Data Base Dekstop

Data Base Dekstop adalah merupakan sebuah system aplikasi database yang sudah disertakan pada saat penginstalan Delphi. Komponen Data Base Dekstop yang merupakan bawaan Delphi meliputi = Paradox, dBse, MsSQL, Oracle, Maccess, Excel dan Lain – lain.

8.2. Langkah – Langkah pembuatan Data Base Dekstop

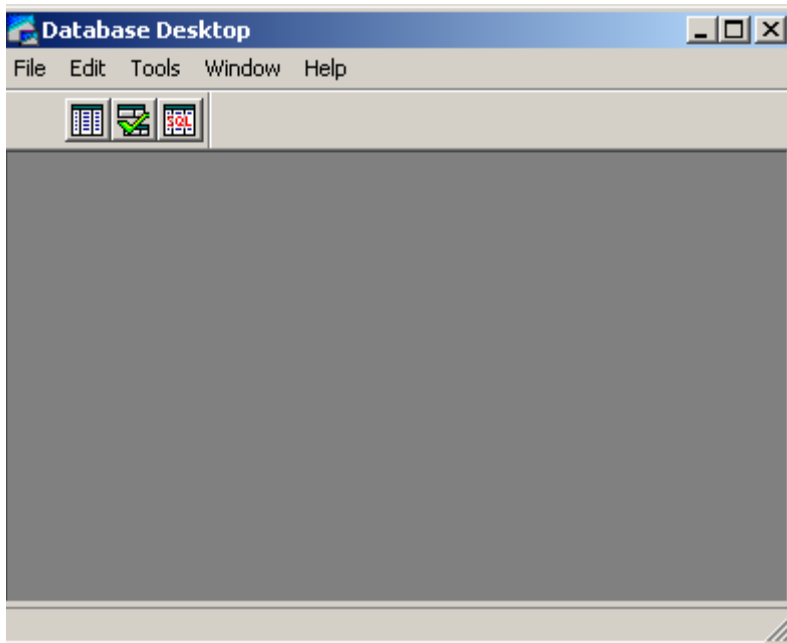
Ada dua cara untuk mengaktifkan database Dekstop

a. Melalui Icon Start

- Klik start | Pilih programs | Pilih Borland Delphi | Klik Data Base Dekstop

b. Melalui IDE Delphi

- Pilih dan klik Menu Tools | pilih dan klik Database Dekstop



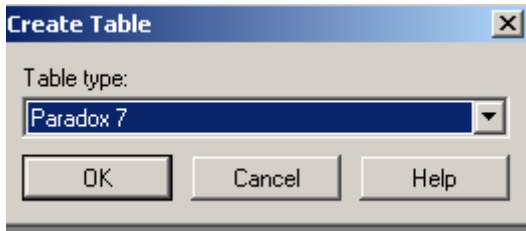
Gambar 8.1 Gambar Database Dekstop

8.3. Membuat Tabel

Setelah DBD diaktifkan, langkah berikut untuk membuat tabel =

1. Klik Menu File | New | table

2. Perhatikan tampilan jendela create table

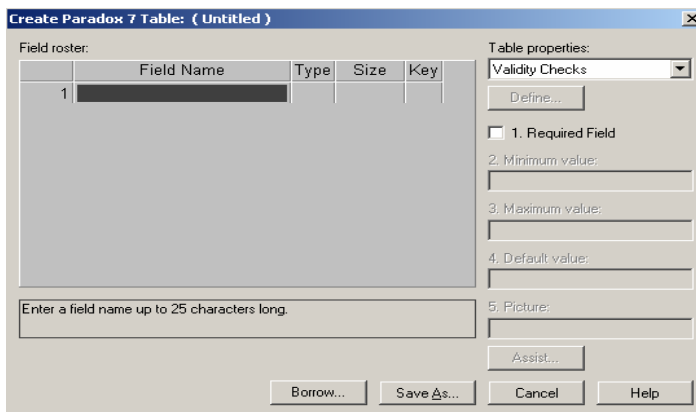


Gambar 8.2 Gambar Create tabel

3. Dari Jendela create table pilih dan klik pilihan table pada table type (mis = Paradox)

4. Klik Ok

5. Perhatikan tampilan jendela baru untuk struktur tabel pada paradox



Gambar 8.3 Gambar Desain Tabel

Penjelasan Untuk Struktur tabel

Field Name = digunakan untuk menuliskan nama field

Syarat Penulisan Nama Field

- Unik dan mudah diingat
- Diawali dengan Karakter
- Tidak diperbolehkan penulisan dengan tanda baca (! , . ? / + & % # @)
- Untuk field dengan dua kata atau lebih dihubungkan dengan Undecsare (_)

Type = Digunakan untuk menentukan tipe data pada Field

Type Data yang Dikenal Pada Paradox adalah

- Alpha = tipe data yang menampung semua karakter baik berupa huruf, angka maupun tanda baca.
- Number = tipe data yang menampung angka (numeric), jangkauannya mempunyai range untuk bilangan negative dan bilangan positif, (-10307 sampai dengan 10308). Dengan digit maksimal adalah 15 digit.

- Money = Tipe data yang hanya untuk angka. Sama seperti dengan Tipe Data *Number* hanya berbeda pada jangkauan. Pada tipe data number tidak mengenal dedesimal, tipe data money mengenal adanya decimal dan format mata uang.
- Short = tipe data angka yang hanya mempunyai jangkauan -32,767 to 32,767
- Long Integer = Sama seperti tipe data integer dengan jangkauan lebih luas (-2147483648 to 2147483647)
- Date = Tipe Data untuk tanggal (1 Januari 9999 BC to 31 Desember 9999 AD).
- Time = Tipe Data yang digunakan untuk setup waktu.
- Memo = Tipe Data untuk semua unit karakter dengan jangkauan 1 sampai dengan 240 karakter.
- Graphic = Tipe Data untuk gambar. (.BMP, .PCX, .TIF, .GIF, and .EPS file formats).
- Logical = Tipe data yang hanya mengenal benar atau salah (Yes or No).

Size = Digunakan untuk menentukan size terhadap suatu field

Catatan : Size yang kompatibel untuk diganti hanya tipe data Alpha dan memo.

Key = Digunakan untuk menentukan kunci utama (primary key)

Catatan = untuk memberikan primary key cukup ditekan *space bar* atau double klik pada field yang akan dibuat sebagai key (ditandai dengan *)

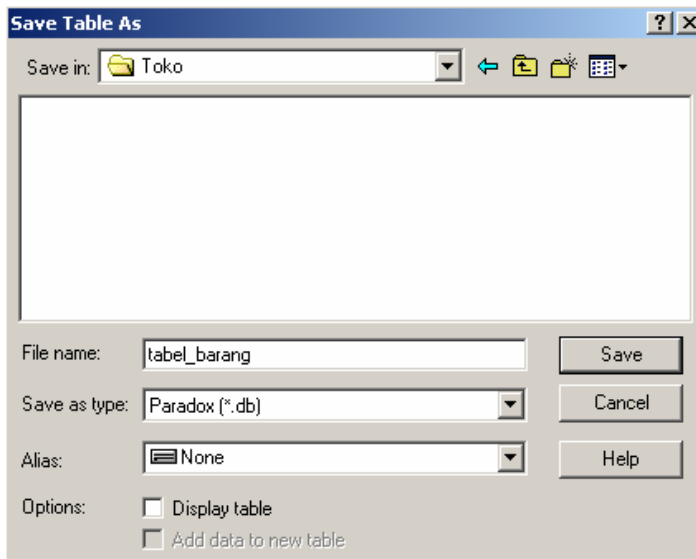
Misal = Buatlah Struktur tabel untuk tabel Mahasiswa seperti berikut ini

Field	Tipe Data	Size	Key
Kdbarang	Alpha	5	*
Nmbarang	Alpha	30	
Hrgbeli	Number		
Hrgjual	Number		
Stok	Number		

8.4. Menyimpan Tabel

langkah – langkah penyimpanan tabel (missal dari tersebut diatas akan kita simpan pada folder took dengan nama tabel Tabel_barang)

1. Pilih dan Klik **Save As**
2. Perhatikan tampilan jendeela baru untuk penyimpanan tabel



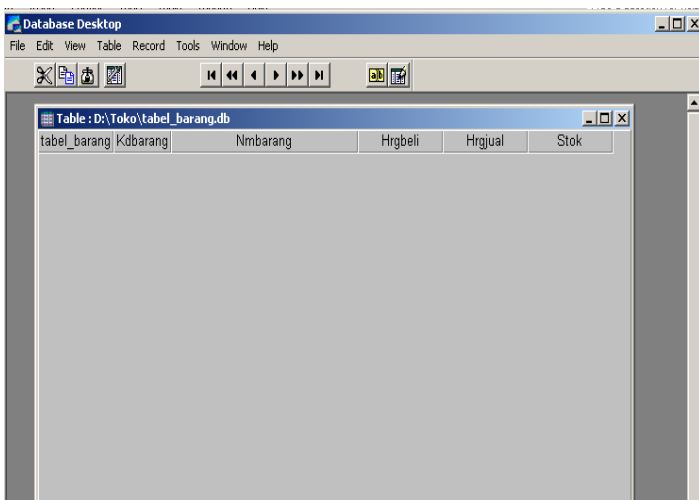
Gambar 8.4. Gambar Save table

3. Perhatikan cara penyimpanan seperti contoh diatas, nama folder harus sesuai dan nama field diketikan pada text box file name.
4. Setelah selesai klik Save.

8.5. Membuka tabel

Langkah – langkahnya adalah

1. Dari Data Base Dekstop, pilih dan klik File | Open
2. Cari tempat penyimpanan tabel dan setelah ditemukan nama tabel pilih dan klik Open
3. Perhatikan tampilan jendela baru untuk Data Base Dekstop




gambar 7.5 Open Table

Dari tampilan tersebut diatas ada beberapa hal yang bisa kita lakukan

a. Kembali ke Struktur Tabel

Langkahnya = Dari jendela tersebut klik icon *restructure* 

b. Mengisi Data

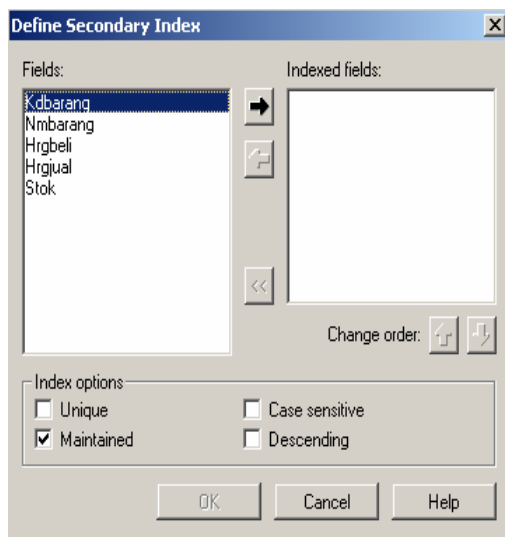
Langkahnya = Dari Jendela tersebut diatas klik icon *Edit Data* 

8.6. Membuat Index (secondary Index)

Index digunakan sebagai kunci untuk pencarian data ataupun digunakan untuk mengurutan data pada tabel. Didalam satu tabel diperkenankan menggunakan lebih dari satu index secondary.

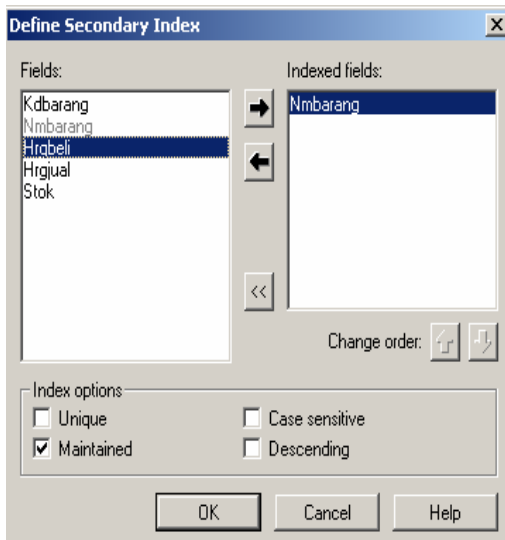
Langkahnya

- Aktifkn tabel barang
- Dari DBD pilih table | restructure atau icon Restructure
- Klik combo table properties dan pilih Secondary Index
- Klik tombol Define

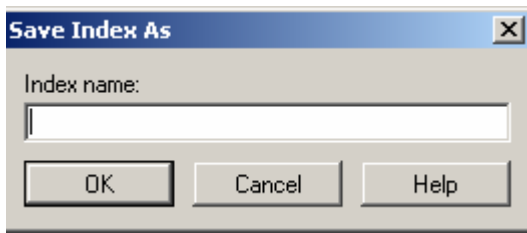


Gambar 8.5 Gambar Create Secondary Index

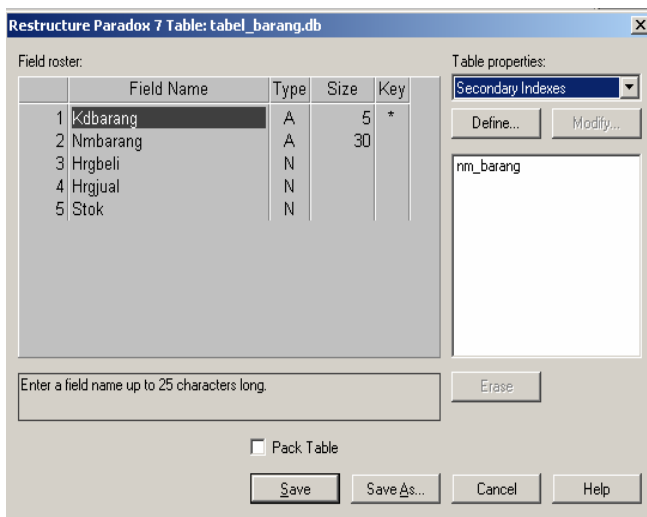
- Pilih dan klik field yang akan dijadikan secondary index (**Misal nama barang**)
- Klik tanda panah kekanan sehingga akan ditampilkan seperti gambar berikut



- Klik OK untuk mengakhiri pembuatan Secondary index
- Berikan nama index (**missal Nm_barang**) ketika ditampilkan jendela sebagai berikut



- Klik Ok untuk menutup form Save index dan kembali ke menu table barang, seperti berikut :



Catatan : Secondary Index dalam satu tabel boleh lebih dari satu dan yang perlu diingat dalam pemberian nama index tidak boleh sama dengan nama field.

8.7. Mengisi Data Pada Tabel

Langkah – langkah Pengisian data pada tabel

- Aktifkan database Dekstop | buka tabel barang
- Pada tampilan data pilih dan klik edit Data
- Isi Data Sebagai Berikut :

Kdbarang	Nama Barang	Harga Beli	Harga Jual	Kdpemasok	Stok
A0000	Mie Goreng Raya Sapi	2.000,00	2.200,00	01001	2
A0001	Mie Goreng Rasa Baso	1.500,00	1.600,00	01001	25
A0002	Indomie Rebus rasa Soto	2.100,00	2.300,00	03001	34
A0003	Mie Rasa Kari Ayam	1.600,00	1.700,00	02001	36
A0004	Biore	4.500,00	5.000,00	03001	56
A0006	Roti tawar	6.000,00	6.500,00	02002	56

8.8. Membuat Alias Manager

Fungsinya digunakan untuk menggantikan fungsi path ketika tabel yang dibuat akan dipanggil pada jendela program.

Contoh =

Tanpa Alias

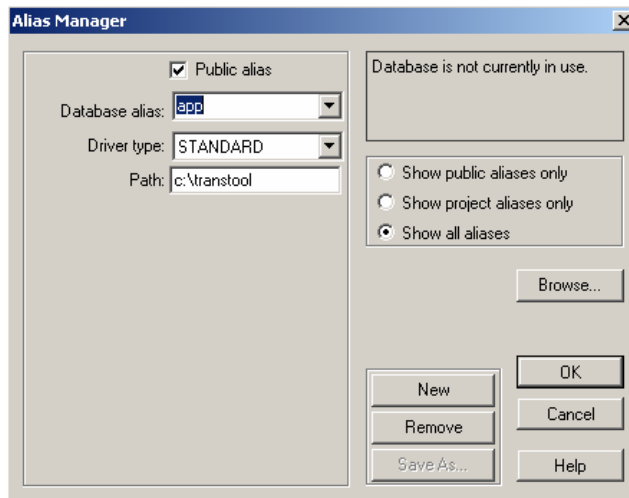
Ketika kita akan mengaktifkan tabel yang disimpan di C didalam Folder Toko dengan nama tabel Barang, maka kita harus menuliskan program sebagai berikut : **C:\Toko\barang.db**. (itu bisa semakin panjang jika folder penyimpanan data semakin bertingkat)

Dengan Alias

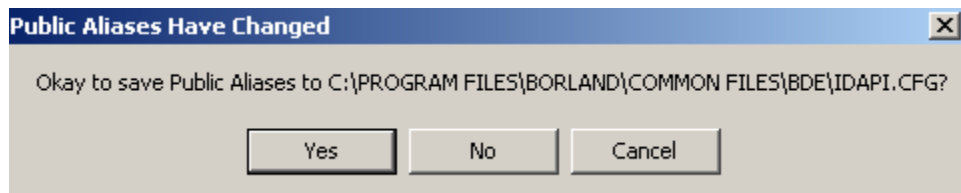
Pemanggilan program dengan **alias** kita tinggal panggil nama Alias baru kita tuliskan nama tabel. Dengan alias tidak mengenal penggunaan folder yang bertumpuk, karena **Alias** ditempatkan pada system **Sistem Operasi**

Langkah - langkahnya

1. Dari jendela Data base desktop pilih tools
2. pilih dan klik **alias manager**, perhatikan tampilan jendela Alias manager



3. Pada Jendela Alias Manger pilih dan klik New
4. Ketikan nama alias pada database alias, misal toko
5. Pada Database Type pilih Standard
6. Pada jendela path ketikan path tempat penyimpanan field tabel atau klik browse untuk pencarian secara otomatis.
7. Klik OK
8. Jika Yes jika muncul tampilan sebagai berikut



9. Jika tidak tampil Public Aliases Have Changed berarti pada kesalahan dalam menuliskan path pada jendela Alias Manager.

Catatan = dalam pembuatan nama alias cukup dibuat satu kali untuk satu project program, walaupun ada penambahan tabel setelah pembuatan Alias karena fungsi Alias sama seperti fungsi dari database.

8.9. Latihan

Tambahkan tabel baru dan simpan pada folder yang sama dengan nama = **Pemasok**

Struktur Tabel

Field	Time Data	Size	Key
Kdpemasok	Alpha	5	*
Nmpemasok	Alpha	30	
Almpemasok	Alpha	50	
Ktpemasok	Alpha	20	
Tlppemasok	Alpha	10	

Ketentuan Lain =

1. Buat Secondary Indeks untuk field nmpemasok dengan nama indek = nama dan ktpemasok dengan nama indeks = kota
2. Isi Data pemasok dengan data sebagai berikut

Kdpemasok	nmpemasok	Almpemasok	Ktpemasok	Tlppemasok
01001	PT Wahana Kencana	Jalan Bangka no 14	Jakarta	021-736666
01002	Pt Kencana Abadi	Jalan Bekasi Raya No 45	Bandung	021-450000
02001	PT Makmur Sejahtera	Jalan Raya Cibubur no 56	Bandung	021-888888
02002	PT Abadi Raya	Jalan Bandung No 35	Jakarta	021-488222
03001	PT Cinta Abadi	Jalan Lenteng Agung NO 62	Jakarta	021-888883
03002	PT Mutiara Makmur	Jalan Jaksa NO 53	Jakarta	021-000211

BAB IX

Komponen Query Atau SQL (Strutured Query Language)

9.1. Sekilas Mengenai SQL

SQL adalah bahasa standar untuk query yang difungsikan untuk memanipulasi suatu data pada Database. Hal itu meliputi DDL (*Data Definition Language*) meliputi pembuatan Data Base ataupun pembuatan tabel dan DML (*Data Manipulation Language*) meliputi perintah – perintah standar query.

9.2. Dasar – Dasar mengenai Metode SQL

9.2.1. DDL (*Data Defenition Language*)

a. Membuat Tabel

Pembuatan tabel dengan perintah SQL dapat dilakukan dengan perintah Create diikuti dengan nama tabel dan field yang dibutuhkan

Bentuk Umum

```
Create table nama_tabel (  
    Nm_filed1 tippedata1,  
    Nm_field2 tippedata2,  
    .....  
    .....  
    nm_fieldN tippedataN  
);
```

Contoh

```
Create table barang (  
    Kdbarang varchar(5) not null primary key,  
    Nmbarang varchar(15)  
);
```

9.2.2. DML (*Data Manipulation Language*)

a. Metode Select

Metode Select digunakan untuk menampilkan dan memilih suatu data dengan kondisi ataupun syarat yang sudah ditentukan dari satu atau beberapa tabel sekaligus dalam satu data base.

1. Select Tanpa Syarat

Merupakan perintah pencarian data tanpa diikuti perintah dengan kondisi ataupun persyaratan.

Bentuk Umum

Select *Daftar Field* **From** *Nama Tabel*

Contoh

Select * From barang

Perintah tersebut adalah digunakan untuk menampilkan semua data pada tabel barang. Untuk menampilkan data pada tabel dengan tidak menampilkan semua field pada tabel dengan menggunakan perintah **Select** diikuti dengan daftar field.

Contoh

Select kdbarang, nmbarang **From** barang

Perintah yang digunakan untuk menampilkan data Kode Barang dan Nama Barang pada tabel Barang.

2. Select Dengan Syarat

Adalah perintah menampilkan data yang diikuti dengan kondisi yang harus terpenuhi.

Bentuk Umum

Select *daftar_field* **From** *nama_tabel* **Where** *kondisi/persyaratan*

Contoh

Select * From barang **where** kdbarang = 'A001';

Adalah menampilkan semua data barang dengan kode barang = A001

Select * From barang **where** stok < 100;

Adalah menampilkan semua data barang dengan kondisi stok yang lebih kecil dari 100.

Select * From barang **where** hargabeli <= 2000 **and** Stok >= 5

Adalah menampilkan seluruh data pada tabel barang untuk harga beli dibawah atau sama dengan 2000 dan Stok diatas sama dengan 5.

Select * From barang where Harga_beli Between 5000 And 10000

Adalah semua data pada tabel barang dengan harga beli di antara 5000 dan 10000

Select * From barang where nmbarang like 'mei%'

Adalah menampilkan seluruh data dari tabel barang dengan nama barang yang berawalan Mei

Select * From barang where nmbarang Like '%mei'

Adalah menampilkan data dari tabel barang dengan nama barang yang berakhiran dengan kata mie.

b. Mengurutkan Data (Order By)

Fungsi ini digunakan untuk menurutkan data berdasarkan kondisi tertentu terhadap hasil Query.

Bentuk Umum

**Select daftar_field From nama_tabel order by nama_field
metode_pengurutan**

Contoh

Select * From barang order by nmbarang ASC

Menampilkan seluruh data barang diurutkan berdasarkan nama barang secara Ascending

c. Mengelompokkan Data (Group By)

Fungsi ini digunakan untuk mengelompokkan data berdasarkan field terpilih.

Bentuk Umum

Select * From barang Group By kdbarang

Contoh

Select * From barang group by kdbarang

Menampilkan seluruh data barang dengan dikelompokkan berdasarkan kode barang.

d. Fungsi Agregasi

Fungsi Agregasi adalah fungsi matematika yang digunakan bersamaan dengan perintah **Select**. Berbagai macam agregasi yang digunakan bersamaan dengan perintah **Select**

Fungsi	Kegunaan
Count	Untuk memperoleh jumlah record hasil Query
Sum	Untuk memperoleh total nilai dari suatu field
Avg	Untuk memperoleh nilai rata – rata
Max	Untuk memperoleh nilai terbesar
Min	Untuk memperoleh nilai terkecil

Bentuk Umum

Select *Fungsi_agregasi (nama_field)* **From** *nama_tabel*

Contoh

Select Sum(stok) **From** barang

e. Query Untuk banyak Tabel

Adalah fungsi Query yang digunakan untuk menampilkan lebih dari satu tabel

Bentuk Umum

Select *index1.daftar_field_tabel1, index2.daftar_field_tabel2* **From** *tabel1 index1, tabel2 index2* **Where** *index2.tabel1 = index1.tabel2*

Atau

Select Distinct *tabel1.daftar_field, tabel2.Daftar_field* **From** *tabel1, tabel2* **Where** *tabel1.nama_field = tabel2.nama_field*

Contoh

Select Distinct *barang.nmbarang, barang.harga_bel, pemasok.nmpemasok, pemasok.alamat* **From** *Pemasok, barang* **Where** *barang.kdpemasok = pemasok.kdpemasok.*

Perintah tersebut akan menampilkan nama barang dari tabel barang, harga beli dari tabel barang, nama pemasok dari tabel pemasok, dan alamat dari tabel pemasok dari tabel pemasok dan tabel barang dengan kondisi dimana kdpemasok di tabel barang sama dengan kode pemasok pada tabel pemasok.

f. Manipulasi Data

Perintah SQL yang digunakan untuk memanipulasi data pada sebuah tabel. Hal ini meliputi = menambah data, mengedit data ataupun menghapus data.

Ada tiga perintah yang sering digunakan untuk perintah SQL dalam hal memanipulasi Data

1. Insert

Perintah SQL yang digunakan untuk menambahkan data pada tabel. Bentuk umum penulisan perintah **Insert**.

Insert Into *nama_tabel (field1, field2, field3,.....fieldn)*
Values (*nilai1, nilai2, nilai3,.....nilain*)

Yang perlu diperhatikan adalah jumlah semua field dengan jumlah nilai yang akan dimasukan adalah sama. Untuk field dengan tipe Alpha (string) maka diantara nilai yang diinput diberikan dengan tanda kutif (' ').

Contoh

Insert Into *barang(kdbarang, nmbarang, hargabeli, hargajual, stok)*
Values ('A001', 'Indomei rasa Soto', 20000, 30000, 23)

2. Update

Perintah SQL yang digunakan untuk mengedit data yang sudah ada sebelumnya pada tabel.

Update *nama_tabel*
Set *field1 = nilai1, field2=nilai2, field3 = nilai3, fieldn = nilain)*
Where *syarat*

Contoh

Update barang
Set kdbarang = 'A001', nmbarang = 'Idomie rasa Ayam', hargabeli = 2000, hargajual = 3000, stok = 34 **where** kdbarang = 'A002'

Perintah SQL tersebut diatas adalah mengganti seluruh data yang ada ditabel barang untuk kode barang = A002 menjadi A001.

3. Delete

Perintah SQL digunakan untuk menghapus data yang ada di tabel. Bentuk umum penulisan perintah **Delete**.

Delete From *nama_tabel* **where** *syarat* (digunakan untuk menghapus record dengan kondisi yang diinginkan).

Atau

Delete From *nama_tabel* (digunakan untuk menghapus seluruh record pada tabel terpilih)

Contoh

Delete From barang **where** kdbarang = 'A001'

Merupakan kondisi dimana ada perintah untuk menghapus data barang untuk kode barang = A001

8.2.3. Penerapan Konsep SQL pada Aplikasi

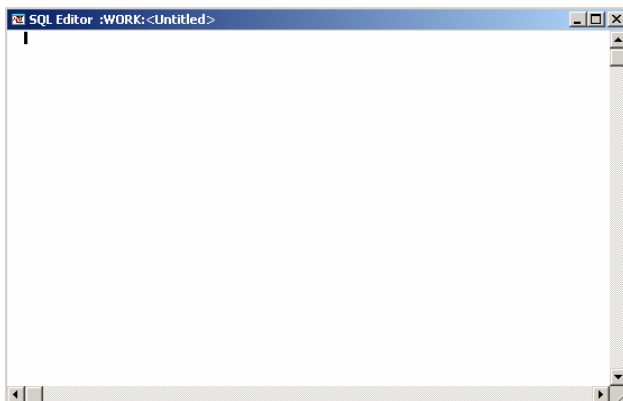
Penerapan konsep pemrograman SQL pada Delphi dapat kita implementasikan dengan dua konsep yaitu dengan konsep Data Base Dekstop dan Konsep pemrograman

9.2.3.1 Konsep Database Dekstop

Penerapan konsep ini dengan memanfaatkan jendela dari data base Dekstop dan SQL File.

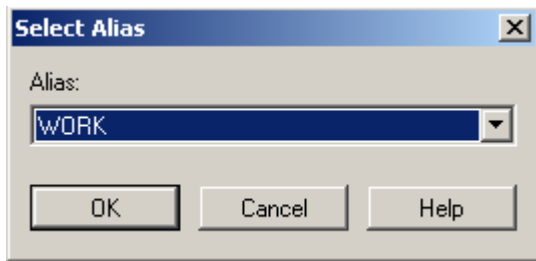
Langkah - langkahnya

- Aktifkan Data Base Dekstop
- Dari Menu **File | New | SQL File**. Jendela SQL Editor akan ditampilkan



Gambar 10.1 Jendela SQL Editor

- d. Dari Menu **SQL** pilih **Select Alias** (untuk tabel yang ada dalam folder penyimpanan), sehingga pada layer akan ditampilkan jendela **Select Alias**.



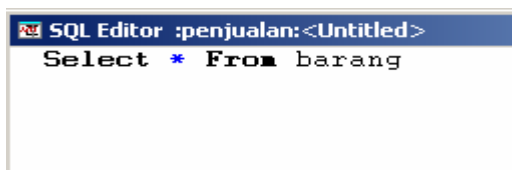
Gambar 9.2. Jendela Select Alias

- e. Dari jendela Select Alias pilih dan aktifkan nama alias yang sudah dibuat sebelumnya (**TOKO**) . Setelah selesai klik **OK**.
- f. Pada jendela editor tulisakn perintah SQL dan jalankan dengan memilih icon **Run SQL**.

Untuk mencoba kita gunakan beberapa kasus berikut ini

- a. Menampilkan semua data barang

Sintaks Penulisan



Hasil

Table : :PRIY:ANSWER.DB						
ANSWER	Kdbarang	Nmbarang	Hrgbeli	Hrgjual	Stok	Kdpemasok
1	A0000	Mie Goreng Raya Sapi	2.000,00	2.200,00	5,00	01001
2	A0001	Mie Goreng Rasa Baso	1.500,00	1.600,00	25,00	01001
3	A0002	Indomie Rebus rasa Soto	2.100,00	2.300,00	34,00	03001
4	A0003	Mie Rasa Kari Ayam	1.600,00	1.700,00	36,00	02001
5	A0004	Biore	4.500,00	5.000,00	56,00	03001
6	A0006	Roti tawar	6.000,00	6.500,00	56,00	02002

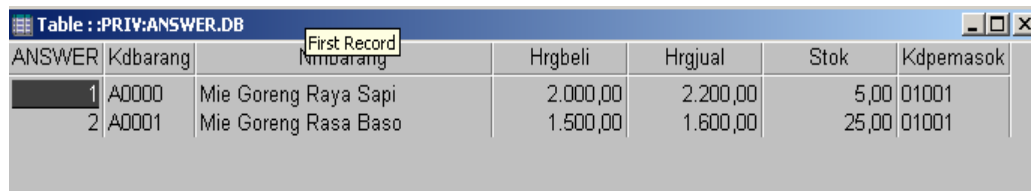
- b. Menampilkan Semua Data barang dengan kode pemasok = 01001

Sintaks Penulisan



```
SQL Editor :penjualan:<Untitled>
Select * From barang where kdpemasok = '01001'
```

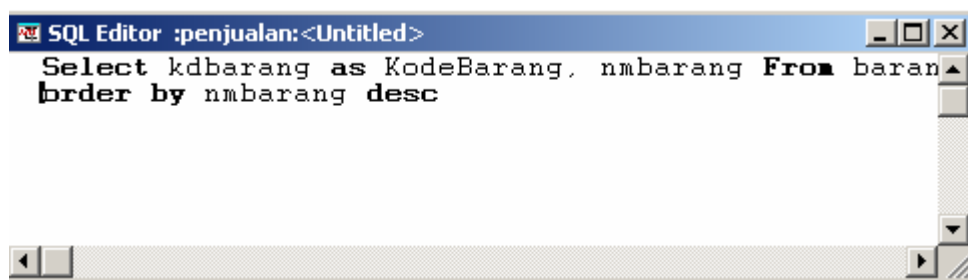
Hasil



ANSWER	Kdbarang	Nmbarang	Hrgbeli	Hrgjual	Stok	Kdpemasok
1	A0000	Mie Goreng Raya Sapi	2.000,00	2.200,00	5,00	01001
2	A0001	Mie Goreng Rasa Baso	1.500,00	1.600,00	25,00	01001

- c. Menampilkan kode barang dan nama barang dari tabel barang dengan pengurutan secara descending berdasar nama barang serta mengganti judul kdbarang menjadi KodeBarang.

Sintaks Penulisan



```
SQL Editor :penjualan:<Untitled>
Select kdbarang as KodeBarang, nmbarang From barang
order by nmbarang desc
```

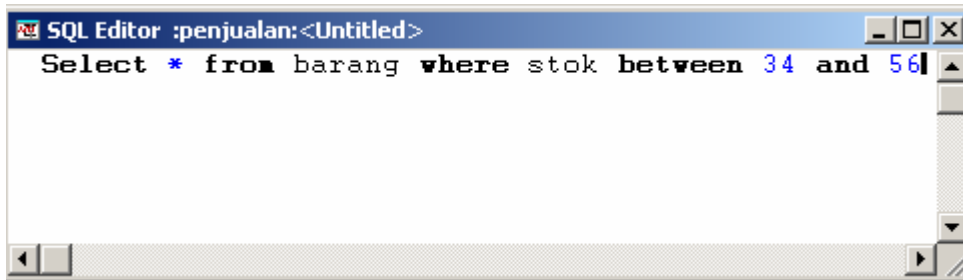
Hasil



ANSWER	KodeBarang	nmbarang
1	A0006	Roti tawar
2	A0003	Mie Rasa Kari Ayam
3	A0000	Mie Goreng Raya Sapi
4	A0001	Mie Goreng Rasa Baso
5	A0002	Indomie Rebus rasa Soto
6	A0004	Biore

- d. Menampilkan Semua Data barang untuk stok diantara 34 sampai dengan 56

Sintaks Penulisan



```
SQL Editor :penjualan:<Untitled>
Select * from barang where stok between 34 and 56
```

Hasil

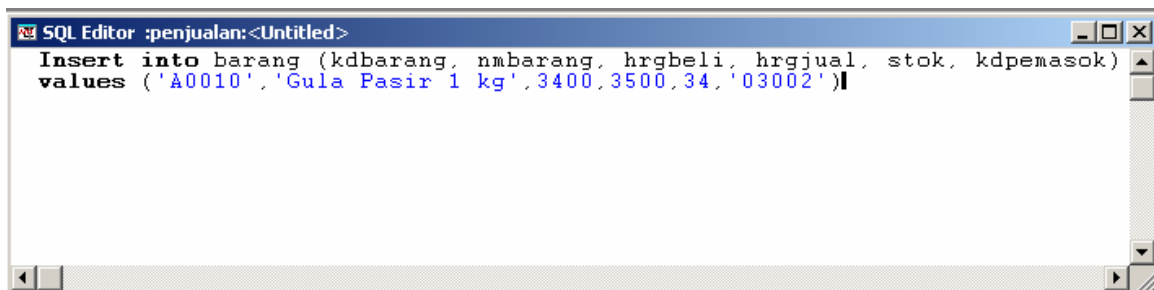


ANSWER	Kdbarang	Nmbarang	Hrgbeli	Hrgjual	Stok	Kdpemasok
1	A0002	Indomie Rebus rasa Soto	2.100,00	2.300,00	34,00	03001
2	A0003	Mie Rasa Kari Ayam	1.600,00	1.700,00	36,00	02001
3	A0004	Biore	4.500,00	5.000,00	56,00	03001
4	A0006	Roti tawar	6.000,00	6.500,00	56,00	02002

- e. Menambahkan data barang dengan data sebagai berikut =

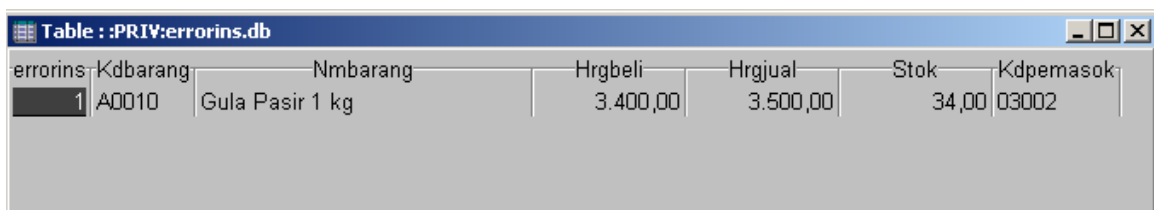
Kdbarang	Nmbarang	Harga Beli	Harga Jual	Stok	Kdpemasok
A0010	Gula Pasir 1 Kg	3400	3500	34	03002

Sintaks Penulisan



```
SQL Editor :penjualan:<Untitled>
Insert into barang (kdbarang, nmbarang, hrgbeli, hrgjual, stok, kdpemasok)
values ('A0010','Gula Pasir 1 kg',3400,3500,34,'03002')
```

Hasil



errorins	Kdbarang	Nmbarang	Hrgbeli	Hrgjual	Stok	Kdpemasok
1	A0010	Gula Pasir 1 kg	3.400,00	3.500,00	34,00	03002

Untuk melihat semua data gunakan perintah menampilkan data seluruhnya

- f. Mengganti Stok barang untuk kode A001 menjadi 15

Sintaks Penulisan

```
SQL Editor :penjualan:<Untitled>
Update barang
set stok = 15
where kdbarang = 'A0001'
```

Hasil

Table : :PRIV:updated.db					
updated-kdbarang	Nmbarang	Hrgbeli	Hrgjual	stok	Kdpemasok
1 A0001	Mie Goreng Rasa Baso	1.500,00	1.600,00	15,00	01001

G. Tugas

1. Hapus Data barang untuk field stok dibawah 20
2. Tampilkan Nama Pemasok, Kode Pemasok dari tabel pemasok dan nama barang, harga beli dari tabel barang. Sesuai dengan pemasok masing – masing seperti yang ada dalam tabel barang. **Dengan hasil seperti berikut ini**

Table : :PRIV:ANSWER.DB				
ANSWER	kdpemasok	nmpemasok	nmbarang	hrghbeli
1	01001	PT Wahana Kencana	Mie Goreng Rasa Baso	1.500,00
2	01001	PT Wahana Kencana	Mie Goreng Raya Sapi	2.000,00
3	02001	PT Makmur Sejahtera	Mie Rasa Kari Ayam	1.600,00
4	02002	PT Abadi Raya	Roti tawar	6.000,00
5	03001	PT Cinta Abadi	Biore	4.500,00
6	03001	PT Cinta Abadi	Indomie Rebus rasa Soto	2.100,00
7	03002	PT Mutiara Makmur	Gula Pasir 1 kg	3.400,00

3. Tampilkan Nama Pemasok, Kode Pemasok dari tabel pemasok dan nama barang, harga beli dari tabel barang sesuai dengan pemasok masing – masing seperti yang ada dalam tabel barang khusus nama barang yang berawalan dengan kata **Mie**. Dengan hasil seperti berikut ini

Table : :PRIV:ANSWER.DD				
ANSWER	kdpemasok	nmpemasok	nmbarang	hrgheli
1	01001	PT Wahana Kencana	Mie Goreng Rasa Baso	1.500,00
2	01001	PT Wahana Kencana	Mie Goreng Raya Sapi	2.000,00
3	02001	PT Makmur Sejahtera	Mie Rasa Kari Ayam	1.600,00

4. Cari rata – rata dari stok barang.

Bab X

Kontrol Data Set Dan Navigator



Kontrol Dataset merupakan control yang diberikan kepada tabel dalam suatu data base. Hal ini meliputi bagaimana data terhubung dengan tabel dari suatu data base. Setelah terhubung bagaimana proses mengerjakan record, entah itu ke awal, ke akhir, ke record sesudah maupun ke record sebelumnya.

10.1. Komponen Table

a. Borland Database Engine(BDE)

Komponen Borland Database Engine merupakan media penghubung antara database dengan aplikasi program. Pada awalnya BDE digunakan untuk paradox. Komponen BDE mendukung akses database yang merupakan bawaan dari Delphi walaupun tidak menutup kemungkinan bisa juga digunakan untuk pengaksesan database yang bersifat client/server. Salah satu kelebihan BDE adalah integrasi yang sudah sangat baik dengan Delphi. Sisi lain penggunaan BDE hanya maksimal untuk pemrograman yang bersifat standalone.


Bentuk Komponen BDE pada Delphi

Komponen Delphi	Keterangan
	Komponen Data Source = Komponen ini digunakan untuk mengakses Data base (terdapat dalam tab Data Access)
	Komponen table = Komponen ini digunakan untuk mengakses tabel yang terdapat dalam Database (terdapat dalam tab BDE)

b. ActiveX Data Object (ADO)

ADO merupakan salah satu teknologi Akses Data Base tingkat tinggi. Kelebihan antar muka ADO adalah merupakan teknologi Akses Independen terhadap setiap aplikasi program. ADO mendukung aplikasi yang bersifat local maupun yang berbasis Client Server. Komponen ADO terdapat dalam tab ADO pada Component Palette.


Bentuk Komponen ADO

Komponen Delphi	Keterangan
	Komponen ADO Connection = Komponen ini digunakan untuk mengakses Data base (terdapat dalam Tab ADO)

c. dbexpress


merupakan teknologi pengaksesan database dengan kemampuan yang bersifat terbatas pada teknologi akses untuk database yang bersifat client/Server. Kemampuan terbaik untuk dbexpress adalah didalam pembuatan laporan.

Bentuk Komponen DBExpress

Komponen Delphi	Keterangan
	Komponen Sql Connaction = Komponen ini digunakan untuk mengakses Data base (terdapat dalam tab Dbexpress)

d. Interbase Express (IBX)

Merupakan teknologi control databse yang bersifat open source. IBX merupakan teknologi akses database server yang bersifat khusus.

Komponen Delphi	Keterangan
	Komponen IBDatabase = Komponen ini digunakan untuk mengakses Data base (terdapat dalam tab Interbase)

10.2. Kontrol Tabel

Kontrol tabe adalah komponen yang menyediakan akses ke dalam sekumulan record terdapat didalam suatu tabel. Kontrol ini merupakan konsep bagaimana Akses ke dalam suatu tabel meenjadi lebih mudah. Salah satu Kontrol Tabel yang bisa digunakan adalah dengan memanfaatkan fasilitas yang ada di page BDE.

- a. Ttable = Digunakan untuk menghubungkan ke tabel dalam suatu Data.
- b. Tquery = Digunakan untuk menempatkan hasil dari sebuah Query.

Kontrol tabel memiliki dua properti

- Properti BOF (*Begin Of File*)
menunjukkan bahwa kontrol data di awal record.
- Properti EOF (*End of File*)
Menunjukkan bahwa kontrol data pada diakhir record.

10.3. Merancang Program Dengan Kontrol Tabel

a. Hasil Yang Diinginkan

The screenshot shows a Java Swing window titled "Form Penjualan Barang". Inside the window, there is a title bar and a main content area. The main content area has a title "Aplikasi Penjualan untuk maintance Daftar Barang" and a table with the following data:

Kdbarang	Nmbarang	Hrgbeli	Hrgjual	Stok	Kdpemas
A0000	Mie Goreng Raya Sapi	2000	2200	5	01001
A0001	Mie Goreng Rasa Baso	1500	1600	15	01001
A0002	Indomie Rebus rasa Soto	2100	2300	34	03001
A0003	Mie Rasa Kari Ayam	1600	1700	36	02001
A0004	Biore	4500	5000	56	03001
A0006	Roti tawar	6000	6500	56	02002
A0010	Gula Pasir 1 kg	3400	3500	34	03002

Below the table, there are several buttons: "Pertama", "Sebelum", "Sesudah", "Akhir", "Go TO", a text input field, and "Close".

b. Desain Form

This screenshot shows the same Java Swing window as the previous one, but with a design grid overlay. A tooltip is visible over the table, displaying the following information:

DBGrid1: TDBGrid
Origin: 8, 56; Size: 513 x 193
Tab Stop: True; Order: 0

c. Komponen Tambahan yang dibutuhkan

1. 6 button dan 1 edit text
2. 1 Dbgrid (dalam komponen **Data Control**)
Digunakan untuk menampilkan data tabel pada form.
3. 1 table (dalam **komponen BDE**)
Digunakan untuk menghubungkan antara tabel dengan database yang dibutuhkan
4. 1 datasource (dalam **Komponen Data Access**)
Digunakan untuk koneksi Database / Alias Data Base Didalam suatu folder.

d. Setting Properties

Object	Properties	
	Name	Caption/text
Label1	Label1	Aplikasi Penj
Button1	Bawal	Pertama
Button2	Bsebelum	Sebelum
Button3	BSesudah	Sesudah
Button4	Bakhir	Akhir
Button5	bLoncat	Go To
Button6	Bclose	Close
Edit1	Eloncat	-
Object	Data Base Name	Table Name
Table1	Toko	Barang.db
Object	Data Set	
Data Source1	Table1	
Object	Data Source	
Dbgrid1	Data Source1	
Object	Active	
Table1	True	

Ketikan program berikut ini

```
procedure TForm1.bpertamaClick(Sender: TObject);
begin
table1.First
end;
```

```
procedure TForm1.bsebelumClick(Sender: TObject);
begin
if table1.Bof then
    showmessage('Anda diawal record')
else
    TABLE1.Prior;
end;
```

```

procedure TForm1.bsesudahClick(Sender: TObject);
begin
if table1.Eof then
    showmessage('Anda berada pada record terakhir')
else
    table1.Last;
end;

procedure TForm1.bakhirClick(Sender: TObject);
begin
    table1.Next;
end;

procedure TForm1.bloncatClick(Sender: TObject);
begin
table1.MoveBy(strtoint(eloncat.Text));
end;

procedure TForm1.bcloseClick(Sender: TObject);
begin
if (application.MessageBox('Anda yakin form akan ditutup','Info',MB_YESNO)=
IDYES) then
    close;
end;

```

Bab XI

Pencarian Data

Salah satu konsep pemrograman berbasis data base proses pencarian data menjadi satu hal yang sangat penting didalam mendukung kesempurnaan hasil didalam sebuah aplikasi program. Fungsi secara umum adanya pencarian data adalah untuk mendapatkan secara cepat data yang diinginkan dari sebuah tabel didalam sebuah database.

11.1 Konsep Pemrograman Pencarian Data

b Konsep Pencarian dengan Properti

Konsep ini merupakan metode pencarian data dimana memanfaatkan kemampuan property untuk mendapatkan data yang diinginkan

Ada 5 konsep pencarian data dengan property

1. Locate

Locate adalah metode pencarian data untuk record yang sama dengan criteria yang sudah ada ataupun mendekati dengan kriteria yang ada. Pencarian locate dapat digunakan untuk tabel dengan index maupun yang tanpa index

Contoh

```
If not table1.locate ('nama', enama.text, []) then  
    Messagedlg('"' + enama.text + '" Tdak ditemukan', mterror, [mbok],0);
```

2. Findkey

Metode finkey digunakan mencari record yang sama . Metode ini dapat digunakan untuk tabel dengan index.

Contoh

```
Table1.indexname := ' ';  
If not table1.findkey ([00099]) then  
    Messagedlg('data Tdak ditemukan', mterror, [mbok],0);
```

Proses pencarian data diatas digunakan untuk table dengan index sebagai primary key. Sedangkan untuk pencarian dengan indes secondary dapat terlihat seperti contoh berikut ini.

```
Table1.indexname := 'nama';  
If not table1.findkey ([enama.text]) then  
    Messagedlg('"' + enama.text + '" Tdak ditemukan', mterror, [mbok],0);  
Else  
    Tampil;
```

3. Find Nearest

Metode ini digunakan untuk tabel dengan index selain itu juga dapat digunakan untuk pencarian record yang paling mendekati.

Contoh

```
Table1.indexname := 'nama';  
Table1.findnearest([enama.text])
```

4. Gotokey

Metode ini sama seperti dengan metode dengan findkey, tetapi dalam hal penulisan lebih rumit karena harus menjalankan terlebih dahulu event Setkey ataupun editkey. Konsep ini juga dapat digunakan untuk tabel dengan index maupun tidak.

Contoh

```
Table1.setkey;  
Table1.fieldbyname('nama').Asstring := enama.text;  
If not table1.gotokey then  
    Messagedlg('"' + enama.text + '" Tdak ditemukan', mterror, [mbok],0);  
Else  
    Tampil;
```

5. Gotonearest

Metode ini hampir sama dengan metoded findnearest. Perbedaan utamanya hanya dalam hal penulisan.

Contoh

```
Table1.setkey  
Table1.fieldbyname('nama').Asstring := enama.text;  
Table1.gotokey;
```

b Konsep pencarian data dengan perintah SQL

Metode ini merupakan metode pencarian data yang banyak digunakan oleh para pengembang aplikasi karena konsep ini dapat digunakan untuk semua bahasa pemrograman dengan berbagai macam database. Luwes merupakan salah satu keunggulan penggunaan konsep pencarian dengan perintah SQL. Selain itu konsep yang digunakan tidak perlu mengingat property yang digunakan.

Contoh

```
Var strsql : string;  
Begin  
Strsql := 'Select * from barang where nama like "' + enama.text + '%" ';  
Query1.sql.clear;  
Query1.sql.add(strsql);  
Query.close;  
Query.open;  
End;
```

11.2. Aplikasi Program Pencarian Data

a. Dengan Propeti

1. Hasil Form Setelah Dijalankan

The screenshot shows a Windows-style application window titled 'Form11'. It contains a table with the following data:

Kdbarang	Nmbarang	Hrgbeli	Hrgjual	Stok	Kdpemasok
A0000	Mie Goreng Raya Sapi	2000	2200	5	01001
A0001	Mie Goreng Rasa Baso	1500	1600	25	01001
A0002	Indomie Rebus rasa Soto	2100	2300	34	03001
A0003	Mie Rasa Kari Ayam	1600	1700	36	02001
A0004	Biore	4500	5000	56	03001
A0006	Roti tawar	6000	6500	56	02002

Below the table, there is a search section titled 'Cari Data Barang'. It includes a 'Kodee Barang' field with the value 'A0003' and a 'Cari' button. To the right, there are three more fields: 'Nama Barang' with the value 'Mie Rasa Kari Ayam', 'Harga Beli' with the value '1600', and 'Kode Pemasok' with the value '02001'. At the bottom left of the search section are 'Batal' and 'Close' buttons.

2. Desain Form

The screenshot shows the same 'Form11' application window, but in a design or layout mode. The table is no longer visible. Instead, there is a large empty rectangular area at the top. Below this area, the search controls are visible, including the 'Cari Data Barang' label, 'Kodee Barang' field, 'Cari' button, 'Nama Barang' field, 'Harga Beli' field, 'Kode Pemasok' field, and 'Batal' and 'Close' buttons. The background of the form has a dotted grid pattern.

3. Desain Properties

Catatan = Koneksi table dan data base akan dilakukan secara programming, jadi untuk setting properties tidak perlu disetup.

Object	Properties	
	Name	Caption/text
Group Box1	Group Box1	Cari Data Barang
Edit1	Ecari	-
Edit2	Enmbarang	-
Edit3	Ehrgheli	-
Edit4	Kdpemasok	-
Label1	Label1	Kode Barang
Label2	Label2	Nama Barang
Label3	Label3	Harga Beli
Label4	Label4	Kode Pemasok
Button1	Ecari	Cari
Button2	Ebatal	Batal
Button3	Eclose	Close
Object	Data Base Name	Table Name
Table1	-	-
Object	Data Set	
Data Source1	-	
Object	Data Source	
Dbgrid1	-	
Object	Active	
Table1	-	

4. Listing Program

Ketika pada saat program dijalankan maka proses koneksi program akan diaktifkan selama form tersebut diaktifkan.

```
procedure TForm11.FormCreate(Sender: TObject);
begin
table1.DatabaseName := 'penjualan' ;
table1.TableName := 'barang.db';
table1.Active:= true;
datasource1.DataSet:= table1;
dbgrid1.DataSource := datasource1;
end;
```

Program pencarian dimulai dengan input kode barang pada txtcari dan diakhiri dengan button cari. Jika data barang tidak ditemukan maka akan ditampilkan pesan, tetapi jika ditemukan maka data barang akan ditampilkan.

```
procedure TForm11.bcariClick(Sender: TObject);
begin
table1.IndexName := '';
if not table1.FindKey([ecari.Text]) then
begin
    showmessage('Data Belum ada');
    exit;
    form11.ActiveControl := ecari;
end
else
enmbarang.Text := table1['nmbarang'];
ehrgbeli.Text := table1['hrgbeli'];
ekdpemasok.Text := table1['kdpemasok'];
end;
```

Penjelasan

Table1.indexname := '' menunjukkan bahwa proses pencarian dengan memanfaatkan kunci utama (*primary key*), sehingga koneksi program dianjurkan dengan menggunakan metode *findkey*. *If Not table1.findkey([ecari.text])* menunjukkan bahwa jika data tidak ada, maka akan ditampilkan pesan bahwa tidak ditemukan dan kursor dikembalikan ke txtcari, tetapi jika ada maka data akan ditampilkan.

Program untuk membatalkan proses pencarian dengan mengaktifkan txtcari kembali.

```
procedure TForm11.bbatalClick(Sender: TObject);
begin
ecari.Text := '';
form11.ActiveControl := ecari;
end;
```

Program untuk menutup form

```
procedure TForm11.bbatalClick(Sender: TObject);
begin
ecari.Text := '';
form11.ActiveControl := ecari;
end;
```

b. Dengan Metode SQL

1. Hasil Setelah Form Dijalankan

The screenshot shows a Windows application window titled 'Form12'. It contains a search section at the top with the label 'Cari Data Barang', a text input field containing 'Mie', and a 'Cari' button. To the right are 'Batal' and 'Close' buttons. Below the search section is a table with the following data:

Kdbarang	Nmbarang	Hrgbeli	Hrgjual	Stok	Kdpemas
A0000	Mie Goreng Raya Sapi	2000	2200	5	01001
A0001	Mie Goreng Rasa Baso	1500	1600	25	01001
A0003	Mie Rasa Kari Ayam	1600	1700	36	02001

2. Desain Form

The screenshot shows the 'Form12' application in design mode. The search section at the top is visible, but the table area is empty. The background is a dotted grid. At the bottom right, there are icons for a table and a query, with the text 'SQL' below them.

3. Desain Properties

Catatan = untuk koneksi dengan perintah SQL maka icon table tidak dibutuhkan tetapi icon query yang dibutuhkan. Icon Query terdapat dalam tab BDE.

Object	Properties	
	Name	Caption/text
Group Box1	Group Box1	Cari Data Barang
Edit1	Ecari	-
Label1	Label1	Nama Barang
Button1	Ecari	Cari
Button2	Ebatal	Batal
Button3	Eclose	Close
Object	Data Base Name	Table Name
Query1	-	-
Object	Data Set	
Data Source1	-	
Object	Data Source	
Dbgrid1	-	
Object	Active	
Query	-	

4. Listing Program

Program koneksi dengan perintah SQL.

```

procedure TForm12.FormCreate(Sender: TObject);
//var strsql : string;
begin
query1.DatabaseName := 'penjualan';
query1.SQL.Add (' Select * from barang');
query1.Active := true;
datasource1.DataSet := query1;
dbgrid1.DataSource := datasource1;
end;

```

Penjelasan

Secara umum koneksi databse dan tabel hampir sama dengan koneksi dengan icon table, hanya terdapat perbedaan ketika harus mengkatifkan tabel. Dengan query harus dituliskan sintaks perintah SQL. Kondisi ini lebih luwes karena bisa menampilkan data lebih dari satu tabel.

Program pencarian nama

```
procedure TForm12.bcariClick(Sender: TObject);
var strsql : string;
begin
strsql := 'Select * from barang where nmbarang like "' + ecari.Text +
'%";
query1.SQL.Clear;
query1.SQL.Add(strsql);
query1.Close;
query1.Open;
end;
```

Program Untuk mengembalikan data ke semua record

```
procedure TForm12.bbatalClick(Sender: TObject);
var sql : string;
begin
sql := 'Select * from barang' ;
query1.SQL.Clear;
query1.SQL.Add(sql);
query1.Close;
query1.Open;
end;
```

Program menutup Form

```
procedure TForm12.bcloseClick(Sender: TObject);
begin
close;
end;
```

BAB XII

FILTER DAN RANGE DATA

Didalam Suatu konsep pemograman data base memberikan sebuah kemudahan didalam penanganan suatu tabel terhadap setiap user adalah sesuatu hal yang mutlak untuk diadakan. Salah satu fasilitas yang dapat memberikan kemudahan didalam pengelolaan data base adalah adanya fungsi range maupun filter. Fungsi ini hanya maksimal digunakan untuk field dengan type data numeric.

12.1. Filter

merupakan fungsi yang digunakan untuk membatasi tampilan data pada setiap tabel sesuai dengan data yang diinginkan. Method yang sering digunakan didalam penulisan fungsi filter adalah :

- a. *Tablefilterrecord* : fungsi ini digunakan untuk kontrol data terhadap field mana yang digunakan sebagai kunci.
- b. *Filtered* : fungsi ini digunakan untuk mengaktifkan fungsi dari event filter.
- c. *Fieldbyname* : berfungsi untuk menentukan field yang digunakan sebagai kunci pengurutan.
- d. *Indexname* : digunakan untuk mengaktifkan nama index dari suatu tabel.

9.2. Range

merupakan fungsi yang dapat digunakan untuk menampilkan data berdasarkan cakupan data atau kelompok data. Dalam hal ini fungsi range membutuhkan nilai yang dapat digunakan sebagai batasan awal kelompok maupun nilai untuk menentukan batasan akhir terhadap kelompok tersebut.

- a. *serangestart* : digunakan untuk menentukan nilai awal terhadap suatu range.
- b. *Setrangeend* : digunakan untuk menentukan nilai akhir terhadap suatu range.
- c. *Fieldbyname*: digunakan sebagai kunci field mana yang digunakan sebagai kunci.
- d. *Applyrange* : digunakan untuk mengeksekusi terhadap batasan range yang diinginkan.
- e. *Canceclrange* : digunakan untuk membatalkan perintah range yang sudah diberikan dan mengembalikan data ke fungsi normal.
- f. *Indexname* : digunakan untuk memanggil index primary key.

12.3. Merancang Aplikasi Program Dengan Fungsi Range dan Filter

a. Hasil setelah form dijalankan

The screenshot shows a window titled "Filter Data". It has two main sections for filtering data based on stock levels. The left section, "Menampilkan Data (berdasarkan Stok)", has input fields for "Batas Bawah" (5) and "Batas Akhir" (36), and a "Tampilkan" button. The right section, "Filter Data (berdasarkan Stok)", has an input field for "Stok Barang" and a "Tampilkan" button. A "Normal" button is also present. Below these sections is a table with the following data:

Kdbarang	Nmbarang	Hrgbeli	Hrgjual	Stok	Kdpemasok
A0000	Mie Goreng Raya Sapi	2000	2200	5	01001
A0001	Mie Goreng Rasa Baso	1500	1600	25	01001
A0002	Indomie Rebus rasa Soto	2100	2300	34	03001
A0003	Mie Rasa Kari Ayam	1600	1700	36	02001

b. Desain Form

The screenshot shows the same "Filter Data" window, but with a different set of data in the table. A tooltip is visible over the "Kdpemasok" column header, displaying the text: "bfilter: TButton", "Origin: 8, 48; Size: 153 x 25", and "Tab Stop: True; Order: 1". The table data is as follows:

Kdbarang	Nmbarang	Hrgbeli	Hrgjual	Stok	Kdpemasok
A0000	Mie Goreng Raya Sapi	2000			001
A0001	Mie Goreng Rasa Baso	1500	1600	25	01001
A0002	Indomie Rebus rasa Soto	2100	2300	34	03001
A0003	Mie Rasa Kari Ayam	1600	1700	36	02001
A0004	Biore	4500	5000	56	03001
A0006	Roti tawar	6000	6500	56	02002

c. Desain Properties

Object	Properties	
	Name	Caption/text
Group Box 1	Group Box 1	Menampilkan Data (berdasarkan kode)
Group Box 2	Group box2	Filter Data (berdasarkan kode)
Edit1	Eawal	-
Edit2	Eakhir	-
Edit3	Efilter	-
Label1	Label1	Batas Bawah
Label2	Label2	Batas Akhir
Label3	Label3	Stok Barang
Button1	Brange	Tampilkan
Button2	bfilter	Tampilkan
Button3	Bnormal	Normal
Object	Data Base Name	Table Name
Table1	Penjualan	Barang
Object	Data Set	
Data Source1	Table1	
Object	Data Source	
Dbgrid1	Data Source1	
Object	Active	
Query	True	

d. Listing Program

Untuk mengaktifkan fungsi filter record klik *table* dan pada **event** pilih dan doubleklik *onfilterrecord*. Ketikkan kode program berikut :

```

procedure TForm3.Table1FilterRecord(DataSet: TDataSet;
  var Accept: Boolean);
begin
accept := table1.FieldName('stok').AsFloat = strtofloat(efilter.Text);
end;

```

Program untuk mengeksekusi hasil filter record, ketikkan program pada button Filter.

```

procedure TForm3.bfilterClick(Sender: TObject);
begin
table1.IndexName := 'indstok';
table1.Filtered := true;

```

end;

Program mengeksekusi range tabel , ketikan program berikut pada button range

```
procedure TForm3.brangleClick(Sender: TObject);
begin
table1.IndexName := 'indstok';
table1.SetRange([strtofloat(eawal.Text)],[strtofloat(eakhir.Text)]);
table1.ApplyRange;
end;
```

Program untuk mengembalikan data ke posisi default

```
procedure TForm3.bnormalClick(Sender: TObject);
begin
table1.IndexName := '';
table1.CancelRange ;
table1.Filtered := false;
end;
```

BAB XIII

MANIPULASI DATA/TABEL MASTER

13.1. Program Tambah Data

Konsep program ini meliputi bagaimana proses pencarian data yang digunakan untuk memberikan batasan agar data yang disimpan bukan merupakan data yang sama, proses penyimpanan dan membatalkan penambahan data.

13.2. Event Yang Digunakan

- findkey* : digunakan untuk pencarian data, apakah data sudah ada atau belum.
- Append* : digunakan untuk membuka record kosong pada tabel
- Post* : event pada tabel ini digunakan untuk menyimpan data pada tabel.
- Databasename* : digunakan untuk memanggil nama data base.
- Table* : digunakan untuk mengaktifkan tabel.
- Dataset* : digunakan untuk menghubungkan antara datasource dengan tabel.
- Datasource* : digunakan untuk menghubungkan grid dengan tabel melalui datasource.

13.3. Aplikasi Tambah Data Barang

a. Hasil Form Setelah Dijalankan

The screenshot shows a window titled "Sistem Informasi Toko". On the left is a form with the following fields and values:

- kode barang: A0000
- nama Barang: Mie Goreng Raya Sapi
- Harga Beli: 2000
- harga Jual: 2200
- Stok: 5

Below the form are four buttons: "Tambah", "Simpan", "Batal", and "Exit". On the right is a table with the following data:

Kdbarang	Nmbarang	Hrgbeli	Hrgju
A0000	Mie Goreng Raya Sapi	2000	
A0001	Mie Goreng Rasa Baso	1500	
A0002	Indomie Rebus rasa Soto	2100	
A0003	Mie Rasa Kari Ayam	1600	
A0004	Biore	4500	
A0006	Roti tawar	6000	

b. Desain Form

c. Desain Properties

Object	Properties	
	Name	Caption/text
Edit1	Ekdbarang	-
Edit2	Enmbarang	-
Edit3	Ehrgheli	-
Edit4	Ehrgjual	-
Edit5	Estok	-
Label1	Label1	Kode Barang
Label2	Label2	Nama Barang
Label3	Label3	Harga Beli
Label4	Label4	Harga Jual
Label5	Label5	Stok
Button1	Btambah	&Tambah
Button2	Bsimpan	&Simpan
Button3	Bbatal	&Batal
Button4	Bexit	&Exit
Object	Data Base Name	Table Name
Table1	-	-
Object	Data Set	
Data Source1	-	
Object	Data Source	
Dbgrid1	-	
Object	Active	
Query	-	

d. Listing Program

➤ **Pembuatan Prosedur**

Ada 4 prosedur yang dibutuhkan : tampil, hidup, kosong, mati.

Langkah pembuatan

Double klik pada form, ketika berada pada jendela unit dari form tempatkan kursor berada declaration, khususnya pada bagian deklarasi public. Setelah itu ketikan deklarasi prosedur seperti berikut :

```
public
  procedure tampil;
  procedure kosong;
  procedure mati;
  procedure hidup;
  { Public declarations }
end;
```

Penulisan Program Prosedur

Penulisan program prosedur secara umum sama seperti penulisan program pada umumnya, hanya disini penulisan nama prosedurnya ditulis oleh programmer. Penulisananya terdapat pada bagian *implementation*.

Program untuk prosedur tampil

Digunakan untuk menampilkan data pada form

```
procedure tform6.tampil;
begin
  ekdbarang.Text := table1['kdbarang'];
  enmbarang.Text := table1['nmbarang'];
  ehrgbeli.Text := floattostr(table1['hrgbeli']);
  ehrgjual.Text := floattostr(table1['hrgjual']);
  estok.Text := floattostr(table1['stok']);
end;
```

Program Untuk Prosedur Mati

Digunakan untuk menonaktifkan semua text box yang ada pada form.

```
procedure tform6.mati;
begin
```

```
ekdbarang.Enabled := false;
enmbarang.Enabled := false;
ehrgbeli.Enabled := false;
ehrgjual.Enabled := false;
estok.Enabled := false;
end;
```

Program Untuk Prosedur Hidup

Fungsi ini digunakan untuk mengembalikan fungsi edit text, seperti fungsi normalnya.

```
procedure tform6.hidup;
begin
ekdbarang.Enabled := true;
enmbarang.Enabled := true;
ehrgbeli.Enabled := true;
ehrgjual.Enabled := true;
estok.Enabled := true;
end;
```

Program Prosedur Kosong

```
procedure tform6.kosong;
begin
ekdbarang.Text := '';
enmbarang.Text := '';
ehrgbeli.Text := '';
ehrgjual.Text := '';
estok.Text := '';
end;
```

- Langkah awal menghubungkan semua perangkat tabel untuk mengaktifkan tabel. Perintah ini terdapat pada *formcreate*.

```
procedure TForm6.FormCreate(Sender: TObject);
begin
table1.DatabaseName := 'penjualan' ;
table1.TableName := 'barang.db';
table1.Active:= true;
datasource1.DataSet:= table1;
dbgrid1.DataSource := datasource1;
```

end;

- Program tampil diaktifkan untuk menampilkan data pada edit text box, program digunakan sema form diaktifkan.

```
procedure TForm6.FormActivate(Sender: TObject);
begin
tampil;
mati;
bsimpan.Enabled := false;
end;
```

- Program tambah

```
procedure TForm6.btambahClick(Sender: TObject);
begin
mati;
ekdbarang.Enabled := true;
bsimpan.Enabled := true;
form6.ActiveControl := ekdbarang;
kosong;
end;
```

- Program Pencarian Data. Control data yang digunakan adalah enter ketika kode barang diinput pada edit text kodebarang.

```
procedure TForm6.ekdbarangKeyPress(Sender: TObject; var Key:
Char);
begin
if key = #13 then
begin
if table1.FindKey([ekdbarang.Text]) then
begin
showmessage('Data Sudah ada');
exit;
end
else
hidup;
ekdbarang.Enabled := false;
form6.ActiveControl := enmbarang;
end;
end;
```

- Program Penyimpanan Data ke Dalam tabel

```
procedure TForm6.bsimpanClick(Sender: TObject);
begin
table1.Append;
table1['kdbarang'] := ekdbarang.Text;
table1['nmbarang'] := enmbarang.Text ;
table1['hrghbeli'] := strtofloat(ehrghbeli.Text);
table1['hrghjual'] := strtofloat(ehrghjual.Text);
table1['stok']:= strtofloat(estok.Text);
table1.Post;
mati;
end;
```

- Program Membatalkan penyimpanan data

```
procedure TForm6.bbatalClick(Sender: TObject);
begin
kosong;
table1.First;
tampil;
mati;
end;
```

- Program Mengaktifkan data sesuai deengan pilihan pada setiap record pada grid.

```
procedure TForm6.DBGrid1CellClick(Column: TColumn);
begin
tampil;
end;
```

- Program untuk menutup form

```
procedure TForm6.bexitClick(Sender: TObject);
begin
table1.Close;
close;
end;
```


12.4. Aplikasi Program Edit Dan Hapus Data

a. Hasil Setelah Form Dijalankan

The screenshot shows a Windows-style application window titled "sISTEM INFORMASI TOKO". On the left is a form with the following fields and values:

- kode barang: A0000
- nama Barang: Mie Goreng Raya Sapi
- Harga Beli: 2000
- harga Jual: 2200
- Stok: 5

Below the form are five buttons: "eDIT", "uPDATE", "HAPUS", "Batal", and "Exit". On the right is a table with the following data:

Kdbarang	Nmbarang	Hrgbeli	Hrgju
A0000	Mie Goreng Raya Sapi	2000	
A0001	Mie Goreng Rasa Baso	1500	
A0002	Indomie Rebus rasa Soto	2100	
A0003	Mie Rasa Kari Ayam	1600	
A0004	Biore	4500	
A0006	Roti tawar	6000	

b Desain Form

This screenshot shows the same application window as in (a), but with a dotted grid background, indicating it is a design or design-time view. The form fields and buttons are positioned on the grid. The table on the right is also visible, showing the same data as in (a).

c. Desain properties

Object	Properties	
	Name	Caption/text
Edit1	Ekdbarang	-
Edit2	Enmbarang	-
Edit3	Ehrghbeli	-
Edit4	Ehrghjual	-
Edit5	Estok	-
Label1	Label1	Kode Barang
Label2	Label2	Nama Barang
Label3	Label3	Harga Beli

Label4	Label4	Harga Jual
Label5	Label5	Stok
Button1	Bedit	&Edit
Button2	Bupdate	&Update
Button3	Bhapus	&Hapus
Button4	Bbatal	&Batal
Button5	Bexit	&Exit
Object	Data Base Name	Table Name
Table1	-	-
Object	Data Set	
Data Source1	-	
Object	Data Source	
Dbgrid1	-	
Object	Active	
Query	-	

d. Listing program

➤ Program Untuk Prosedur

```

procedure tform7.tampil;
begin
ekdbarang.Text := table1['kdbarang'];
enmbarang.Text := table1['nmbarang'];
ehrgbeli.Text := floattostr(table1['hrghbeli']);
ehrgjual.Text := floattostr(table1['hrghjual']);
estok.Text := floattostr(table1['stok']);
end;

```

```

procedure tform7.kosong;
begin
ekdbarang.Text := '';
enmbarang.Text := '';
ehrgbeli.Text := '';
ehrgjual.Text := '';
estok.Text := '';
end;

```

```

procedure tform7.mati;
begin
ekdbarang.Enabled := false;

```

```

enmbarang.Enabled := false;
ehrgbeli.Enabled := false;
ehrgjual.Enabled := false;
estok.Enabled := false;
end;

```

```

procedure TForm7.hidup;
begin
ekdbarang.Enabled := true;
enmbarang.Enabled := true;
ehrgbeli.Enabled := true;
ehrgjual.Enabled := true;
estok.Enabled := true;
end;

```

- Program Untuk Mengaktifkan Database dan tabel pada form

```

procedure TForm7.FormCreate(Sender: TObject);
begin
table1.DatabaseName := 'penjualan' ;
table1.TableName := 'barang.db';
table1.Active:= true;
datasource1.DataSet:= table1;
dbgrid1.DataSource := datasource1;
end;

```

- Program Selama Form Aktif

```

procedure TForm7.FormActivate(Sender: TObject);
begin
tampil;
mati;
bupdate.Enabled := false;
bhapus.Enabled := false;
end;

```

- Program untuk button Edit

```

procedure TForm7.beditClick(Sender: TObject);
begin

```

```

mati;

ekdbarang.Enabled := true;
bupdate.Enabled := true;
bhapus.Enabled := true;
form7.ActiveControl := ekdbarang;
kosong;
end;

```

➤ Program Pencarian Data

```

procedure TForm7.ekdbarangKeyPress(Sender: TObject; var Key: Char);
begin
  if key = #13 then
  begin
    if not table1.FindKey([ekdbarang.Text]) then
    begin
      showmessage('Data Belum ada');
      exit;
    end
  else
    hidup;
    tampil;
    ekdbarang.Enabled := false;
    form7.ActiveControl := enmbarang;
  end;
end;

```

➤ Program Update

```

procedure TForm7.bupdateClick(Sender: TObject);
begin
  table1.Edit;
  table1['kdbarang'] := ekdbarang.Text;
  table1['nmbarang'] := enmbarang.Text;
  table1['hrgbeli'] := strtofloat(ehrgbeli.Text);
  table1['hrgjual'] := strtofloat(ehrgjual.Text);
  table1['stok'] := strtofloat(estok.Text);
  table1.Post;
  mati;

```

end;

➤ Program hapus Data

```
procedure TForm7.bhapusClick(Sender: TObject);  
begin  
  table1.Delete;  
  mati;  
  table1.First;  
end;
```

➤ Program Batal

```
procedure TForm6.bbatalClick(Sender: TObject);  
begin  
  kosong;  
  table1.First;  
  tampil;  
  mati;  
end;
```

➤ Program Untuk Dbgrid

```
procedure TForm6.DBGrid1CellClick(Column: TColumn);  
begin  
  tampil;  
end;
```

➤ Program Menutup Form

```
procedure TForm6.bexitClick(Sender: TObject);  
begin  
  table1.Close;  
  close;  
end;
```

BAB XIV

APLIKASI FORM FILE TRANSAKSI (PENJUALAN)

14.1. Menambah Tabel Transaksi

a. Struktur Tabel Yang Dibutuhkan

Field Name	Type	Size	Key	
Notrans	A	5	*	
Kdbarang	A	5		
Jmlbeli	N			
Subtotal	N			
Totbayar	N			
Cash	N			
Kembali	N			

b. Simpan tabel tersebut pada folder project dengan nama transaksi

c. Struktur Tabel Semtransaksi

Field Name	Type	Size	Key	
Kdbarang	A	5		
Jmlbeli	N			
Subtotal	N			
nmbarang	N			
hrngjual	N			

14.2. Aplikasi Program File Transaksi

file digunakan untuk mencatat semua transaksi penjualan yang terjadi.

a. Hasil Setelah Form Dijalankan

Sistem Informasi Penjualan Toko

No Transaksi:

Kode Barang: Nama Barang: Harga Jual Barang: Stok Barang: Jumlah Beli:

Kdbarang	Nmbarang	Hrgjual	Jmlbeli	Subtotal
A0001	Mie Goreng Rasa Baso	1600	1	1600
A0002	Indomie Rebus rasa Soto	2300	1	2300

New Simpan Close

Batal

Total Bayar:

Cash:

Kembali:

d. Desain Form

Sistem Informasi Penjualan Toko

No Transaksi:

Kode Barang: Nama Barang: Harga Jual Barang: Stok Barang: Jumlah Beli:

Kdbarang	Nmbarang	Hrgjual	Jmlbeli	Subtotal

New Simpan Close

Batal

Total Bayar:

Cash:

Kembali:

e. Desain properties

Object	Properties	
	Name	Caption/text
Edit1	Enotransaksi	-
Edit2	Ekdbarang	-
Edit3	Enmbarang	-
Edit4	Ehrgjual	-
Edit5	Estok	-

Edit6	Ebeli	-
Edit7	Etotbayar	-
Edit8	Ecash	-
Edit9	Ekembali	-
Label1	Label1	No Transaksi
Label2	Label2	Kode Barang
Label3	Label3	Nama Barang
Label4	Label4	Harga Jual
Label5	Label5	Stok
Label6	Label6	Jumlah beli
Label7	Label7	Total Bayar
Label8	Label8	Cash
Label9	Label9	Kembali
Button1	Bnew	&New
Button2	Bsimpan	&Simpan
Button3	Bclose	&Close
Button4	Bbatal	&Batal
Object	Data Base Name	Table Name
Table1	-	-
Table2	-	-
Table3	-	-
Object	Data Set	
Data Source1	-	
Object	Data Source	
Dbgrid1	-	
Object	Active	
Table1	-	
Table2	-	
Table3	-	

d. Listing Program

- Prosedur yang dibutuhkan

```

procedure tform13.kosong;
begin
ekdbarang.Text := '';
enmbarang.Text := '';
ehrgjual.Text := '';
estok.Text := '';
ebeli.Text := '';
end;

```

```

procedure tform13.mati;

```



```

begin
enotransaksi.Enabled := false;
ekdbarang.Enabled := false;
ebeli.Enabled := false;
ecash.Enabled := false;
end;

```

```

procedure TForm13.hidup;
begin
ekdbarang.Enabled := true;
ebeli.Enabled := true;
ecash.Enabled := true;
end;

```

- Program Menghubungkan database dan tabel

```

procedure TForm13.FormCreate(Sender: TObject);
begin
table1.DatabaseName := 'penjualan';
table2.DatabaseName:= 'penjualan';
table3.DatabaseName := 'penjualan';
table1.TableName:= 'transaksi';
table2.TableName := 'barang';
table3.TableName:= 'semtransaksi';
table1.Active := true;
table2.Active:=true;
table3.Active:=true;
datasource1.DataSet := table3;
dbgrid1.DataSource := datasource1;
mati;
bnew.Enabled := true;
bsimpan.Enabled := false;
bbatal.Enabled := false;
end;

```

- Program Penomoran untuk No Transaksi Secara otomatis

```

procedure TForm13.FormActivate(Sender: TObject);
var
c:string;

```

```

a:integer;
begin
if table1.RecordCount = 0 then
begin
  enotransaksi.Text := '00001';
  exit;
end
else
table1.Last;
c:=table1['notrans'];
a:= strtoint(c) +1;
if a < 10 then
begin
  enotransaksi.Text := '0000' + (inttostr(a));
end
else
if a < 100 then
begin
  enotransaksi.Text := '000' + (inttostr(a));
end
else
if a < 1000 then
begin
  enotransaksi.Text := '00' + (inttostr(a));
end
else
if a < 10000 then
begin
  enotransaksi.Text := '0' + (inttostr(a));
end
else
  enotransaksi.Text := inttostr(a);
end;

```

- Program untuk memulai pengisian data transaksi

```

procedure TForm13.bnewClick(Sender: TObject);
begin
while table3.RecordCount > 0 do
begin

```

```

table3.First;
table3.Delete;
table3.Next ;
end;
ekdbarang.Enabled := true;
form13.ActiveControl := ekdbarang;
ecash.Enabled := true;
bsimpan.Enabled := true;
etotbayar.Text := '0';
kosong;
end;

```

while table3.RecordCount > 0 merupakan perintah program yang digunakan untuk membaca tabel sementara apakah jumlah recordnya lebih dari satu atau kurang, jika kondisinya lebih dari satu maka dilakukan perintah penghapusan data pada tabel semtransaksi. Hal ini ditunjukkan dengan perintah *table3.Delete*; keadaan ini akan terus berulang selama kondisi jumlah record masih lebih dari satu.

➤ Program Pengisian data penjualan

```

procedure TForm13.ekdbarangKeyPress(Sender: TObject; var Key:
Char);
begin
if key = #13 then
begin
if table2.FindKey([ekdbarang.Text]) then
begin
enmbarang.Text := table2['nmbarang'];
estok.Text := table2['stok'];
ehrgjual.Text := table2['hrgjual'];
ebeli.Enabled := true;
form13.ActiveControl := ebeli;
end
else
showmessage('Data tidak ditemukan');
exit;
end;
end;

```

if table2.FindKey([ekdbarang.Text]) then kondisi ini digunakan untuk membaca kondisi apakah data barang yang ada dalam tabel barang ada atau tidak.

Sesuai dengan data yang diinput melalui edit text kode barang. Jika ada maka rincian data akan ditampilkan tetapi jika tidak maka akan ditampilkan pesan.

- Program Penyimpanan data transaksi ke tabel sementara.

```
procedure TForm13.ebeliKeyPress(Sender: TObject; var Key: Char);
var
  d:real;
begin
  if key = #13 then
  begin
    if (table2['stok'] < 0) or (table2['stok'] < (strtofloat(ebeli.Text))) then
    begin
      showmessage('Stok barang tidak mencukupi');
      exit;
    end
  else
    table2.Edit;
    table2['stok'] := table2['stok'] - strtofloat(ebeli.Text);
    table2.Post;
    table3.Append;
    table3['kdbarang'] := ekdbarang.Text ;
    table3['nmbarang']:= enmbarang.Text;
    table3['hr gjual']:= strtofloat(ehr gjual.Text);
    table3['jmlbeli']:= strtofloat(ebeli.Text);
    table3['subtotal']:= table3['hr gjual'] * table3['jmlbeli'];
    d := table3['subtotal'];
    table3.Post;
    etotbayar.Text := floattostr(d + strtofloat(etotbayar.Text));
    kosong;
    form13.ActiveControl := ekdbarang;
  end;
  bbatal.Enabled := true;
end;
```

Penjelasan

```
if (table2['stok'] < 0) or (table2['stok'] < (strtofloat(ebeli.Text))) then
begin
  showmessage('Stok barang tidak mencukupi');
```

```

    exit;
end

```

Penggalan program tersebut untuk memberikan batasan apakah stok pada tabel barang ada atau tidak atau memberikan batasan agar kondisi jumlah barang yang dibeli tidak melebihi jumlah stok barang. Jika kondisinya benar maka akan ditampilkan pesan bahwa stok barang sudah tidak mencukupi.

```

table2.Edit;
table2['stok'] := table2['stok'] - strtofloat(ebeli.Text);
table2.Post;

```

kondisi tersebut diatas dilakukan untuk tabel barang yang kondisinya stok lebih dari satu atau jumlah beli lebih kecil dari jumlah stok pada tabel barang. Perintah awal yang dilaksanakan adalah melakukan edit data pada tabel barang, khususnya jumlah stok yang berkurang sesuai dengan jumlah beli.

```

table3.Append;
table3['kdbarang'] := ekdbarang.Text ;
table3['nmbarang']:= enmbarang.Text;
table3['hrngjual']:= strtofloat(ehrngjual.Text);
.....
.....

```

Kondisi selanjutnya adalah menyimpan data penjualan kedalam tabel sementara penjualan.

- Program Pembatalan Transaksi sebelum penyimpanan

```

procedure TForm13.bbatalClick(Sender: TObject);
begin
    table3.First;
    while not table3.Eof do
    begin
        table2.Edit;
        table2['stok'] := table2['stok'] + table3['jmlbeli'];
        table2.Post;
        table3.Next;
    end;
end;

```

Penggalan program ini digunakan untuk membatalkan transaksi sebelum data disimpan didalam tabel penjualan dan mengembalikan stok barang ke kondisi semula.

➤ Program Simpan Data

```
procedure TForm13.bsimpanClick(Sender: TObject);
begin
bbatal.Enabled := false;
table3.First;
while not table3.Eof do
begin
table1.Append;
table1['notrans'] := enotransaksi.Text;
table1['kdbarang'] := table3['kdbarang'];
table1['jmlbeli'] := table3['jmlbeli'];
table1['subtotal'] := table3['subtotal'];
table1['total'] := strtofloat(etotbayar.Text);
table1['cash'] := strtofloat(ecash.Text);
table1['kembali']:= strtofloat(ekembali.Text);
table1.Post;
table3.Next;
end;
end;
```

➤ Program menghitung pembayaran

```
procedure TForm13.ecashKeyPress(Sender: TObject; var Key: Char);
var
a: real;
b: real;
c: real;
begin
if key = #13 then
begin
a := strtofloat(etotbayar.Text );
b := strtofloat(ecash.Text);
c:= b-a;
ekembali.Text := floatttostr(c);
end;
end;
```

- Program menutup form

```
procedure TForm13.bcloseClick(Sender: TObject);  
begin  
  table1.Close;  
  table2.Close;  
  table3.Close;  
  close;  
end;
```

BAB XV

QUERY DAN SQL LANJUTAN

Perintah Sql merupakan perintah dengan tingkat akurasi dan fleksibilitas yang tinggi, sehingga mampu diterapkan untuk semua aplikasi program baik itu berupa penggunaan perintah standart maupun perintah dengan tingkat kesulitan yang lebih tinggi. Kemudahan itu adalah salah satu kelebihan dengan menggunakan perintah SQL.

15.1. Aplikasi Program Hapus Dan Retur Stok Barang

a. Hasil Setelah Form Dijalankan

Sistem Informasi Penjualan Toko

No Transaksi: 00003
Kode Barang: A0002

Retur Data Close

Rincian Data Barang

Kode Barang	Nama Barang	Stok Barang	Harga Jual	Jumlah Beli
A0002	Indomie Rebus ras	30	2300	1

notrans	jmbeli	kdbarang	nmbarang	hrjual	subtotal
00003	1	A0002	Indomie Rebus rasa Soto	2300	2300

b. Desain Form

Sistem Informasi Penjualan Toko

No Transaksi:
Kode Barang: [Kode barang]

Retur Data Close

Rincian Data Barang

Kode Barang	Nama Barang	Stok Barang	Harga Jual	Jumlah Beli
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

SQL SQL SQL SQL

c. Desain Properties

Object	Properties	
	Name	Caption/text
Edit1	Enotransaksi	-
Edit2	Ekdbarang	-
Edit3	Enmbarang	-
Edit4	Estok	-
Edit5	Ehrgjual	-
Edit6	Ejmlbeli	-
Combo box1	ckdbarang	[Kode Barang]
Label1	Label1	No Transaksi
Label2	Label2	Kode Barang
Label3	Label3	Nama Barang
Label4	Label4	Harga Jual
Label5	Label5	Stok
Label6	Label6	Jumlah beli
Button1	Bretur	&Retur Penjualan
Button2	BClose	&Close
Object	Data Base Name	Table Name
Table1	-	-
Table2	-	-
Table3	-	-
Object	Data Set	
Data Source1	-	
Object	Data Source	
Dbgrid1	-	
Object	Active	
Table1	-	
Table2	-	
Table3	-	

d. Listing Program

➤ Program Menghubungkan Form dengan Database

```

procedure TForm14.FormCreate(Sender: TObject);
begin
  query1.DatabaseName := 'penjualan';
  query1.SQL.Add ('Select p.notrans, p.jmlbeli, p.kdbarang, b.nmbarang,
  b.hrgjual, p.subtotal from transaksi p, barang b where b.kdbarang =
  p.kdbarang');
  query2.DatabaseName := 'penjualan';

```

```

query2.SQL.Add('Select * from barang');
query3.DatabaseName := 'penjualan';
query1.Active := true;
datasource1.DataSet := query1;
dbgrid1.DataSource := datasource1;
end;

```

Penjelasan

Query.databasename : 'penjualan' . penggalan program untuk menghubungkan database dengan form. Pada grid kalau diperhatikan ada field – field dari dua tabel yaitu tabel transaksi dengan tabel barang dengan hanya data barang yang ada ditabel transaksi yang ditampilkan. Hal itu bisa dilihat dari perintah *'Select p.notrans, p.jmlbeli, p.kdbarang, b.nmbarang, b.hrgjual, p.subtotal from transaksi p, barang b where b.kdbarang = p.kdbarang'*. **Select p.notrans, p.jmlbeli, p.kdbarang, b.nmbarang, b.hrgjual, p.subtotal from transaksi p, barang b** adalah menampilkan data dari tabel barang dan transaksi. **where b.kdbarang = p.kdbarang** adalah kondisi dimana hanya data barang yang ada ditabel transaksi yang ditampilkan, tanpa diberikan kondisi seperti ini maka semua data barang akan ditampilkan walaupun ditabel transaksi tidak ada.

- Program Untuk menampilkan data transaksi dan data barang sesuai dengan pilihan nomor transaksi yang ada di text nomor transaksi. Dan menampilkan kode barang pada combo box sesuai dengan jumlah barang yang dibeli

```

procedure TForm14.enotransaksiKeyPress(Sender: TObject; var Key: Char);
var strsql : string;
begin
if key = #13 then
begin
strsql := 'Select p.notrans, p.jmlbeli, p.kdbarang, b.nmbarang, b.hrgjual,
p.subtotal from transaksi p, barang b where notrans like "' +
enotransaksi.Text + '%" and p.kdbarang = b.kdbarang';
query1.SQL.Clear;
query1.SQL.Add(strsql);
query1.Close;
query1.Open;
while not query1.Eof do
begin
ckdbarang.Items.Add (query1['kdbarang']);
query1.Next;
end;

```

```
end;  
end;
```

Penjelasan

Secara umum perintah ini adalah sama seperti pada perintah untuk menampilkan data barang sesuai dengan data barang yang ada ditabel transaksi. Perbedaan disini adalah kondisi dimana data yang ditampilkan tidak semuanya tetapi hanya disesuaikan dengan pilihan kondisi sesuai dengan nomor transaksi yang sudah diinput (*where notrans like '' + enotransaksi.Text + '%' and p.kdbarang = b.kdbarang'*). Kondisi yang selanjutnya adalah menampilkan data barang sesuai dengan data barang yang ada ditabel transaksi. Hal itu bisa dilihat dari penggalan program berikut :

```
while not query1.Eof do  
begin  
ckdbarang.Items.Add (query1['kdbarang']);  
query1.Next;  
end
```

Program untuk menampilkan data pada grid diperoleh dari penggalan program berikut :

```
query1.Close;  
query1.Open;
```

- Program menampilkan rincian data barang ketikan kode barang terpilih

```
procedure TForm14.ckdbarangClick(Sender: TObject);  
var sql : string;  
begin  
sql := 'Select p.jmlbeli, b.kdbarang, b.nmbarang, b.hrgjual, b.stok from  
transaksi p, barang b where kdbarang like '' + ckdbarang.Text + '%' and p.kdbarang = b.kdbarang';  
query2.SQL.Clear;  
query2.SQL.Add(sql);  
query2.Close;  
query2.Open;  
ekdbarang.Text := query2['kdbarang'];  
enmbarang.Text := query2['nmbarang'];  
estok.Text := query2['stok'];  
ehrgjual.Text := query2['hrjual'];  
ejmlbeli.Text := query2['jmlbeli'];
```

end;

- Program untuk menghapus data transaksi dan mengupdate data stok yang ada ditabel barang.

```
procedure TForm14.hapusitemClick(Sender: TObject);
var sql : string;
strsql : string;
strsql1 : string;
a : integer;
b : integer;
c : integer ;
begin
a := strtoint(ejmlbeli.Text);
b := strtoint(estok.Text);
c := a + b;
sql := 'update barang set stok = "%d" where kdbarang = ' + ckdbarang.Text + ''';
query2.SQL.Clear;
query2.SQL.Add(format(sql,[c]));
query2.ExecSQL;
strsql := 'Delete from transaksi where kdbarang = ' + ckdbarang.Text + '' and notrans = ' + enotransaksi.Text + ''';
query3.SQL.Clear;
query3.SQL.Add(strsql);
query3.ExecSQL;
strsql1 := 'Select p.notrans, p.jmlbeli, p.kdbarang, b.nmbarang, b.hrgjual, p.subtotal from transaksi p, barang b where p.kdbarang = b.kdbarang';
query1.SQL.Clear;
query1.SQL.Add(strsql1);
query1.Close;
query1.Open;
form14.ActiveControl := enotransaksi;
end;
```

Penjelasan

```
sql := 'update barang set stok = "%d" where kdbarang = ' + ckdbarang.Text + ''';
query2.SQL.Clear;
```

```
query2.SQL.Add(format(sql,[c]));  
query2.ExecSQL;
```

Penggalan program tersebut untuk mengupdate data barang sesuai dengan kondisi kode barang sama dengan kode barang yang dipilih. *%d* adalah format masukan yang diijinkan didalam penggunaan perintah SQL. (*%s* untuk data dengan tipe string, *%d* untuk field dengan tipe data integer, *%n* untuk field data dengan tipe numeric). *query2.SQL.Add(format(sql,[c]));* adalah penggalan program untuk menambahkan program kedalam Query. Eksekusi program dilaksanakan dengan perintah *query2.ExecSQL*:

```
strsql := 'Delete from transaksi where kdbarang = "' + ckdbarang.Text  
+ '" and notrans = "' + enotransaksi.Text + '"';  
query3.SQL.Clear;  
query3.SQL.Add(strsql);  
query3.ExecSQL;
```

Adalah penggalan program untuk menghapus data barang yang ada ditabel transaksi. Hal itu di dikarenakan pada query diberikan perintah *'Delete from transaksi where kdbarang = "' + ckdbarang.Text + '" and notrans = "' + enotransaksi.Text + '"*:

```
strsql1 := 'Select p.notrans, p.jmlbeli, p.kdbarang, b.nmbarang, b.hrgjual,  
p.subtotal from transaksi p, barang b where p.kdbarang = b.kdbarang';  
query1.SQL.Clear;  
query1.SQL.Add(strsql1);  
query1.Close;  
query1.Open;
```

Adalah penggalan program untuk menampilkan data transaksi setelah adanya pengurangan data pengembalian (retur).

BAB XVI

PEMBUATAN LAPORAN

16.1 Membuat laporan Data Barang

a. Laporan Yang Dihasilkan

LAPORAN DATA BARANG toko Surya kencana				
Kode Barang	Nama Barang	Harga Beli	Harga Jual	Stok
A0000	Mie Goreng Raya Sapi	2000	2200	5
A0001	Mie Goreng Rasa Baso	1500	1600	25
A0002	Indomie Rebus rasa Soto	2100	2300	34
A0003	Mie Rasa Kari Ayam	1600	1700	36
A0004	Biore	4500	5000	56
A0006	Roti tawar	6000	6500	56
Total Stok				212

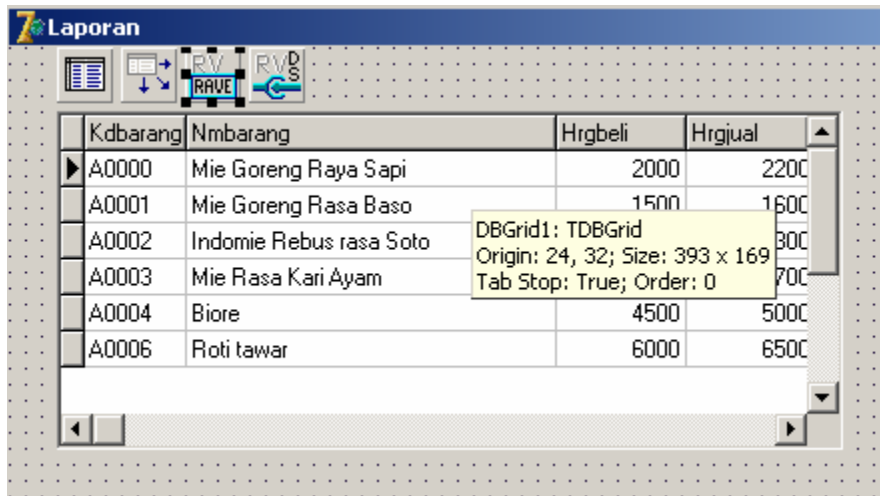
b. Desain laporan

LAPORAN DATA BARANG toko Surya kencana				
Kode Barang	Nama Barang	Harga Beli	Harga Jual	Stok
[Kdbarang]	[Nmbarang]	[Hrgbeli]	[Hrgjual]	[Stok]
Total Stok				[Sum(DataVi

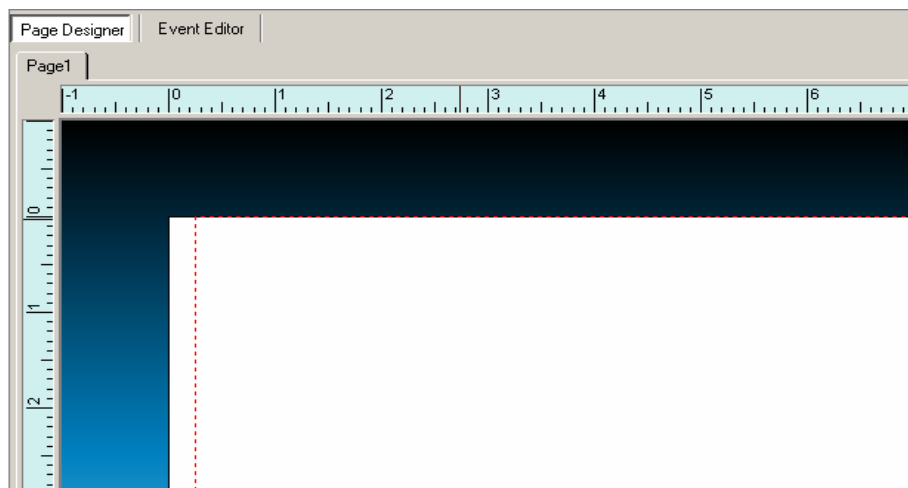
c. Langkah - langkahnya

Mendesain laporan

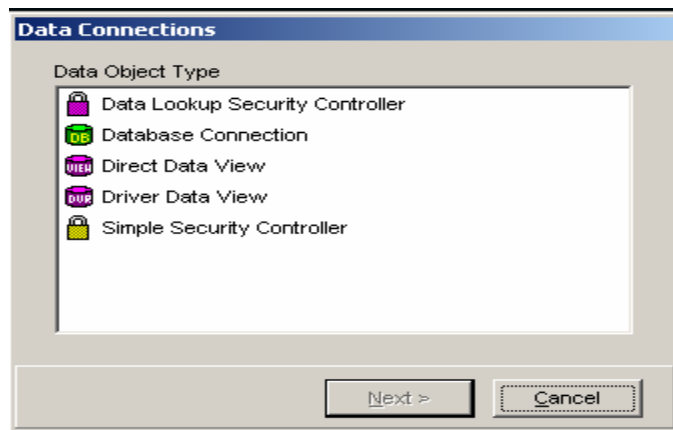
1. Buat dan tambahkan form baru pada jendela project
2. Desain Form, sehingga membentuk tampilan sebagai berikut



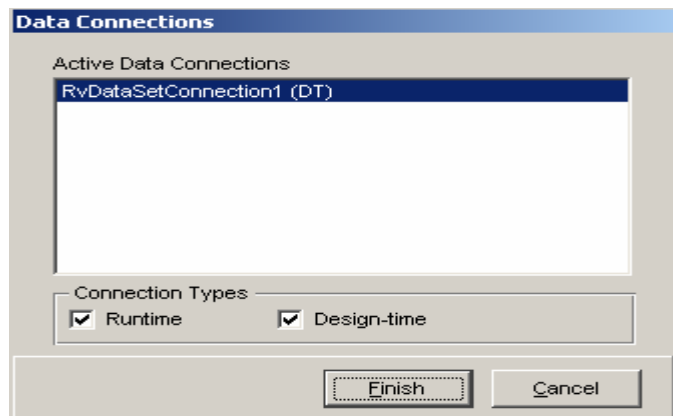
3. Komponen yang dibutuhkan, Table (Komponen BDE), Data Source (Data Access), DBgrid1 (Data Control), Rvproject1 dan RvDatasetconnection (Komponen Rave).
4. Koneksi Table (Data Base name, Table name, Active), Data source (Data Set) dan Dbgrid1 (Data Source) sesuai dengan komponen Masing – masing.
5. Klik RvDataConnection, pada properties pilih dataset dan koneksi ke Table1
6. Doubleklik pada Rvproject, sehingga pada form akan ditampilkan layer sebagai berikut.



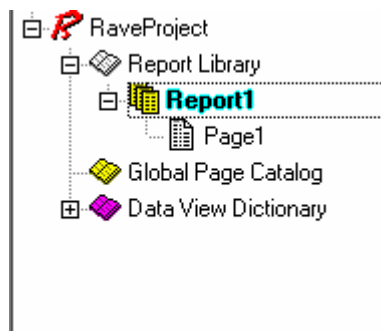
7. Dari Jendela Report Rave , Klik File – New Data Object, perhatikan tampilan jendela baru pada jendela report



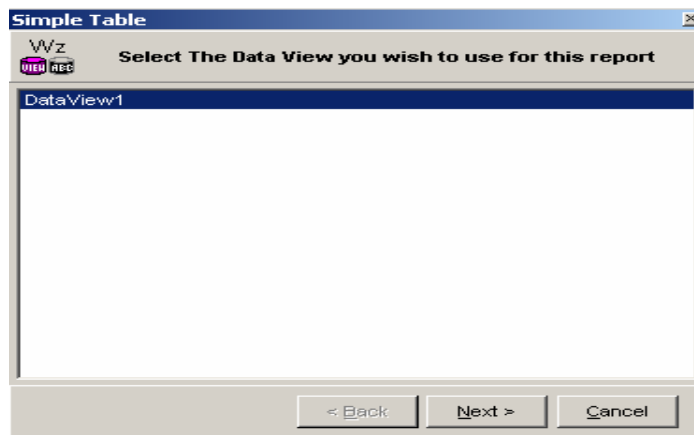
8. Dari jendela Data Connention, pilih Direct Data View dan klik Next.



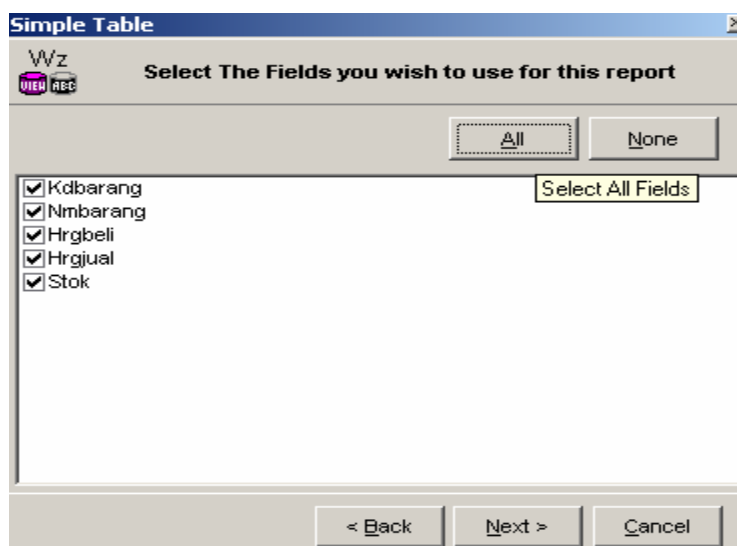
9. Klik Finish
10. Pada Jendeela Onbject tri View, Klik report Library, dan klik report 1, seperti yag terlihat pada gambar berikut.



11. Klik Report1, pada properties pilih name dan ganti menjadi Lapbarang
12. Dari menu Tools, Pilih Riport Wizard, Simple table

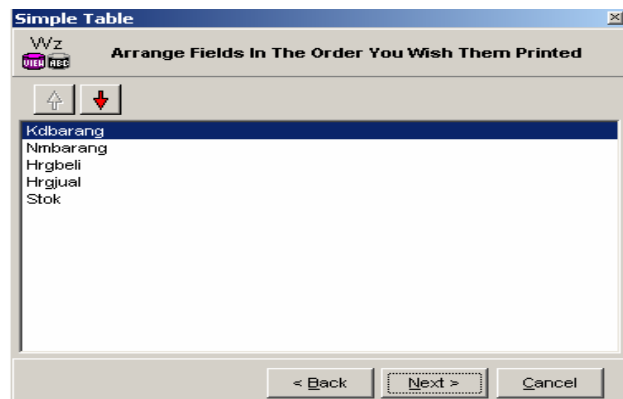


13. Klik Next



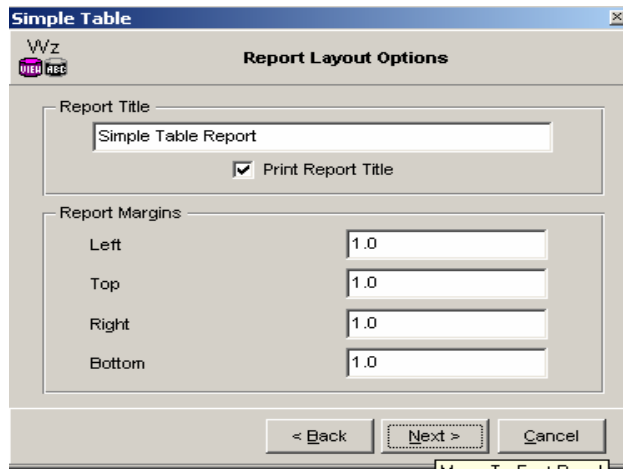
14. Klik All, jika ingin menampilkan Data Dalam table pada jendeela Report, tetapi jika hanya sebagian field dijadikan report, cek list satu per satu field yang dibutuhkan.

15. klik next



16. Klik mode pengurutan data, secara ascending maupun descending.

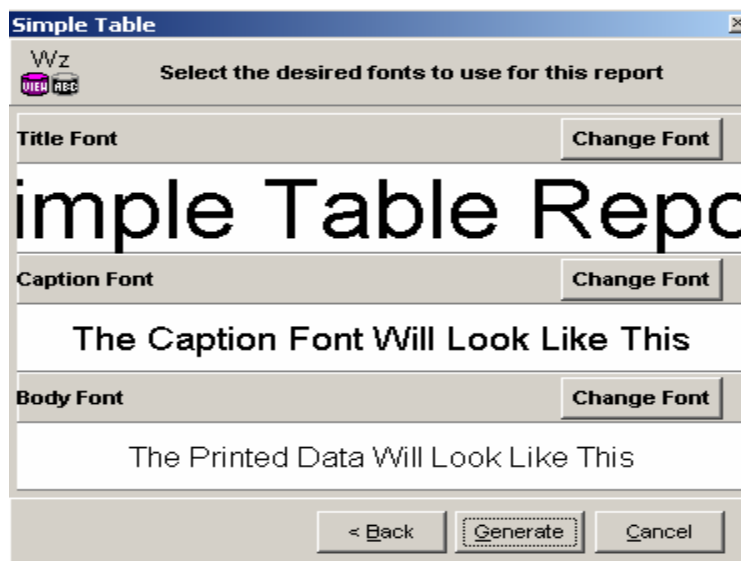
17. klik Next



The dialog box titled "Simple Table" with a subtitle "Report Layout Options". It contains a "Report Title" section with a text field containing "Simple Table Report" and a checked checkbox labeled "Print Report Title". Below this is a "Report Margins" section with four input fields: "Left" (1.0), "Top" (1.0), "Right" (1.0), and "Bottom" (1.0). At the bottom are three buttons: "< Back", "Next >", and "Cancel".

18. Pada text Report Title, hapus dan ganti dengan judul laporan yang diinginkan.misal = Laporan Data Barang.

19. Klik Next



The dialog box titled "Simple Table" with a subtitle "Select the desired fonts to use for this report". It has three sections: "Title Font" with a "Change Font" button and a preview of "imple Table Repo"; "Caption Font" with a "Change Font" button and a preview of "The Caption Font Will Look Like This"; and "Body Font" with a "Change Font" button and a preview of "The Printed Data Will Look Like This". At the bottom are three buttons: "< Back", "Generate", and "Cancel".

20. Pada Wizard ini berfungsi untuk mengganti pilihan font untuk setiap masing = masing bagian dari report. Caranya klik pada command **change Font**

21. Jika sudah selesai klik generate untuk mengakhiri perintah. Dan perhatikan perubahan tampilan pada jendela report seperti berikut.

DataView1Region: DataView1TitleBand (BGDRgb 1PC)				
Simple Table Report				
DataView1Region: DataView1Band (BGDRgb 1PC)				
Kdbarang	Nmbarang	Hrgbeli	Hrgjual	Stok
DataView1Region: DataView1DataBand (Master 1PC)				
{Kdbarang}	{Nmbarang}	{Hrgbeli}	{Hrgjual}	{Stok}

Memanipulasi Tampilan

a. Menggeser Document Laporan

- Klik Document (pada document yang berwarna gelap), sehingga setiap sisi document diberikan handle kecil berwarna hijau.
- Klik dan drag mouse sesuai dengan tampilan yang diinginkan.

b. Mengganti text pada Judul

- Klik text (misal = **Kdbarang**), pada tab **Data View1Region, Data View1 Band**. Klik text pada jendela properties dan ganti tulisan sesuai dedngan tampilan yang diinginkan. (misal = Kode Barang)

Catatan = ada 2 text dengan tulisan yang sama, text pada tab pertama (Data View 1 band) ini hanya merupakan judul, jadi pad tekx disini tulisan bisa diganti maupun diedit. Tetapi untuk teks pada Tab ke 2 (DataView1DataBand) jangan diganti karena berhubungan dengan nama field yang ada pada tabel.

c. Memperlebar jendela setiap tab

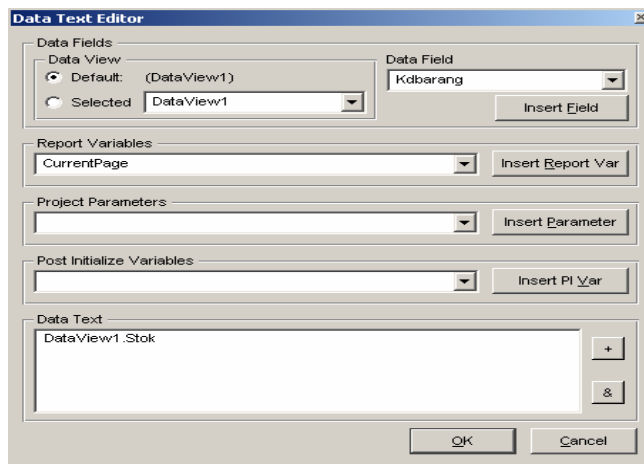
- Klik mouse anda pada judul tab yang ingin diperlebar jendelanya, perhatikan pada sub tab tersebut akan ditampilkan handle kecil berwarna hijau
- Klik hande tersebut, dan drag mouse dan geser untuk mendapatkan ruang jendela sesuai dengan kebutuhan.

d. Membuat garis

- Komponen garis ada pada komponen drawaing pada tab Drawing.
- Klik icon garis (sesuai dengan kebutuhan), drag pada document laporan sehingga diperoleh desain laporan yang diinginkan.
- Ulangi langkah tersebut untuk membentuk laporan lebih kompleks.

e. Membuat teks untuk menjumlah data

- Klik icon Calc teks component (pada komponen Report)
- Drag dan desain pada document dibawah stok.
- Setting jendela properties sebagai berikut =
 - Untuk Data Field
 1. klik (...)



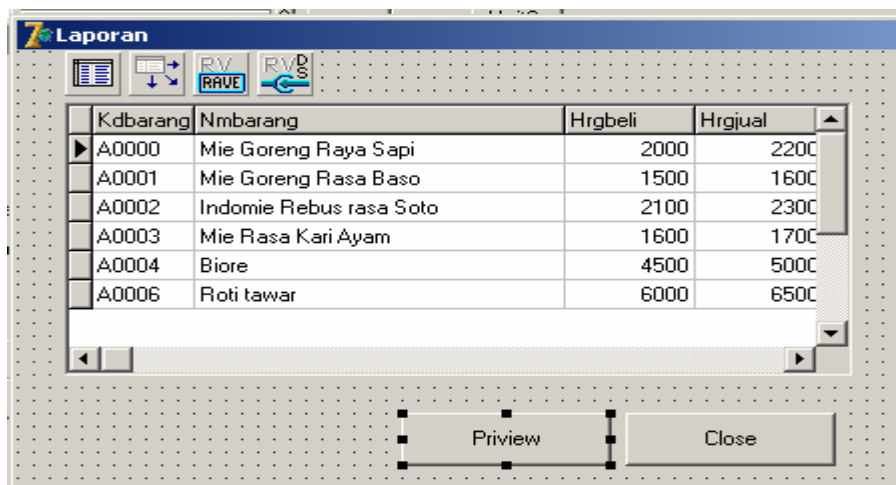
2. Pada Combo Data Filed pilih Field Stok
 3. Klik Command Insert Field
 4. Klik OK
- Pada Data View pilih data view1
 - Pada Controller = pilih Data View1DataBand

f. Menjalankan Laporan

- o Tekan F9
- o Pilih option priview
- o Klik OK

15.2. Mencetak laporan Melalui Form

1. Kembali ke jendela Project Delphi, pada form rev Project
2. tambahkan dua button dang anti nama masing menjadi priview dan close



3. Ketikkan program sebagai berikut

```
procedure TForm8.Button1Click(Sender: TObject);  
begin  
    rvproject1.ExecuteReport('lapbarang');  
end;
```

```
procedure TForm8.Button2Click(Sender: TObject);  
begin  
    close  
end;
```

BAB XVII

MENU UTAMA

a. Form Setelah Dijalankan

notrans	imbeli	kdbarang	nmbarang	hrgjual	subtotal
00001	1	A0001	Mie Goreng Rasa Baso	1600	1600
00001	1	A0002	Indomie Miebus rasa Soto	2300	2300
00002	1	A0002	Indomie Miebus rasa Soto	2300	2300

b. Langkah - langkahnya

1. tambahkan form baru
2. pada jendela properties ganti caption menjadi = Sistem Informasi TOKO
3. Pada Windows State = pilih maximized
4. tambahkan icon main Menu (komponen Standart), tempatkan pad sembarang form
5. Double Klik Icon main menu, sehingga ditampilkan jendela sebagai berikut:

5. ketikkan Desain menu Utam Seperti tabel berikut

Master	Transaksi	Laporan	Exit
- Cari Data	Penjualan	- Laporan Barang	
- Tambah Data	Retur Penjualan		
- Edit Data			

6. Setelah Selesai klik tanda (X), untuk kembali ke form menu utama
7. Double Click Setiap object dan ketikan program sebagai berikut :

```
procedure TForm9.Exit1Click(Sender: TObject);  
begin  
close;  
end;
```

```
procedure TForm9.CariData1Click(Sender: TObject);  
begin  
form2.Show;  
end;
```

```
procedure TForm9.ambahData1Click(Sender: TObject);  
begin  
form6.Show;  
end;
```

```
procedure TForm9.EditData1Click(Sender: TObject);  
begin  
form7.show;  
end;
```

```
procedure TForm9.penjualanClick(Sender: TObject);  
begin  
form13.show;  
end;
```

```
procedure returpenjualanClick(Sender: TObject);  
begin  
form14.show;  
end;
```

Catatan :

Pada saat form dijalankan, dan kita mengaktifkan salah satu pilihan menu. Ada dua hal bisa terjadi. Pertama form langsung bisa diaktifkan dan yang kedua form belum bisa diaktifkan. Ketika form tidak bisa langsung diaktifkan, dan muncul pertanyaan, maka pilih yes, pada jendela object treeView, pilih uses dan double click unit yang akan diaktifkan, setelah itu coba dijalankan ulang.