
BAB 1

Kebutuhan Sistem & Pengenalan Software

1.1 Software Pendukung

Pemrograman database dengan Visual Basic .NET 2005 dan MySQL dapat dilaksanakan jika beberapa kebutuhan software telah ter-install dalam komputer, antara lain:

- **Visual Studio 2005**

Dalam Visual Studio 2005 terdapat bahasa pemrograman Visual Basic 2005 yang akan digunakan dalam buku ini. Visual Studio 2005 bisa diganti dengan produk lain yang sejenis, seperti **Visual Studio 2008** atau **Visual Basic 2005/2008 Express Edition** (file dapat diunduh melalui alamat <http://www.microsoft.com/express/vb/Default.aspx> yang masih berbentuk file ISO, sehingga Anda harus membakarnya terlebih dahulu ke DVD, kemudian menginstallnya ke komputer). **Visual Basic 2005/2008 Express Edition** merupakan edisi gratis yang diperuntukkan bagi pengembang.

- **Database MySQL 5.1**

Database yang akan digunakan dalam buku ini adalah database **MySQL 5.1**. Untuk memperoleh MySQL versi terbaru bisa diunduh melalui situs web resmi MySQL, yaitu <http://dev.mysql.com/downloads/>.

- **MySQL-Connector-Net**

Merupakan abstraksi class yang mengenalkan fitur-fitur MySQL pada lingkungan bahasa pemrograman Visual Basic 2005. MySQL-Connector telah tersedia di dalam CD penyerta dengan nama ***mysql-connector-net-5.2.5.zip*** karena file masih dalam bentuk termampatkan maka Anda harus menguraikan terlebih dahulu kemudian menginstallnya.

- **Navicat 8.1.16 Lite (non commercial)**

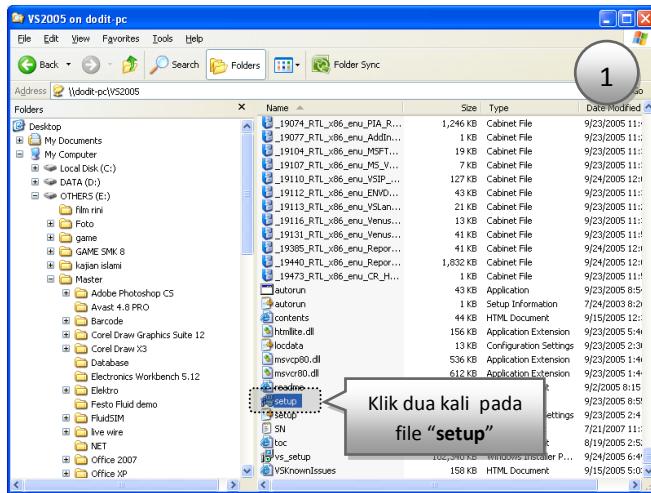
Merupakan software bantu yang dibuat oleh **Premiumsoft** bertujuan untuk memudahkan kita dalam memanipulasi *database*, *view*, *store procedure*, *store function*, *grant user*, menciptakan table, membuat query, dan lain sebagainya. Navicat juga sudah tersedia dalam CD penyerta buku. Selain Navicat Anda bisa pula menggunakan software alternatif seperti **DreamCoder**, **Query Browser**, **HeidiSQL**, dan lain sebagainya.

1.2 Instalasi Software

Berikut ini tata cara instalasi software pendukung yang dibutuhkan. Anda tinggal mengikuti tahapan-tahapan sesuai dengan tuntunan yang diuraikan di bawah ini.

1.2.1 Visual Studio 2005

Klik dua kali pada file **setup.exe** dari master Visual Studio 2005 dari Windows Explorer.

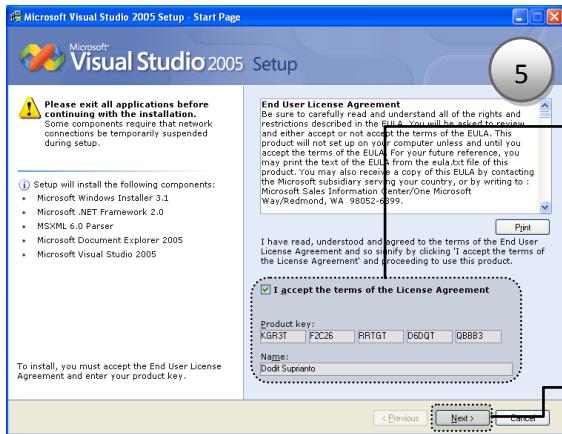




Proses loading instalasi komponen.



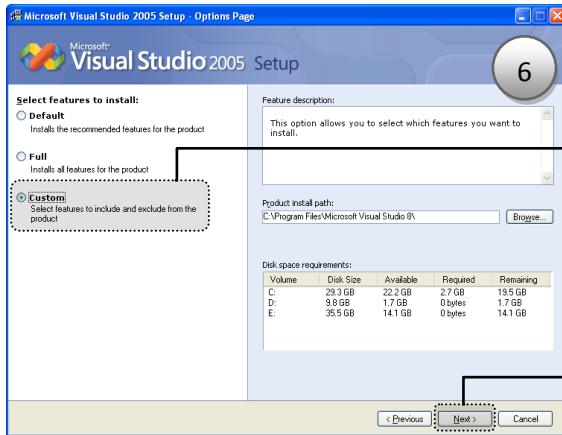
Klik tombol "Next" untuk melanjutkan.



Centang pada pernyataan “**I accept the terms of the License Agreement**” sebagai tanda persetujuan atas penggunaan Visual Studio 2005.

Masukkan **Product Key** sebanyak 16 digit, serta Nama Anda.

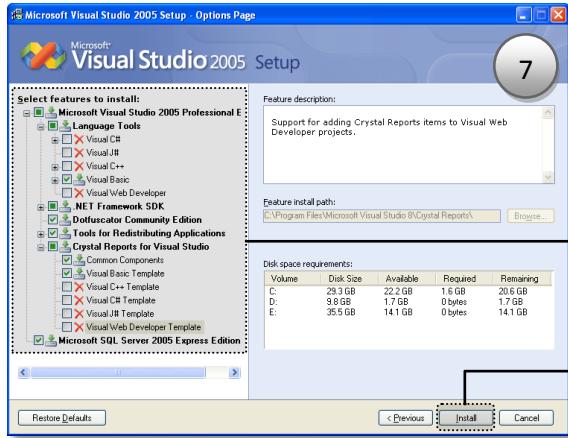
Lanjutkan dengan meneklik tombol “**Next**”



Pilih pada “**Custom**” untuk menginstall beberapa bagian dari Visual Studio 2005.

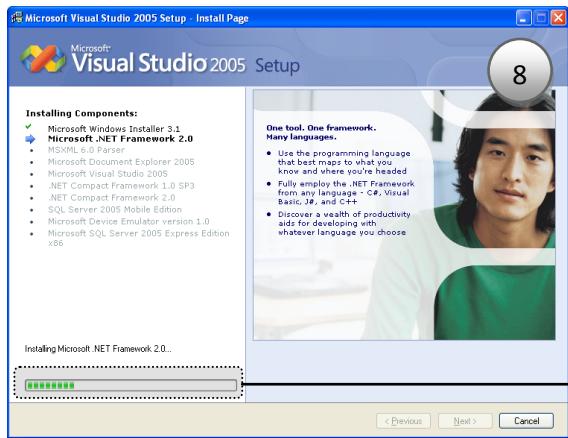
Kasus kali ini kita akan menginstall semua fitur yang berhubungan dengan Visual Basic 2005 saja.

Lanjutkan dengan mengklik tombol “**Next**”



Pilih fitur-fitur yang dibutuhkan saja. Seperti tampak pada gambar.

Lanjutkan dengan meneklik tombol “Install”



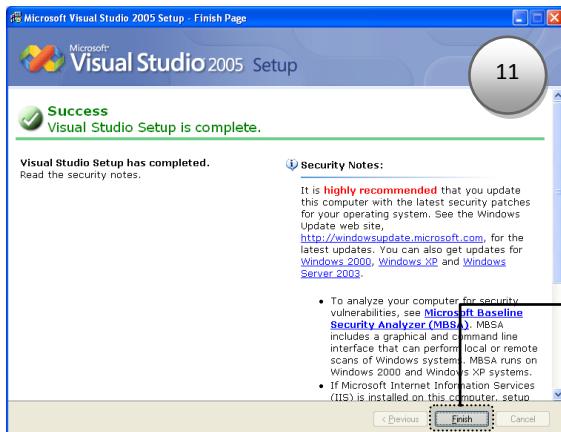
Proses instalasi komponen



Klik tombol “Restart Now” untuk memperoleh efek dari instalasi sebelumnya.



Jika setelah Restart tidak terjadi instalasi lanjutan secara otomatis, maka lakukan instalasi ulang, sama seperti pada tahap 1 sampai 7, hingga mencapai tahap 10 ini.



Instalasi telah selesai. Klik tombol “Finish”

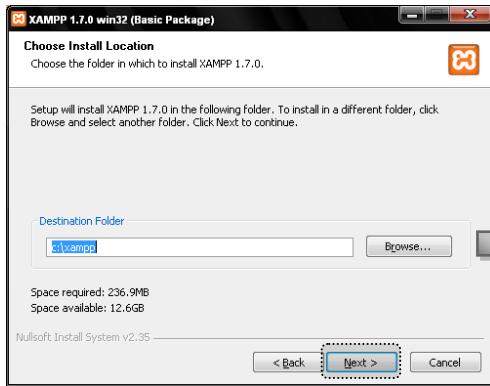
1.2.2 MySQL - XAMPP



Jalankan file **xampp-win32-1.7.0-installer.exe** hingga muncul pilihan bahasa pengantar instalasi, pilih bahasa inggris. Kemudian tekan tombol “OK”.



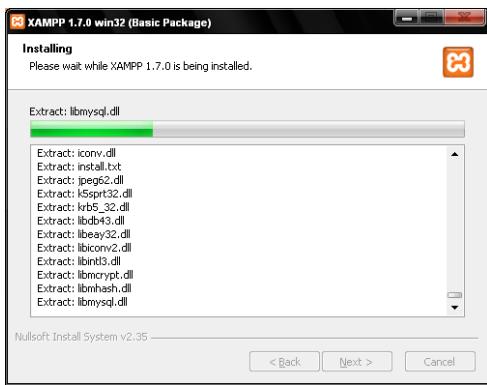
Window “**Welcome to the XAMPP 1.7.0 Setup Wizard**” berisi ucapan selamat datang di instalasi setup XAMPP 1.7.0. Lanjutkan dengan menekan tombol “**Next**”.



Window “**Choose Install Location**” menanyakan tentang folder peletakkan XAMPP, secara normal file-file XAMPP akan diletakkan pada folder **c:\xampp**. Alamat folder dapat diubah dengan mengeklik tombol “**Browse**” kemudian tentukan folder yang dikehendaki. Tekan tombol “**Next**” untuk melanjutkan proses instalasi.



Window “**XAMPP Options**” menampilkan pilihan-pilihan instalasi. Centang pada pilihan “**Install Apache as service**” dan “**Install MySQL as service**” pada bagian “**SERVICE SECTION**” yang berarti kedua pilihan tersebut (Web Server dan Database MySQL) akan dijalankan secara otomatis setiap kali komputer dinyalakan. Lanjutkan dengan menekan tombol “**Install**”.

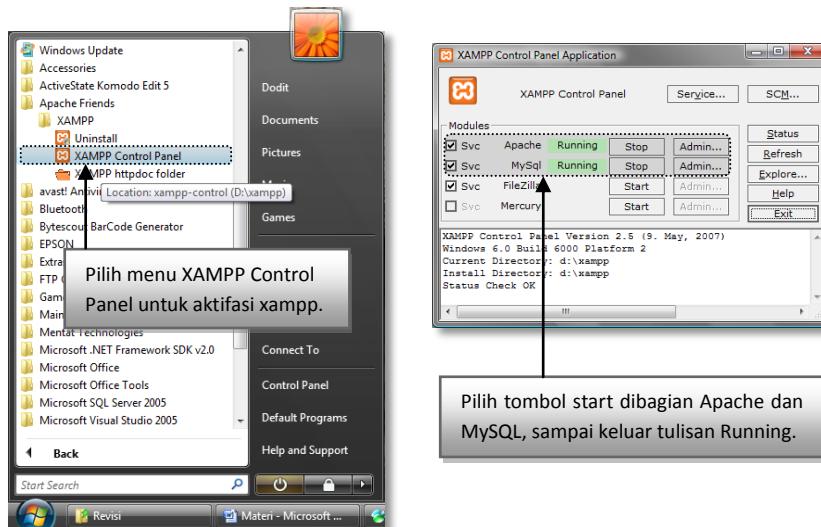


Window “**Installing**” menandakan bahwa proses instalasi sedang berlangsung, biarkan saja sampai tanda *progress bar* berwarna hijau berjalan sampai pada posisi sebelah kanan akhir.

Jika keluar window seperti tampak di bawah, maka proses instalasi telah berhasil dan selesai. Lanjutkan dengan menekan tombol “**Finish**”.

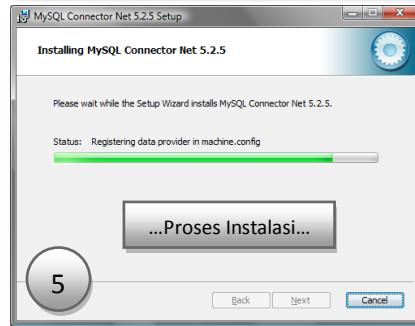
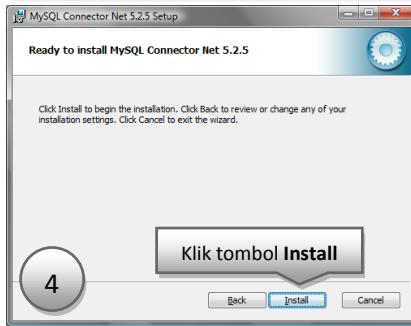
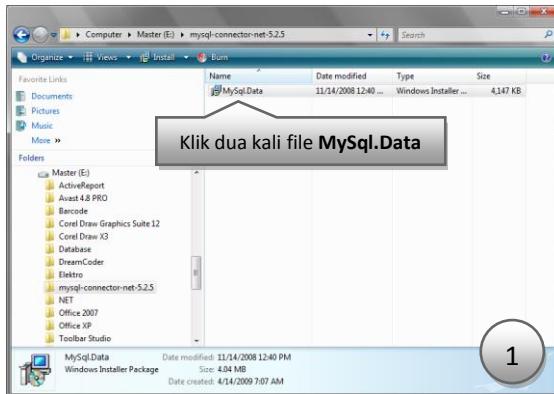


Agar MySQL dapat bekerja maka **xampp** harus diaktifkan. Cara pengaktifkannya sangat mudah, buka menu **Start -> All Programs->Apache Friends->XAMPP->XAMPP Control Panel**, seperti tampak gambar berikut ini:



1.2.3 MySQL Connector .NET

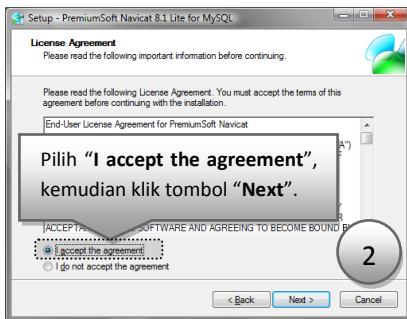
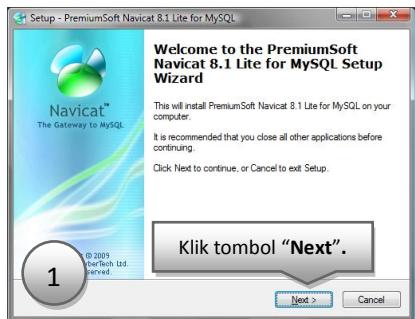
klik dua kali file
MySQL.Data untuk
menginstall **MySQL**
Connector .NET.
Selanjutnya ikuti
langkah instalasinya:

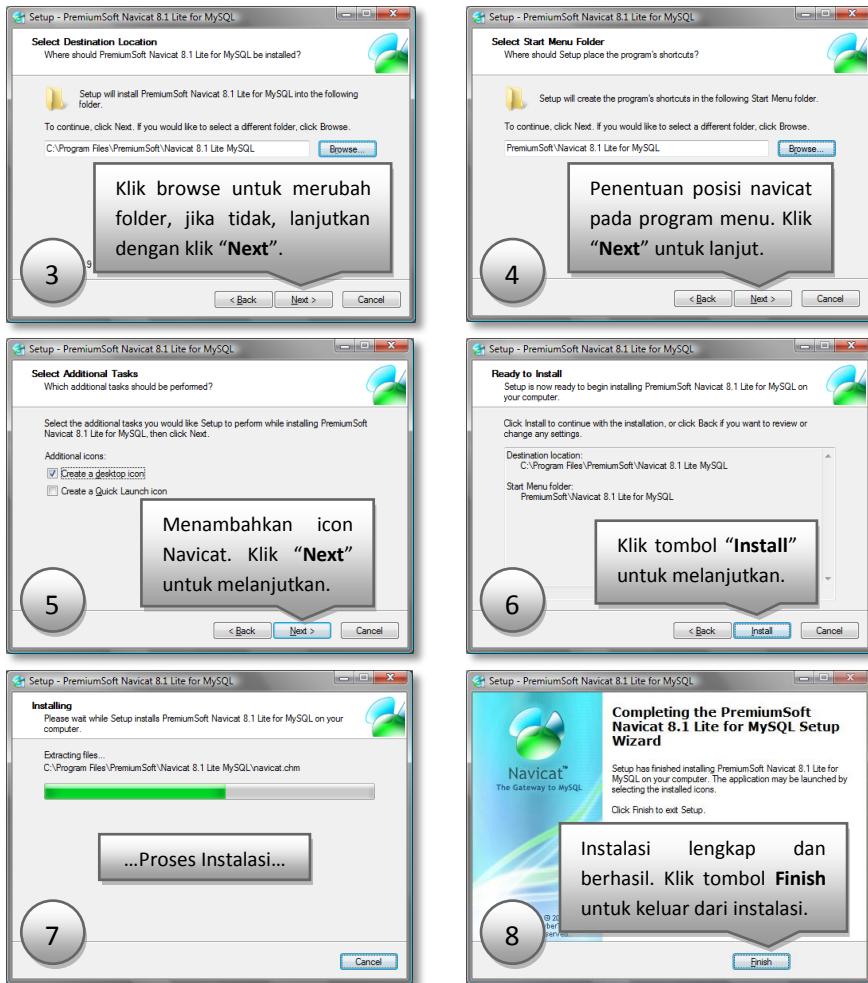




1.2.4 Navicat 8.1.16 Lite

Jalankan file **navicat8lite_mysql_en** yang tersedia dalam CD penerta buku dengan mengkliknya dua kali. Selanjutnya ikuti langkah-langkah instalasinya seperti berikut ini:



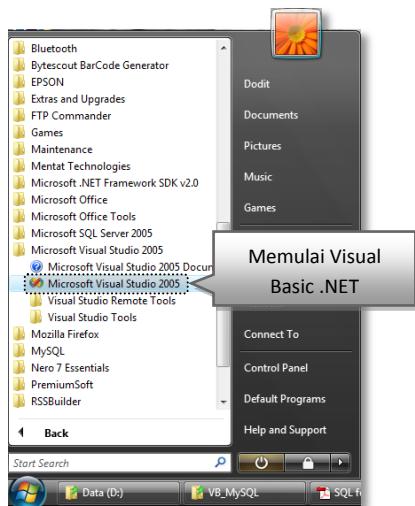


1.3 Mengenal IDE Visual Basic 2005

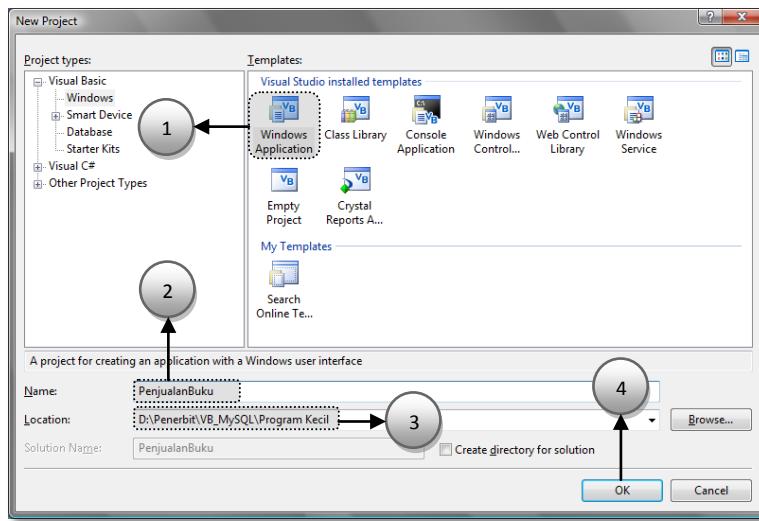
Bagi programer pemula perlu mengenal lingkungan Visual Basic .NET 2005, terutama IDE (*Integrated Developing environment*). Di buku ini tidak dijelaskan terlalu dalam hanya sebatas keperluan saja pemrograman

database saja. Hal lain yang berkaitan dengan IDE Visual Basic .NET 2005 dapat dipelajari sendiri sambil terus mencoba.

Buka Visual Studio 2005 dari menu **Start->All Programs->Microsoft Visual Studio 2005->Microsoft Visual Studio 2005**. Seperti tampak gambar di bawah ini:



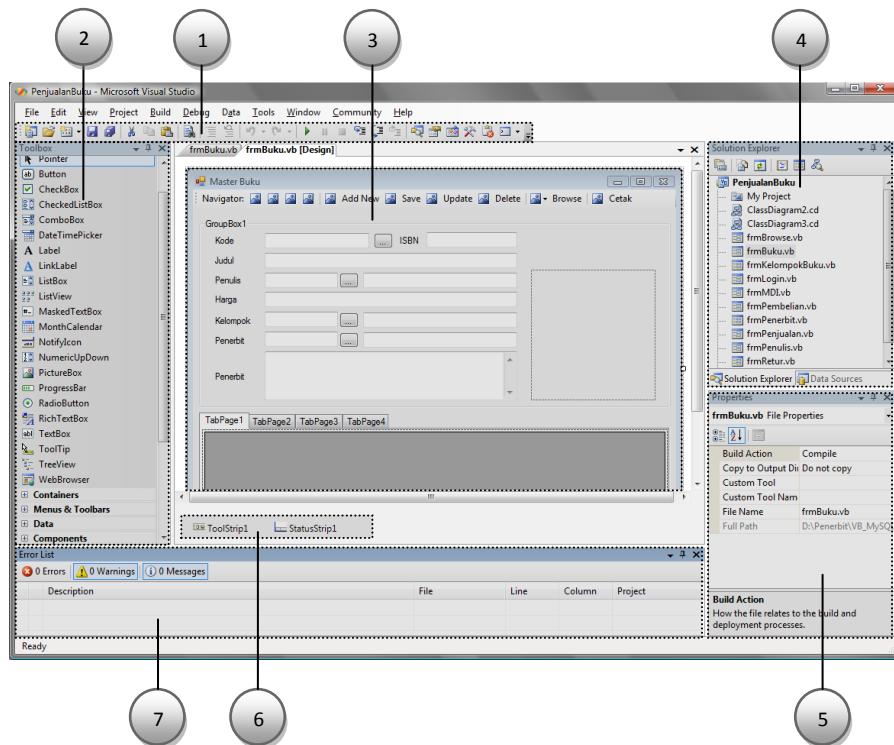
Setelah Visual Studio 2005 terbuka, lanjutkan dengan membuat project baru dari menu **File->New->Project**, kemudian akan keluar *windows dialog* yang meminta kita untuk menentukan nama project baru yang sedang dibuat. Pilih juga **Visual Studio installed templates** dengan **Windows Application**, lanjutkan dengan menekan tombol **OK** seperti tampak gambar berikut ini:



Keterangan gambar:

1. Pilih Visual Studio installed templates dengan **Windows Application** untuk menciptakan aplikasi form windows.
2. Cantumkan nama project yang sedang dibuat dengan nama **PenjualanBuku**.
3. Penentuan lokasi folder dimana file-file project disimpan.
4. Tekan tombol **OK** untuk melanjutkan ke tahap berikutnya.

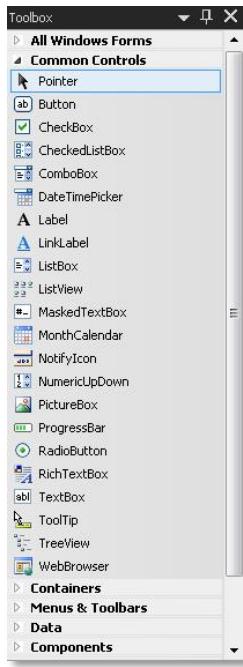
Setelah menekan tombol **OK** maka project baru yang bernama **PenjualanBuku** telah terbentuk, dan otomatis terbentuk pula sebuah Form Windows baru. Visual Basic .NET 2005 juga akan menyediakan kelengkapan perangkat penunjang pembuatan project tersebut. Apa saja yang perlu diketahui? bisa dilihat dari gambar berikut:



Keterangan gambar:

1. **Toolbar**, terdiri dari ikon-ikon berlaku sebagai jalan pintas (*shortcut*) pengganti menu *pulldown*. Ikon **toolbar** berisi perintah-perintah yang sering digunakan oleh programer sehingga Anda lebih cepat mengoperasikan perintah-perintah yang ada.

2. **Toolbox**, berisi kontrol object terutama pembentuk antarmuka (*interface*) windows seperti **textbox**, **panel**, **button**, **groupbox**, dan lain sebagainya.
3. **Form Design**, sebuah Form baru akan terbentuk dengan sendirinya saat project dibuat untuk kali pertama.
4. **Solution Explorer**, terdiri dari object-object pendukung project, seperti **form**, **module**, **report**, **DataControl**, dan lain sebagainya.
5. **Properties**, menampilkan *property* setiap object yang terpilih. Dari *Windows Property* Anda dapat mengubah atau mengatur nilai masing-masing *properties object*.
6. **Status object**, menampilkan daftar kontrol yang telah digunakan oleh Form bersangkutan. Dari kasus di atas kontrol yang digunakan adalah **StatusStrip1** dan **ToolStrip1**.
7. **Error List**, merupakan Window yang akan menampilkan pesan tertentu jika terjadi kesalahan atau *error* pada saat proses pengembangan program atau kompilasi program.



Toolbox, berisi beberapa jenis kontrol misalnya: **TextBox**, **Button**, **Label**, **Menustrip**, **Toolbars** dan masih banyak lagi. Isi ToolBox sendiri sangat banyak, Anda bisa mencoba sendiri kontrol-kontrol yang tersedia, gunakan sesuai kebutuhan.

ToolBox terbagi menjadi beberapa tab, antara lain:

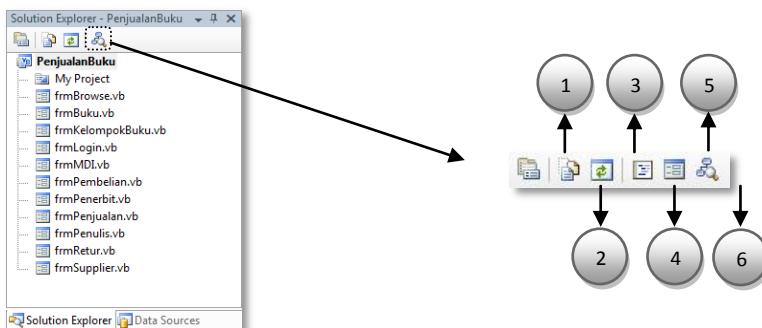
- **All Windows Form**, semua kontrol Toolbox.
- **Common Controls**, kontrol yang paling sering digunakan.
- **Containers**, kontrol yang berkaitan dengan pengelompokkan kontrol-kontrol dalam Form.
- **Menu & Toolbars**, kontrol yang berhubungan dengan penanganan menu **pull down** dan **toolbars**.
- **Data**, kontrol yang menangani akses data ke database dan manipulasi database.
- **Components**, kontrol yang berhubungan dengan masukkan atau keluaran data baik berupa *image*, *bit data*, penanganan *cursor*, dan lain sebagainya.
- **Printing**, kontrol yang berhubungan dengan *open dialog* pencetakan, *preview* (tampilan hasil cetak), dan lain sebagainya.
- **Dialog**, kontrol yang berhubungan dengan *open dialog* direktori, *open*

dialog pemilihan jenis font, open dialog penamaan file, dan lain sebagainya.

- **Crystal Reports**, kontrol yang berhubungan dengan pencetakan laporan. Pada Visual Basic 6 komponen **Crystal Reports** tidak termasuk dari bagian Visual Basic karena **Crystal Report** tergolong perangkat lunak yang berasal dari pihak ketiga, namun sekarang sudah tergabung menjadi satu dengan Visual Studio 2005.

Anda tidak perlu bingung dengan kontrol-kontrol di atas karena pada kenyataannya hanya sedikit komponen yang akan digunakan.

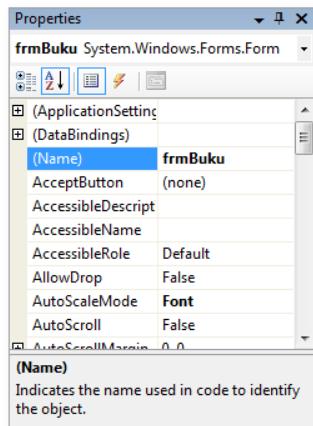
Solution Explorer, merupakan bagian-bagian pembentuk project yang sedang dibangun, bisa berupa form, kode program, module, class, dan lain-lain. Jika kita menambahkan bagian baru dalam project maka secara otomatis **Solution Explorer** akan menambahkan bagian tersebut ke daftar **Solution Explorer**.



Keterangan gambar:

1. **Properties**, ikon yang akan menuju ke properties Form.
2. **Show All Files**, ikon yang akan menampilkan seluruh daftar file yang terlibat dalam project.

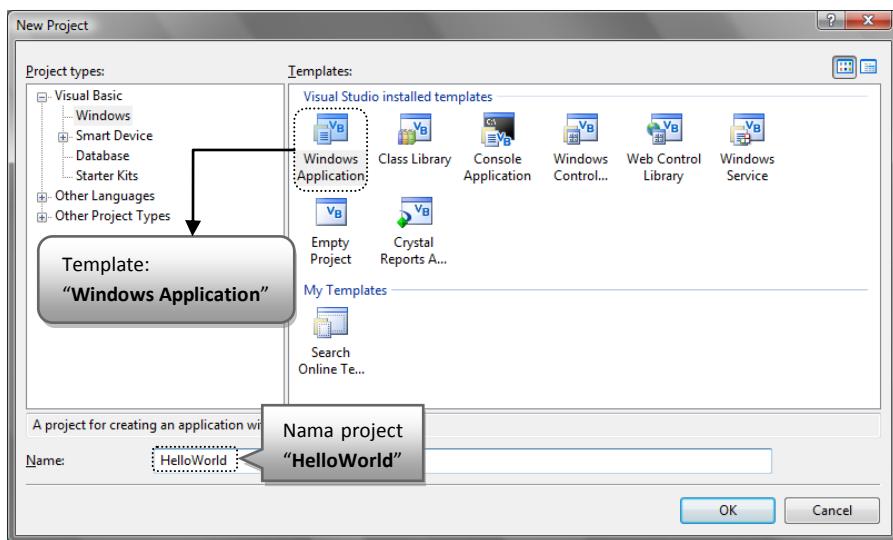
3. **Refresh**, ikon yang bertujuan untuk menyegarkan kembali file-file yang ada dalam project.
4. **View Code**, ikon yang akan menuju ke editor kode program dari form dan module.
5. **View Designer**, ikon yang akan menuju ke mode desain form dari editor kode program.
6. **View Class Diagram**, ikon yang menuju ke tools untuk menampilkan *class model* dalam bentuk diagram. Untuk saat ini **View Class Diagram** tidak digunakan dalam buku ini.



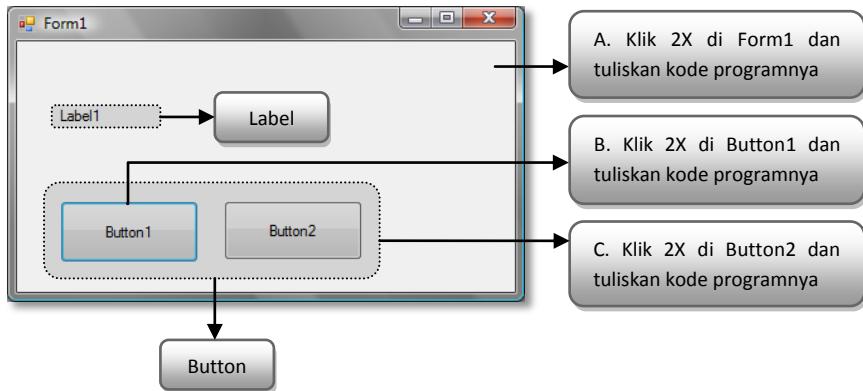
Properties, berkaitan dengan atribut/properties salah satu bagian dari kontrol (form, textbox, label, dll). Gambar disamping merupakan properties dari form buku yang bernama **frmBuku**. Nilai-nilai atribut/properties dapat diubah sesuai kebutuhan.

1.3.1 Program Sederhana “Hello World”

Agar lebih semangat! Sekarang mari kita buat aplikasi sederhana dengan Visual Basic 2005. Buka Visual Basic 2005, klik menu **File->New->Project**, beri nama project dengan **HelloWorld**, pilih template dengan **Windows Application**.



Tambahkan satu **Label** dan dua **Button** pada **Form1**, kemudian klik dua kali pada form, tuliskan kode programnya seperti berikut ini:



A. Event Form1_Load

```
Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load
    Me.Text = "Hello World"
    Label1.Text = ""
    Button1.Text = "OK"
    Button2.Text = "Clear"
End Sub
```

B. Event Button1_Click

```
Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Button1.Click
    Label1.Text = "Hello World! " & vbCrLf & "ini program pertamaku dengan Vusial Basic
2005."
End Sub
```

C. Event Button2_Click

```
Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Button2.Click
    Label1.Text = ""
End Sub
```

Kode Lengkap

```
Public Class Form1

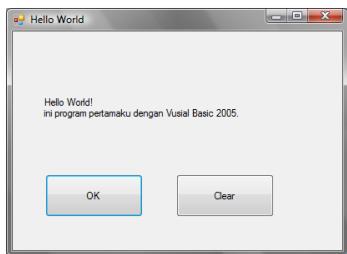
    Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles MyBase.Load
    Me.Text = "Hello World"
    Label1.Text = ""
    Button1.Text = "OK"
    Button2.Text = "Clear"
End Sub

    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Button1.Click
    Label1.Text = "Hello World! " & vbCrLf & "ini program pertamaku dengan Vusial Basic
2005."
End Sub

    Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Button2.Click
    Label1.Text = ""
End Sub

End Class
```

Jalankan programnya dengan menekan tombol keyboard **F5** (atau melalui menu **Debug->Start Debugging**). Untuk melihat hasilnya klik **Button1** dan **Button2**. Tentunya Anda dapat dengan mudah menarik kesimpulan dari kode program tersebut.

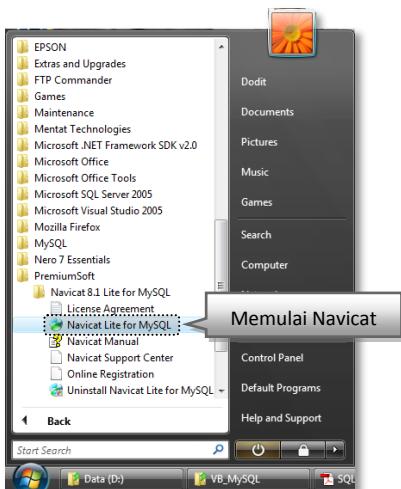


1.4 Program Navicat Untuk MySQL

Navicat adalah software yang berfungsi untuk memudahkan kita saat menciptakan (*DDL, data definition language*) dan memanipulasi (*DML, data manipulation language*) database MySQL seperti menciptakan table, menghapus table, memasukkan baris, menghapus baris, mengubah baris, menampilkan data, menciptakan user, dan lain sebagainya.

Dengan navicat Anda tidak harus hebat dalam penguasaan script SQL karena navicat menyediakan sarana untuk memanipulasi database dengan mudah dan ramah (*user friendly*). Sebagai contoh, untuk memasukkan baris data maka perintah SQL yang digunakan adalah pernyataan **INSERT**, dengan navicat Anda dapat memasukkan baris data melalui **grid** layaknya *spread sheet Excel*.

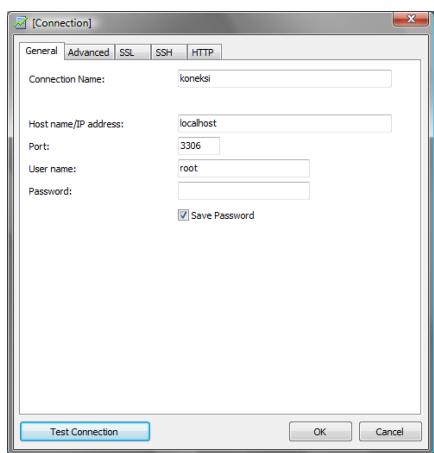
1.4.1 Koneksi Ke Database MySQL



Kita belum bisa menggunakan fitur-fitur yang ada dalam Navicat sebelum software tersebut terkoneksi dengan MySQL. Langkah pertama yang harus dilakukan adalah mengkoneksikan Navicat ke MySQL.

Buka software navicat melalui menu **Start->All Programs->PremiumSoft->Navicat 8.1 Lite for MySQL->Navicat Lite for MySQL** seperti tampak gambar disamping.

Setelah navicat terbuka selanjutnya adalah menciptakan koneksi ke database MySQL. Pilih menu **File->New Connection** atau bisa langsung melalui ikon **Toolbar Connection**.

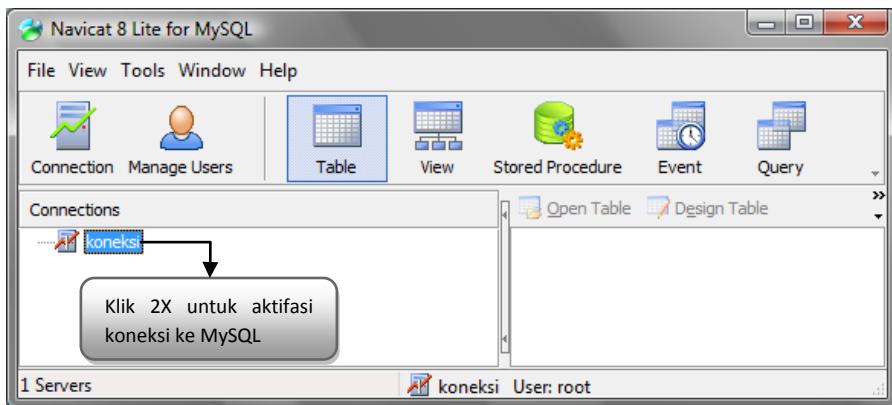


Tab **General** berisi parameter umum yang harus diisi, karena menggunakan database dengan distribusi XAMPP pada komputer lokal maka konfigurasinya adalah:

- Connection Name: **koneksi** (pemberian nama bebas).
- Hostname/IP Adress: **localhost** atau **172.0.0.1**
- Port: **3306**, port default untuk koneksi database MySQL.
- User: **root**.
- Password : <biarkan kosong>



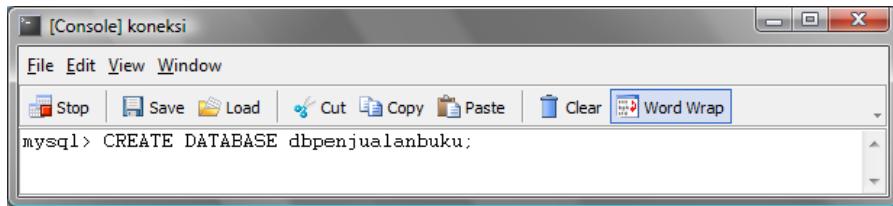
Pengujian koneksi MySQL dilakukan dengan cara mengeklik tombol **Test Connection**. Jika keluar pesan seperti disamping berarti koneksi ke MySQL sukses. Lanjutkan dengan klik tombol **OK**.



1.4.2 Menciptakan Database

Sekarang kita akan menciptakan database MySQL baru yang diberi nama **dbpenjualanbuku**. Buka menu **Tools->Console** atau melalui tombol keyboard **F6**. Kemudian tuliskan pernyataan SQL berikut:

```
mysql> CREATE DATABASE dbpenjualanbuku;
```



[Console] koneksi

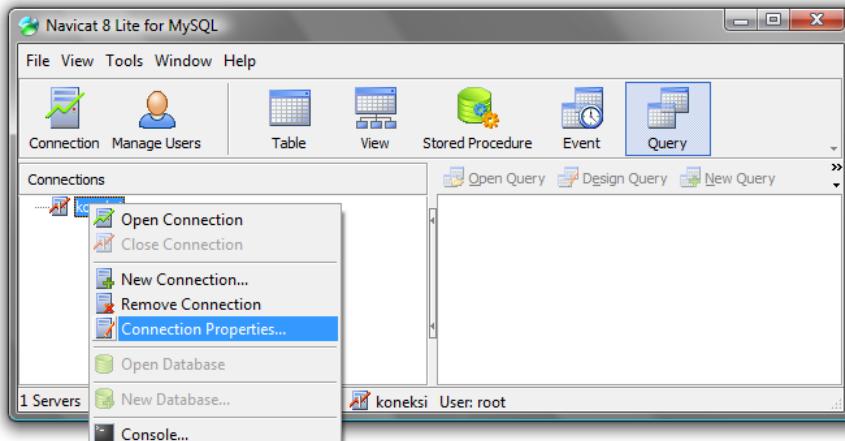
File Edit View Window

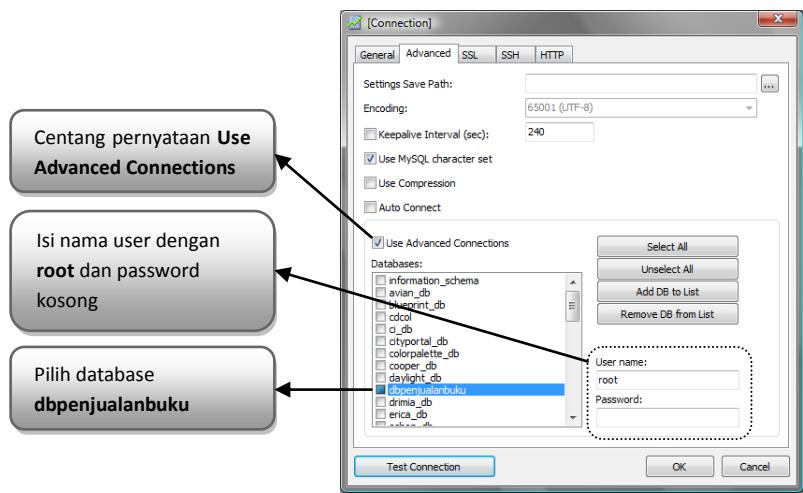
Stop Save Load Cut Copy Paste Clear Word Wrap

```
mysql> CREATE DATABASE dbpenjualanbuku;
```

Untuk menjalankannya tekan tombol keyboard **Enter**. Sekarang Anda telah memiliki database **dbpenjualanbuku**.

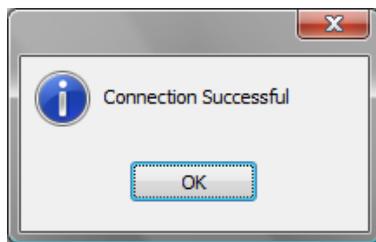
Agar koneksi MySQL terfokus hanya pada database **dbpenjualanbuku** maka diperlukan sedikit konfigurasi pada *property* koneksi. Klik kanan pada **koneksi** kemudian pilih **Connection Properties**.





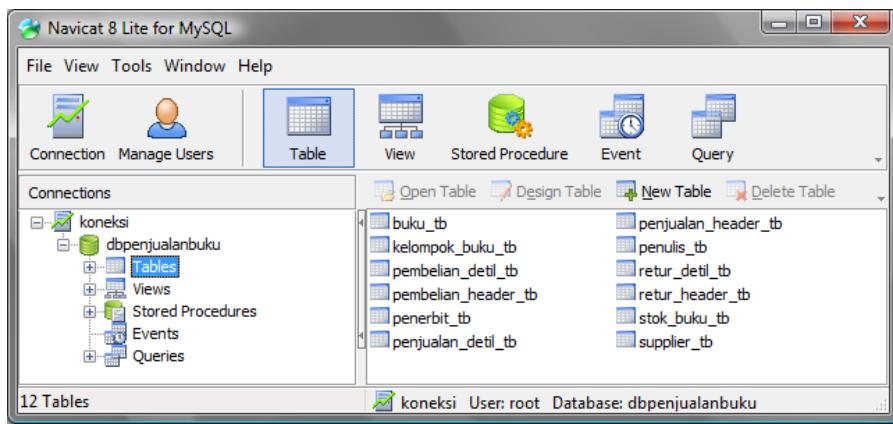
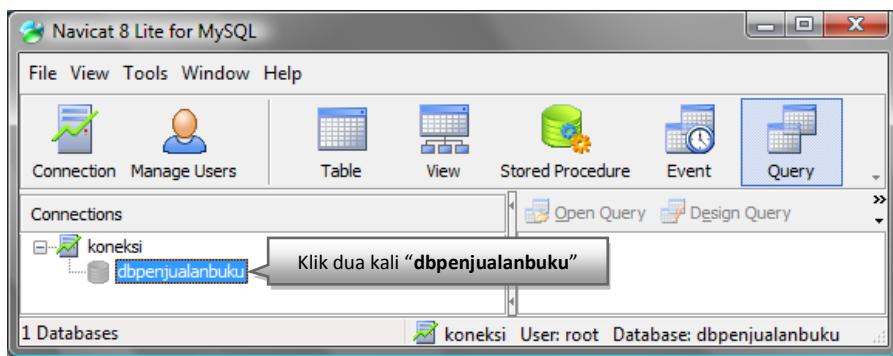
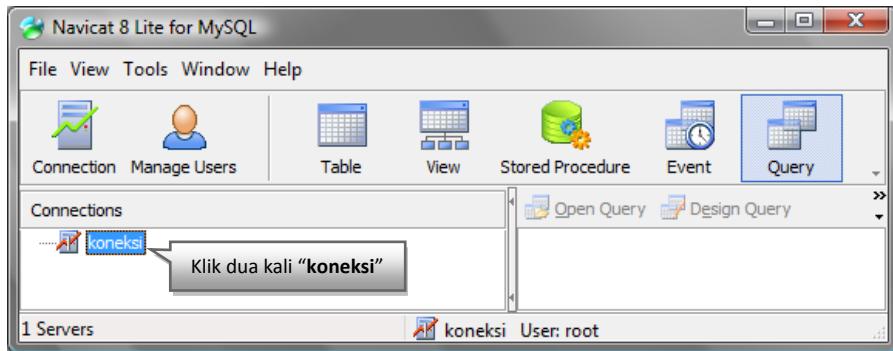
Jika hanya menitikberatkan pada koneksi ke satu database maka klik tab **Advanced**. Parameter yang harus diisi adalah:

- Databases: **dbpenjualanbuku**
- User name: **root**
- Password: <biarkan kosong>



Untuk memastikan apakah navicat sudah terkoneksi dengan database **dbpenjualanbuku**, klik tombol **Test Connection**. Jika sukses lanjutkan dengan mengeklik tombol **OK** dan koneksi pun terbentuk.

Setelah itu klik dua kali pada connection “**koneksi**” dan database “**dbpenjualanbuku**” hingga keluar daftar *table*, *view*, *store procedure*, *events* dan *query*, seperti gambar di bawah ini:



1.4.3 Mengenal Toolbar Navicat

Toolbar Navicat terdiri atas item **Connection**, **Manage Users**, **Table**, **View**, **Store Procedure**, **Event** dan **Query**, masing-masing item terdiri atas beberapa sub item.

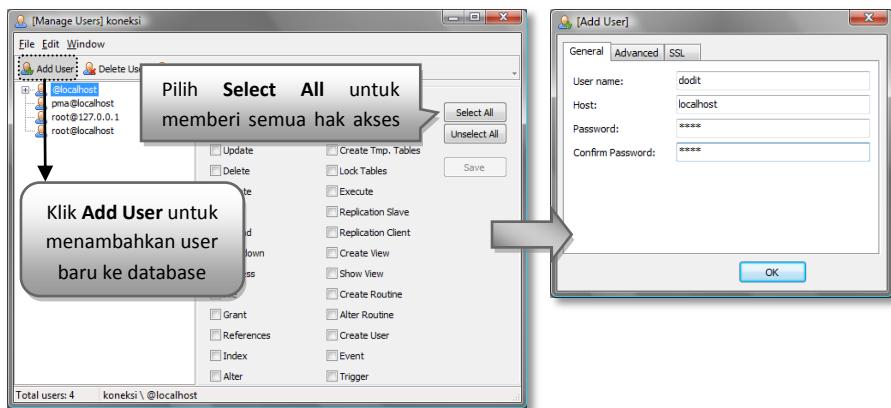


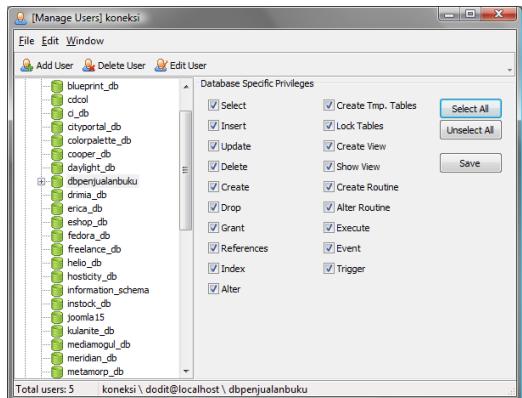
Item **Connection**, untuk membentuk koneksi ke database MySQL. Item connection cukup dibuat satu kali, berikutnya saat navicat dibuka kembali maka secara otomatis koneksi sudah terbentuk.



Item **Manage Users**, berfungsi untuk menambahkan atau mengubah pengguna database, termasuk penambahan dan pengurangan hak akses atas database yang aktif saat ini (**dbpenjulanbuku**). Item **Manage Users** terdiri atas beberapa item antara lain **Add User**, **Delete User** dan **Edit User**.

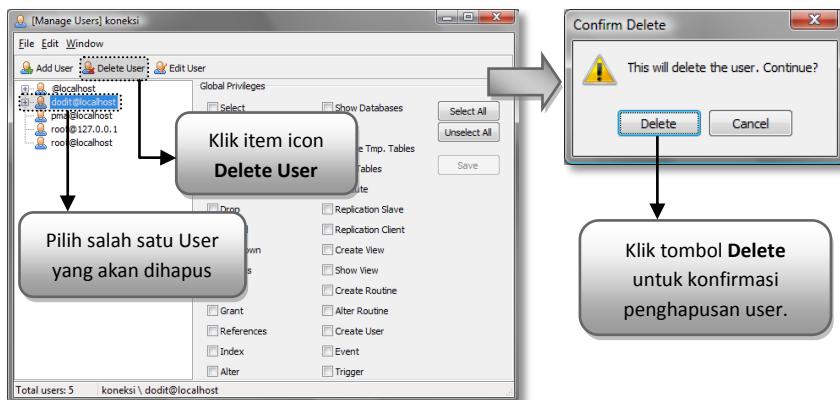
Untuk menambahkan user baru ke database, klik item **Add User**, kemudian isikan beberapa parameter yang ada, seperti berikut ini:



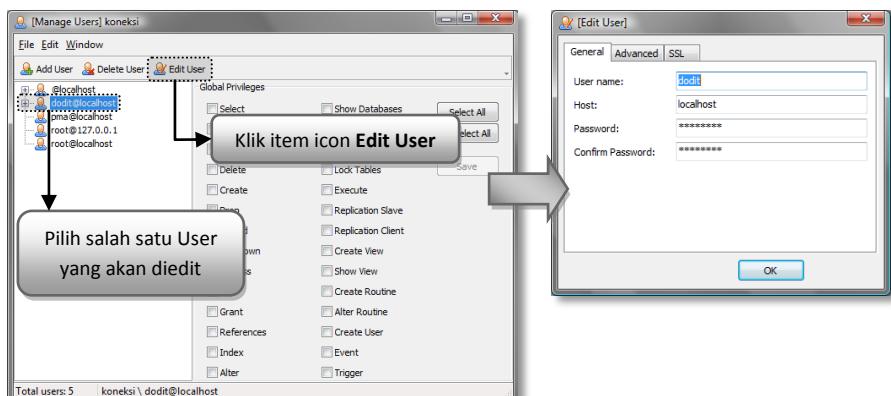


Selanjutnya adalah penentuan database mana user **dodit** akan diletakkan. Disini yang dipilih adalah database **dbpenjualanbuku** dan hak akses (*privileges*) penuh diberikan kepada user **dodit** sehingga tombol **Select All** harus diklik.

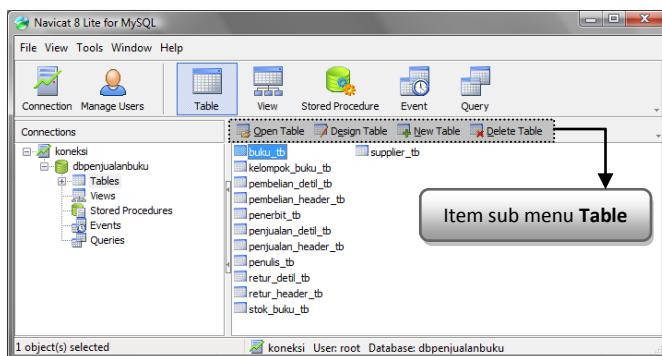
Untuk menghapus user dari database, pilih salah satu user yang akan dihapus kemudian klik item **Delete User** sampai bertemu dengan tampilan window yang meminta konfirmasi penghapusan user:



Untuk mengganti password user database, pilih salah satu user yang akan diganti passwordnya, kemudian klik item **Edit User**:



 Item **Table**, berfungsi untuk membuka table termasuk mengisikan baris-baris table, menciptakan table baru, mengubah table. Item **Table** memudahkan kita saat mengoperasikan table karena semuanya dapat dilakukan tanpa menggunakan script SQL sama sekali. Item **Table** terdiri atas **Open Table**, **Design Table**, **New Table** dan **Delete Table**.



Item **Open Table**, berfungsi jika Anda hendak memasukkan baris data ke dalam table atau sekedar mengetahui isi table. Untuk melakukannya, pilih salah satu table yang akan dilihat kemudian klik **Open Table**.

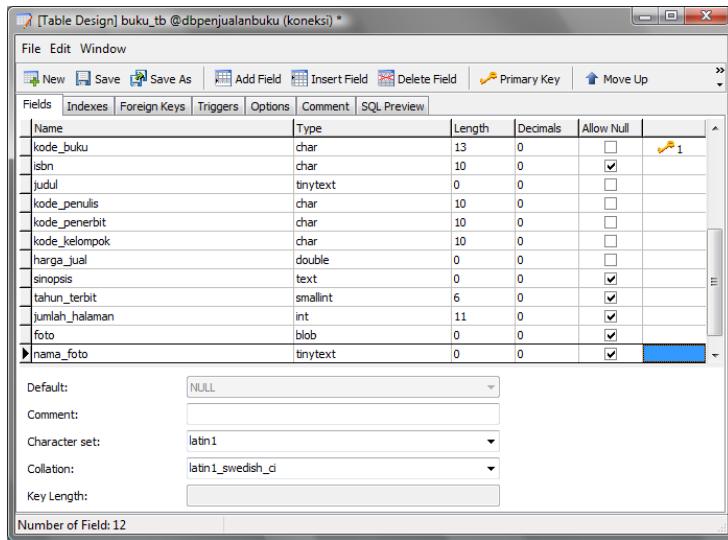


kode_buku	isbn	judul	kode_penulis	kode_penerbit	kode_kelompok	harga_jual	sinkron	tgl_update
9789791167123	9791167125	Buku Pintar Pemrograman PHP	0000000002	0000000002	0000000001	29500	(Me)	2012-07-10 10:45:20
9789792075830	9792075836	Aplikasi Manajemen Perkuliahan Berbasis Access 97/20	0000000005	0000000006	0000000001	35000	(Me)	2012-07-10 10:45:20
9789792405569	9792405569	Tips & Trik Baru Menembus Tes Bakat Skolastik,	0000000015	0000000005	0000000005	15000	(Me)	2012-07-10 10:45:20
9789793338125	979338121	Eserensi Bahasa Pemrograman JAVA Disertai Lab	0000000004	0000000003	0000000001	90000	(Me)	2012-07-10 10:45:20
9789793338729	979338729	Membuat Aplikasi Web Interaktif dengan ASP	0000000014	0000000003	0000000001	45000	(Me)	2012-07-10 10:45:20
9789793767079	9793767073	Efek Partikel 3ds max 6	0000000001	0000000001	0000000001	35000	(Me)	2012-07-10 10:45:20
9789793767543	9793767545	7 Jam Belajar Visual Basic .NET Untuk Orang Awam	0000000010	0000000001	0000000001	40000	(Me)	2012-07-10 10:45:20
9789793767666	9793767669	7 Jam Belajar Interaktif Autocad Untuk Orang Awam	0000000001	0000000001	0000000001	45000	(Me)	2012-07-10 10:45:20
9789795336730	9795336730	Internet Marketing	0000000013	0000000007	0000000006	25000	(Me)	2012-07-10 10:45:20
9789797633134	9797633136	Kumpulan Latihan Pemrograman Delphi						
9789797941307	9797941302	Joomla Cara Cepat & Mudah Membuat Website						
9789797941468	9797941469	Tanya Jawab Seputarn Joomla						

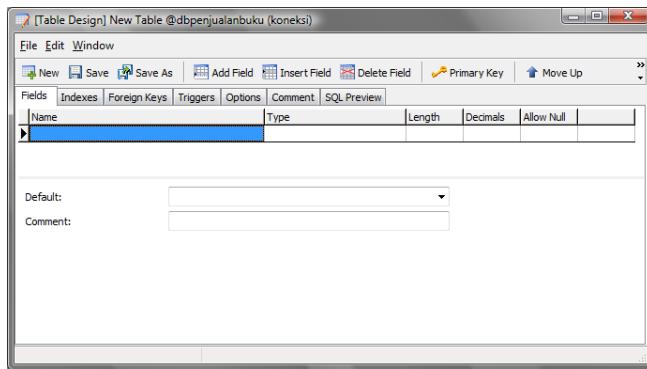
SELECT * FROM buku_tb LIMIT 0,1000

RECORD 2 OF 1211 RECORDS

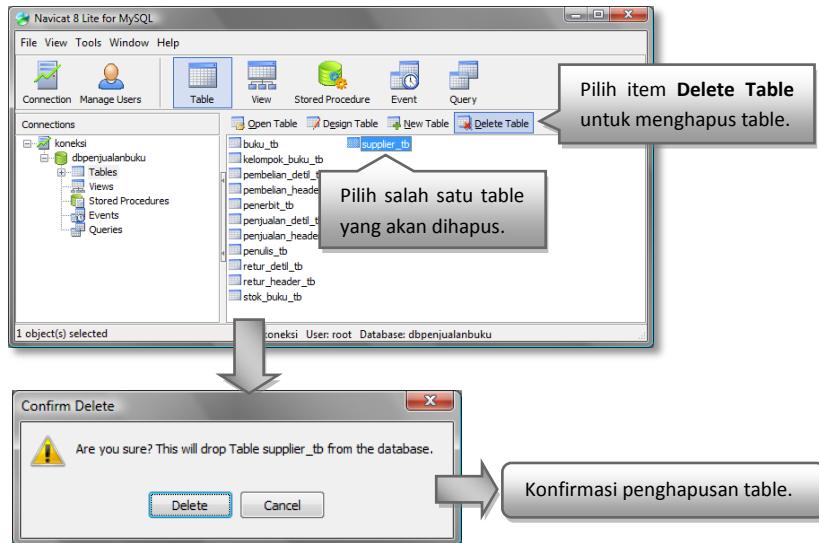
Item **Design Table** berfungsi untuk melihat serta mengubah struktur table. Perubahan struktur table bisa berupa penambahan atau pengurangan kolom table, mengubah tipe data kolom, menambahkan atau menghapus *primary key*, memindahkan urutan kolom dari atas ke bawah atau sebaliknya, menambahkan trigger (akan dibahas pada bab selanjutnya), dan lain sebagainya.



Item **New Table** berfungsi untuk menciptakan table baru secara *wizard* tanpa menggunakan pernyataan SQL **CREATE TABLE**, semua dibuat semudah mungkin dan memanjakan Anda.



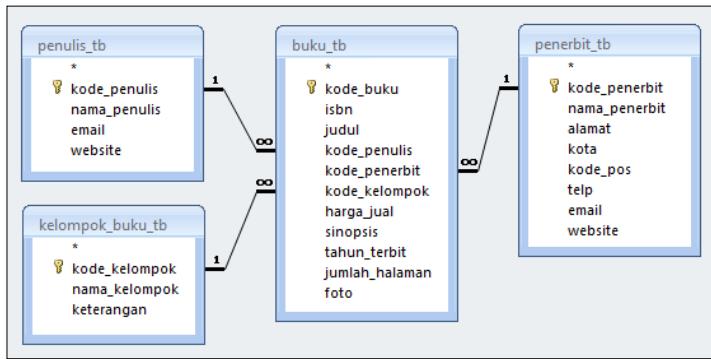
Item **Delete Table** berfungsi untuk menghapus table dari salah satu daftar table yang terpilih.



 Item **View**, View adalah semacam table maya. Dengan view Anda seolah-olah menciptakan sebuah table baru dimana kolom-kolom view diperoleh dari kolom-kolom yang berasal dari table atau view lainnya. Tentu saja Anda harus mempertimbangkan aturan relasi table (*integrity constraint*) saat menciptakan view. View akan menyederhanakan pernyataan SELECT yang melibatkan banyak table.

Di sisi frontend (Visual Basic .NET 2005) nantinya akan memanggil view melalui pernyataan SELECT layaknya memanggil table sehingga tidak dibutuhkan lagi pernyataan SELECT yang rumit.

Ambil contoh, suatu *integrity constraint* seperti berikut:



Kita akan menampilkan kolom **kode_buku**, **isbn**, **judul**, **harga_jual**, **tahun_terbit**, **jumlah_halaman** dari table **buku_tb**, kolom **nama_penulis** dari table **penulis_tb**, kolom **nama_kelompok** dari table **kelompok_buku_tb**, dan kolom **nama_penerbit** dari table **penerbit_tb**, dimana baris data yang ditampilkan urut dari kecil ke besar berdasarkan **tahun_terbit** antara tahun 2000 sampai 2008, serta diurutkan berdasarkan **judul** dari kecil ke besar. Sehingga perintah SQL SELECT untuk menampilkan kolom-kolom di atas akan menjadi:

```

SELECT buku_tb.kode_buku, buku_tb.isbn, buku_tb.judul,
kelompok_buku_tb.nama_kelompok, penulis_tb.nama_penulis,
penerbit_tb.nama_penerbit, buku_tb.harga_jual, buku_tb.tahun_terbit,
buku_tb.jumlah_halaman

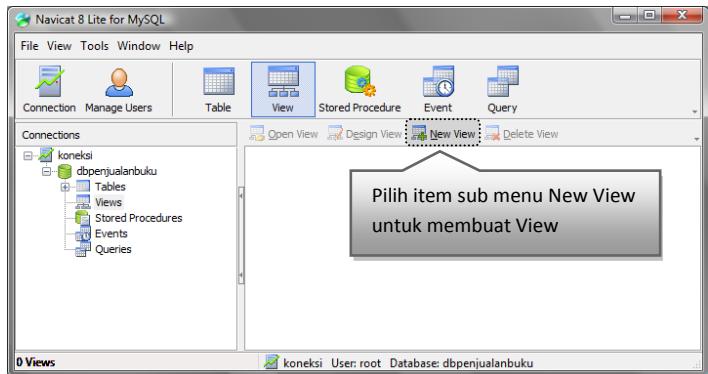
FROM penerbit_tb INNER JOIN (penulis_tb INNER JOIN (kelompok_buku_tb
INNER JOIN buku_tb ON kelompok_buku_tb.kode_kelompok =
buku_tb.kode_kelompok) ON penulis_tb.kode_penulis =
buku_tb.kode_penulis) ON penerbit_tb.kode_penerbit =
buku_tb.kode_penerbit

WHERE buku_tb.tahun_terbit BETWEEN 2000 AND 2008

ORDER BY buku_tb.judul;

```

Untuk menciptakan view dengan navicat sangat mudah, buka item **View->New View** sampai keluar window baru.



Tuliskan script SELECT di atas ke editor tab **Definition** :

```

1 SELECT buku_tb.kode_buku, buku_tb.bn, buku_tb.judul, kelompok_buku_tb.nama_kelompok,
2 penulis_tb.nama_penulis, penerbit_tb.nama_penerbit, buku_tb.harga_jual, buku_tb.sinopsis,
3 buku_tb.tahun_terbit, buku_tb.jumlah_halaman
4
5 FROM penerbit_tb INNER JOIN (penulis_tb INNER JOIN (kelompok_buku_tb INNER JOIN buku_tb ON
6 kelompok_buku_tb.kode_kelompok = buku_tb.kode_kelompok) ON penulis_tb.kode_penulis =
7 buku_tb.kode_penulis) ON penerbit_tb.kode_penerbit = buku_tb.kode_penerbit
8
9 WHERE buku_tb.tahun_terbit BETWEEN 2000 AND 2008
10
11 ORDER BY buku_tb.judul;
12

```

Tip & Trik:

Untuk men-generate perintah select yang rumit karena melibatkan banyak table atau view maka penulis menggunakan tool Query dari Microsoft Access. Di dalamnya terdapat fitur *integrity constraint* dan query generator sehingga kita tidak dituntut ahli dalam penulisan perintah SELECT. Anda tinggal membuat duplikasi table di Access dengan syarat struktur table sama dengan di MySQL, setelah men-generate SELECT, Anda tinggal meng-copy hasil script SQL-nya ke Navicat.

Perlu diketahui bahwa khusus Navicat versi berbayar fitur generate Query yaitu **SELECT Designer** telah disediakan sehingga mirip dengan fitur yang ada di MS Access atau MS SQLServer.

Jika item **Preview** diklik maka Anda akan menuju ke data grid yang menampilkan baris (*record*) data hasil pernyataan SELECT melalui view yang baru saja dibuat.

The screenshot shows the MySQL Workbench interface with a preview window titled "View Design New View @dbpenjualanbuku (koneksi)". The window contains a table with 13 rows of book data. Below the table, the SQL query is displayed: "SELECT buku_tb.kode_buku, buku_tb.isbn, buku_tb.judul, kelompok_buku_tb.nama_kelompok, penulis_tb.nam...". The status bar at the bottom right indicates "Record 1 of 13".

kode_buku	isbn	judul	nama_kelompok	nama_penulis	nama_penerbit	harga_jual	tahun_terbit	jumlah_halaman
9789799992260	9799992265	30 Menit Jadi Webmaster	Komputer	Haris Supriyansyah	Oase Media	29500	2007	127
9789793767666	9793767669	7 Jam Belajar Interaktif Autocad Untuk Or. Komputer	Komputer	Handi Chandra	Maxikom CV.	45000	2006	181
9789793767543	9793767545	7 Jam Belajar Visual Basic .NET Untuk Or. Komputer	Komputer	Firdaus	Maxikom CV.	40000	2006	191
9789792075830	9792075836	Aplikasi Manajemen Perekuturan Berbasis A	Komputer	Hengky W Pramana	Elex Media Komputindo	35000	2005	352
9789791167123	9791167125	Buku Pintar Pemrograman PHP	Komputer	Dodit Suprianto	Oase Media	29500	2008	340
9789793767079	9793767073	Efek Partikel 3ds max 6	Komputer	Handi Chandra	Maxikom CV.	35000	2004	175
9789793338125	9793338121	Esenensi Bahasa Pemrograman JAVA	Komputer	Bambang Hariyanta	Informatika Bandung	90000	2005	832
9789795336730	7975336738	Internet Marketing	Marketing	Riyuke Ustdiyanto	Andi Offset	25000	2001	171
9789797941307	9797941302	Joomla Cara Cepat & Mudah Membuat We	Komputer	Erlina Oneto	Media Kita	45000	2008	161
9789797633134	9797633136	Kumpulan Latihan Pemrograman Delphi	Komputer	Jaja Jamaludin Malik	Andi Offset	25000	2006	116
9789793338729	9793338725	Membuat Aplikasi Web Interaktif Dengan A	Komputer	Bernard Renaldi Suteja	Informatika Bandung	45000	2006	280
9789797941468	9797941469	Tanpa Jawab Sepertu Joomla	Komputer	Bunafit Nugroho	Media Kita	45000	2008	194
9789792405569	9792405569	Tips & Trik Baru Menembus Tes Bakat Skol:	Psikologi	Tjahyanto Pudji Juwono	Pustaka Horizona	15000	2006	175

Jika view sudah benar, sekarang lakukan penyimpanan terhadap view tersebut dengan mengeklik item **Save**. Selanjutnya cantumkan nama view **contoh_vw** seperti berikut ini:

The screenshot shows the Navicat 8 Lite interface. A dialog box titled "View Name" is open, showing the entry "contoh_vw". An arrow points from this dialog to the main Navicat window, which displays a tree view of database objects under the connection "koneksi". The "Views" node has a child node "contoh_vw". A callout bubble points to this node with the text "Terbentuk view baru 'contoh_vw'".



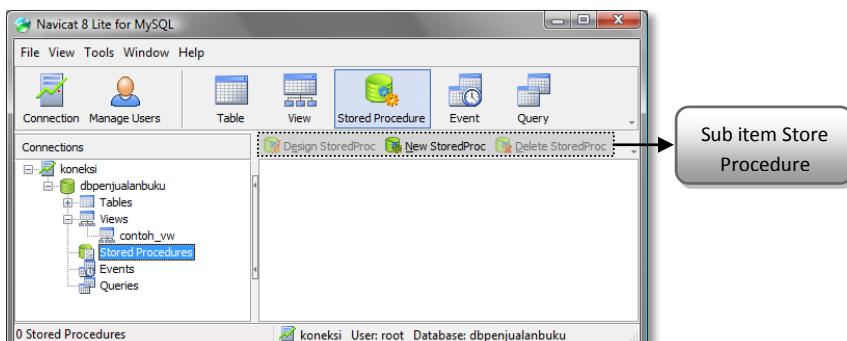
Item **Store Procedure** adalah potongan-potongan script SQL yang biasanya dilewati oleh beberapa parameter. Item Store Procedure Navicat terdiri atas item **Design StoredProcedure**, **New StoredProcedure** dan **Delete StoredProcedure**.

Pada dasarnya Store Procedure terbagi menjadi dua jenis:

- **Store Procedure**, jika potongan script SQL tidak menghasilkan nilai kembalian sehingga store procedure tidak dapat dioperasikan dalam suatu ekspresi.
- **Store Function**, jika kumpulan script SQL menghasilkan nilai kembalian sehingga store function dapat dioperasikan dalam suatu ekspresi.

Jika Anda sudah terbiasa dengan bahasa pemrograman komputer, maka store procedure serupa dengan routine/procedure pada bahasa pemrograman komputer umumnya.

Store Procedure terletak disisi database MySQL dan akan dipanggil dari sisi *frontend* (Visual Basic .NET 2005) atau dipanggil dari store procedure lainnya dalam database. Karena store procedure banyak dilibatkan pada program database maka kita akan membahas konsep store procedure lebih detil pada bab selanjutnya. Namun untuk saat ini kita hanya perlu tahu bagaimana menciptakan suatu store procedure melalui Navicat.



Sekarang mari kita coba bagaimana membuat suatu store procedure baru yang intinya adalah mengoperasikan pernyataan SQL INSERT, UPDATE dan DELETE pada table **penulis_tb**. Buka menu **Query->New Query** (agar lebih jelas bagaimana penggunaan fitur Query Navicat silahkan baca pada

bagian item **Query** di bawah) dan jalankan script penciptaan table **penulis_tb** berikut:

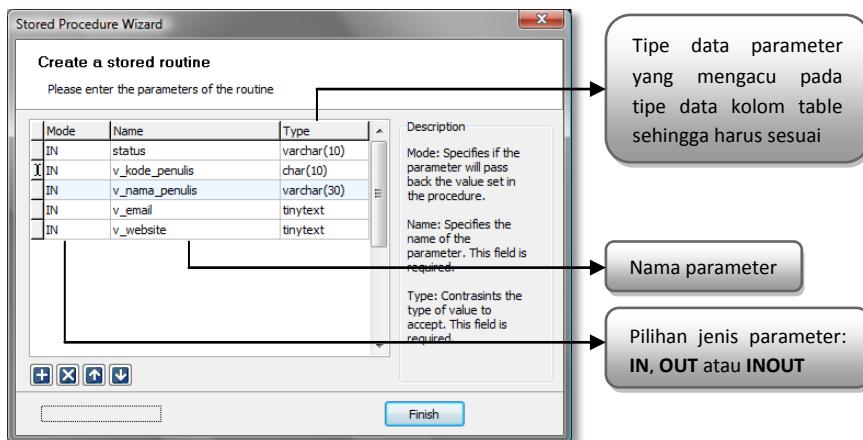
```
CREATE TABLE IF NOT EXISTS `penulis_tb` (
  `kode_penulis` char(10) NOT NULL DEFAULT '',
  `nama_penulis` varchar(30) NOT NULL,
  `email` tinytext,
  `website` tinytext,
  `tentang_penulis` text,
  PRIMARY KEY (`kode penulis`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

INSERT INTO `penulis_tb` (`kode_penulis`, `nama_penulis`, `email`,
`website`, `tentang_penulis`) VALUES
('0000000001', 'Handi Chandra', NULL, NULL, NULL),
('0000000002', 'Dodit Suprianto', 'd0dit@yahoo.com',
'doditsuprianto.com', NULL),
('0000000003', 'Bunafit Nugroho', NULL, NULL, NULL),
('0000000004', 'Bambang Hariyanto', NULL, NULL, NULL),
('0000000005', 'Hengky W Pramana', NULL, NULL, NULL),
('0000000006', 'Ahmad Sofyan', NULL, NULL, NULL),
('0000000007', 'Rini Agustina', NULL, NULL, NULL),
('0000000008', 'Tiffany Azhar Izzuddin', NULL, NULL, NULL),
('0000000009', 'Jaja Jamaludin Malik', NULL, NULL, NULL),
('0000000010', 'Firdaus', NULL, NULL, NULL),
('0000000011', 'Erima Oneto', NULL, NULL, NULL),
('0000000012', 'Haris Supriansyah', NULL, NULL, NULL),
('0000000013', 'Riyeket Ustadiyanto', NULL, NULL, NULL),
('0000000014', 'Bernard Renaldy Suteja', NULL, NULL, NULL),
('0000000015', 'Tjahjanto Pudji Juwono', NULL, NULL, NULL);
```

Klik item **New StoredProcedure**:



Masukkan **mode** parameter (IN, OUT atau INOUT), **name** (nama parameter) dan **type** (tipe data parameter yang mengacu pada tipe data kolom table), seperti tampak di bawah ini:



Selanjutnya klik tombol **Finish**, masukkan script SQL ini:

```
IF status='insert' THEN
    INSERT INTO penulis_tb (kode_penulis, nama_penulis, email,
website)
    VALUES (v_kode_penulis, v_nama_penulis, v_email, v_website);
ELSEIF status='update' THEN
```

```

    UPDATE penulis_tb SET nama_penulis=v_nama_penulis, email=v_email,
website=v_website
      WHERE kode_penulis=v_kode_penulis;
ELSE
    DELETE FROM penulis_tb WHERE kode_penulis=v_kode_penulis;
END IF;

```

Jika parameter **status** bernilai '**insert**' maka operasinya adalah INSERT, jika parameter **status** bernilai '**update**' maka operasinya adalah UPDATE dan jika parameter **status** bernilai '**delete**' maka operasinya adalah DELETE.

The screenshot shows a MySQL Workbench window titled '[Stored Procedure] New StoredProcedure @dbpenjualanbuku [koneksi] *'. The main area contains the following SQL code:

```

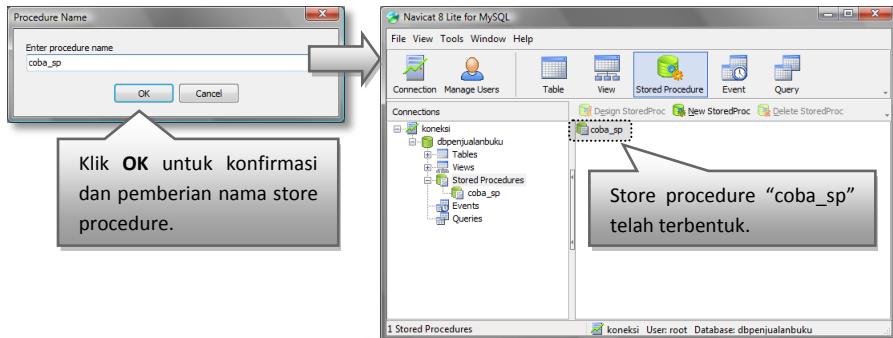
1 BEGIN
2   #Routine body goes here...
3   IF status='insert' THEN
4     INSERT INTO penulis_tb (kode_penulis, nama_penulis, email, website)
5       VALUES (v_kode_penulis, v_nama_penulis, v_email, v_website);
6
7   ELSEIF status='update' THEN
8     UPDATE penulis_tb SET nama_penulis=v_nama_penulis, email=v_email, website=v_website
9       WHERE kode_penulis=v_kode_penulis;
10
11   ELSE
12     DELETE FROM penulis_tb WHERE kode_penulis=v_kode_penulis;
13   END IF;
14
15 END;

```

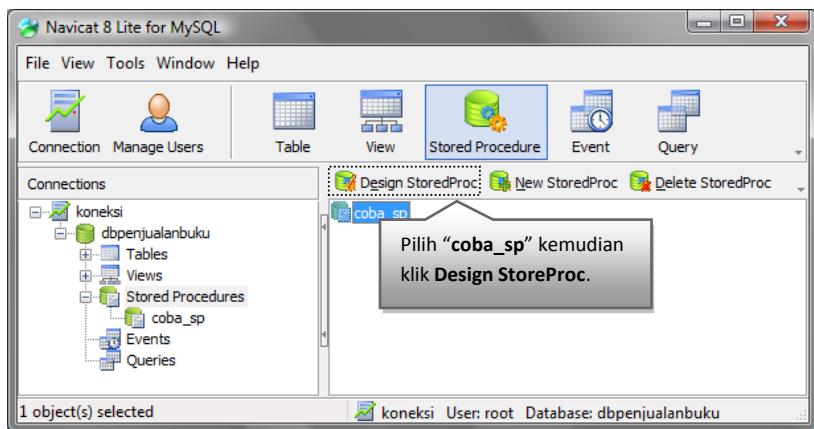
Below the code, there are three input fields:

- Parameter: IN `status` varchar(10),IN `v_kode_penulis` char(10),IN `v_nama_penulis` varchar();
- Return Type: (empty)
- Type: PROCEDURE

Selanjutnya klik item **Save** untuk menyimpan store procedure, beri nama store procedure dengan **coba_sp**.



Jika Anda hendak mengubah struktur store procedure **coba_sp** yang baru saja dibuat, klik satu kali pada **coba_sp**, kemudian pilih item **Design StoreProc**.



The screenshot shows the MySQL Workbench interface with a stored procedure named 'coba_sp'. The code is as follows:

```

1 BEGIN
2     #Routine body goes here...
3     IF status='Insert' THEN
4         INSERT INTO penulis_tb (kode_penulis, nama_penulis, email, website)
5             VALUES (v_kode_penulis, v_nama_penulis, v_email, v_website);
6     ELSEIF status='Update' THEN
7         UPDATE penulis_tb SET nama_penulis=v_nama_penulis, email=v_email, website=v_website
8             WHERE kode_penulis=v_kode_penulis;
9     ELSE
10        DELETE FROM penulis_tb WHERE kode_penulis=v_kode_penulis;
11    END IF;
12
13 END

```

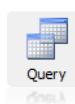
Parameter: IN `status` varchar(10),IN `v_kode_penulis` char(10),IN `v_nama_penulis` varchar();

Return Type:

Type: PROCEDURE

A callout bubble points to the code area with the text "Lakukan editing store procedure".

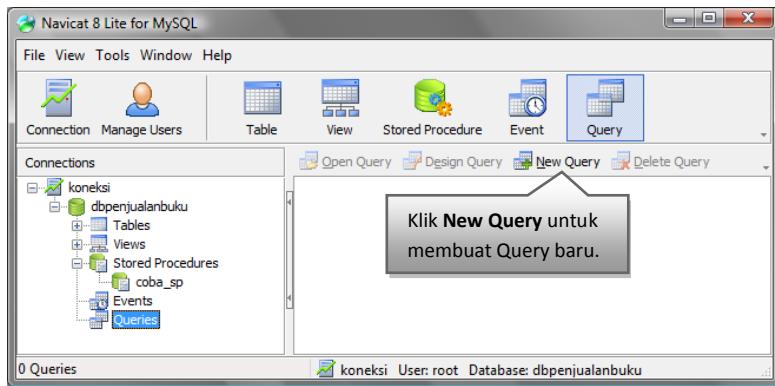
Untuk menghapus Store Procedure sangat mudah, pilih salah satu procedure yang akan dihapus, kemudian klik item **Delete StoreProc** seperti tampak gambar di bawah:



Item **Query**, berfungsi sebagai editor script SQL. Anda dapat menjalankan (mengeksekusi) script SQL tersebut dan langsung mengetahui hasilnya. Melalui fitur **Query** Anda dapat membuat berbagai macam pernyataan SQL seperti manipulasi table, view, store procedure, store function, dan sebagainya. Sebagai contoh, kita akan membuat perintah SQL yang akan memasukkan baris data ke table

penulis_tb dengan cara memanfaatkan store procedure **coba_sp** yang telah dibuat sebelumnya, kemudian menampilkan seluruh baris table **penulis_tb** dengan perintah **SELECT**.

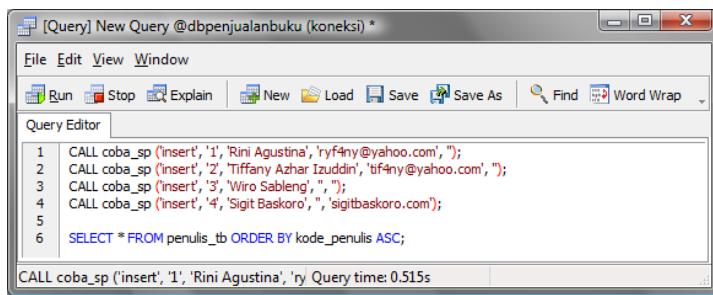
Buka menu **Query->New Query**, seperti tampak gambar di bawah ini:



Tuliskan editor Query dengan pernyataan-pernyataan SQL berikut ini:

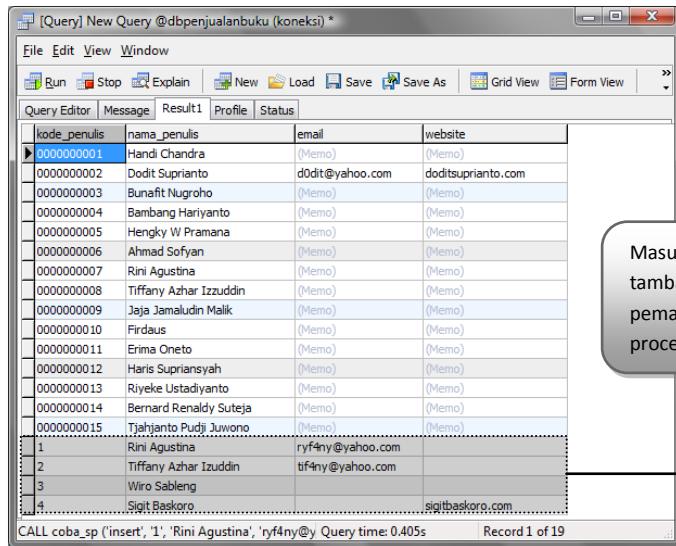
```
CALL coba_sp ('insert', '1', 'Rini Agustina', 'ryf4ny@yahoo.com',
 '');
CALL coba_sp ('insert', '2', 'Tiffany Azhar Izuddin',
 'tif4ny@yahoo.com', '');
CALL coba_sp ('insert', '3', 'Wiro Sableng', '', '');
CALL coba_sp ('insert', '4', 'Sigit Baskoro', '',
 'sigitbaskoro.com');

SELECT * FROM penulis_tb ORDER BY kode_penulis ASC;
```



Catatan: Call Nama_Store_Procedure (Param1, Param2, ParamN) adalah pernyataan SQL untuk memanggil Store Procedure.

Adapun cara untuk menjalankan serta mengetahui hasil dari pernyataan-pernyataan SQL tersebut di atas adalah dengan mengeklik item **Run**  . Untuk menyimpan script klik item **Save** .



1.4.4 Editor MySQL Console

Penulisan pernyataan-pernyataan SQL selain menggunakan Query (menu **Query->New Query**) bisa digunakan Editor **Console**. Editor Console dapat dibuka melalui menu **Tools->Console**, seperti tampak berikut ini:

The screenshot shows the MySQL Workbench interface. On the left, a toolbar with 'Tools', 'Window', and 'Help' tabs is visible. Below it are buttons for 'Console...', 'History Log...', 'Ctrl+H', and 'Options...'. A large arrow points from this toolbar area towards the main console window on the right. The main window has a title bar '[Console] koneksi'. The menu bar includes 'File', 'Edit', 'View', and 'Window'. Below the menu are standard file operations: Stop, Save, Load, Cut, Copy, and Clear. The SQL query entered is:

```
mysql> select kode_penulis, nama_penulis  
-> from penulis_tb;
```

The results are displayed in a table:

kode_penulis	nama_penulis
0000000001	Handi Chandra
0000000002	Dodit Suprianto
0000000003	Bunafit Nugroho
0000000004	Bambang Hariyanto
0000000005	Hengky W Pramana
0000000006	Ahmad Sofyan
0000000007	Rini Agustina
0000000008	Tiffany Azhar Izzuddin
0000000009	Jaja Jamaludin Malik
0000000010	Firdaus
0000000011	Erima Oneto
0000000012	Haris Supriansyah
0000000013	Riyuke Ustadiyanto
0000000014	Bernard Renaldy Suteja
0000000015	Tjahjanto Fudji Juwono

Below the table, the message '15 rows in set' is shown, followed by the MySQL prompt 'mysql>'.

BAB 2

Database Program Demo

Contoh demo project **PenjualanBuku** yang ada dalam CD penyerta buku sebagai bonus, datanya mengacu pada database **dbpenjualanbuku**, sehingga perlu dijelaskan lebih detail tentang database tersebut agar nantinya lebih mudah dipahami.

2.1 Program “**PenjualanBuku**”

Kasus yang diambil adalah transaksi yang terjadi pada sebuah toko buku antara lain: transaksi pembelian buku dari supplier, penjualan buku ke konsumen (konsumen boleh siapa saja, konsumen tidak harus menjadi anggota terlebih dahulu untuk membeli buku, sehingga konsumen tidak perlu didata terlebih dahulu ke dalam database), pengembalian/retur buku dari toko ke supplier jika terjadi kerusakan buku, tetapi retur tidak berlaku jika pengembalian buku berasal dari konsumen/pembeli ke toko buku. Dalam toko buku tersebut terdapat berbagai jenis buku yang dikelompokkan berdasarkan beberapa kategori, antara lain kelompok komputer, novel, agama, ekonomi dan lain-lain.

Database **dbpenjualanbuku** terdiri dari tiga belas table, antara lain:

- **Buku_tb** - berhubungan dengan data setiap buku, antara lain: judul buku, kode kelompok buku, kode penulis, kode penerbit, jumlah halaman, sinopsis, foto buku. Struktur table **buku_tb** adalah sebagai berikut:

Table buku_tb				
Nama Kolom	Tipe Data	Lebar	Null	Keterangan
kode_buku	CHAR	13	Tidak	Primary Key bersifat unik
isbn	CHAR	10	Ya	
judul	TINYTEXT		Tidak	Judul buku
kode_penulis	CHAR	10	Tidak	Foreign Key terhadap kolom kode_penulis dari table penulis_tb
kode_penerbit	CHAR	10	Tidak	Foreign Key terhadap kolom kode_penerbit dari table penerbit_tb
kode_kelompok	CHAR	10	Tidak	Foreign Key terhadap kolom kode_kelompok dari table kelompok_buku_tb
harga_jual	DOUBLE		Tidak	Harga jual standar buku. Harga bisa berubah saat transaksi penjualan
sinopsis	TEXT		Ya	Keterangan singkat tentang materi buku
tahun_terbit	SMALLINT	6	Ya	Tahun terbit buku
jumlah_halaman	INT	11	Ya	Jumlah halaman buku
foto	BLOB		Ya	File foto buku.

Pembuatan table **buku_tb** dapat dilakukan dengan dua cara (hal ini berlaku untuk semua pembuatan table lainnya). Pertama melalui cara *wizard* seperti yang sudah dijelaskan pada bab **1.4.3 Mengenal Toolbar Navicat**, yaitu dengan membuka menu **Table->New Table**, kemudian isikan nama, tipe data dan lebar kolom sesuai struktur table di atas.

[Table Design] buku_tb @dbpenjualanbuku (koneksi) *

Fields							Indexes	Foreign Keys	Triggers	Options	Comment	SQL Preview
Name	Type	Length	Decimals	Allow Null								
kode_buku	char	13	0	<input type="checkbox"/>								1
isbn	char	10	0	<input checked="" type="checkbox"/>								
judul	tinytext	0	0	<input type="checkbox"/>								
kode_penulis	char	10	0	<input type="checkbox"/>								
kode_penerbit	char	10	0	<input type="checkbox"/>								
kode_kelompok	char	10	0	<input type="checkbox"/>								
harga_jual	double	0	0	<input type="checkbox"/>								
sinopsis	text	0	0	<input checked="" type="checkbox"/>								
tahun_terbit	smallint	6	0	<input checked="" type="checkbox"/>								
jumlah_halaman	int	11	0	<input checked="" type="checkbox"/>								
foto	blob	0	0	<input checked="" type="checkbox"/>								

Default:
 Comment:
 Key Length:

Number of Field: 11

Cara kedua adalah melalui **Query**, buka menu **Query->New Query** kemudian tuliskan script **CREATE TABLE buku_tb** seperti ini:

```

DROP TABLE IF EXISTS `buku_tb`;
CREATE TABLE `buku_tb` (
  `kode_buku` char(13) NOT NULL DEFAULT '',
  `isbn` char(10) DEFAULT NULL,
  `judul` tinytext NOT NULL,
  `kode penulis` char(10) NOT NULL,
  `kode_penerbit` char(10) NOT NULL,
  `kode_kelompok` char(10) NOT NULL,
  `harga jual` double NOT NULL DEFAULT '0',
  `sinopsis` text,
  `tahun terbit` smallint(6) DEFAULT NULL,
  `jumlah_halaman` int(11) DEFAULT NULL,
  `foto` blob,
  `nama_foto` tinytext,
  PRIMARY KEY (`kode_buku`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

```

```

1 DROP TABLE IF EXISTS `buku_tb`;
2 CREATE TABLE `buku_tb` (
3   `kode_buku` char(13) NOT NULL DEFAULT '',
4   `isbn` char(10) DEFAULT NULL,
5   `judul` tinytext NOT NULL,
6   `kode_penulis` char(10) NOT NULL,
7   `kode_penerbit` char(10) NOT NULL,
8   `kode_kelompok` char(10) NOT NULL,
9   `harga_jual` double NOT NULL DEFAULT '0',
10  `sinopsis` text,
11  `tahun_terbit` smallint(6) DEFAULT NULL,
12  `jumlah_halaman` int(11) DEFAULT NULL,
13  `foto` blob,
14  `nama_foto` tinytext,
15  PRIMARY KEY (`kode_buku`)
16 ) ENGINE=InnoDB DEFAULT CHARSET=latin1;

```

Query time: 0.000s

- **Kelompok_buku_tb** – merupakan daftar pengelompokan buku, bertujuan untuk memudahkan kita saat melakukan proses pencarian buku berdasarkan kelompoknya. Struktur table **kelompok_buku_tb** adalah:

Table kelompok_buku_tb				
Nama Kolom	Tipe Data	Lebar	Null	Keterangan
kode_kelompok	CHAR	10	Tidak	Primary Key bersifat unik
nama_kelompok	VARCHAR	30	Tidak	Nama kelompok buku
keterangan	TINYTEXT		Ya	Keterangan terhadap kelompok buku (jika perlu)

Script **CREATE TABLE kelompok_buku_tb** adalah sebagai berikut:

```

DROP TABLE IF EXISTS `kelompok_buku_tb`;
CREATE TABLE `kelompok_buku_tb` (
  `kode_kelompok` char(10) NOT NULL DEFAULT '',

```

```

`nama_kelompok` varchar(30) NOT NULL,
`keterangan` tinytext,
PRIMARY KEY (`kode_kelompok`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

```

- **Penerbit_tb** – Merupakan daftar penerbit buku. Setiap buku pasti diterbitkan oleh penerbit dan setiap penerbit menerbitkan banyak buku. Struktur table **penerbit_tb** adalah:

Table penerbit_tb				
Nama Kolom	Tipe Data	Lebar	Null	Keterangan
kode_penerbit	CHAR	10	Tidak	Primary Key bersifat unik.
nama_penerbit	VARCHAR	30	Tidak	Nama penerbit
alamat	VARCHAR	50	Ya	Alamat penerbit
kota	VARCHAR	25	Ya	Kota penerbit
kode_pos	VARCHAR	6	Ya	
telp	VARCHAR	15	Ya	
email	TINYTEXT		Ya	Alamat email penerbit
website	TINYTEXT		Ya	Alamat website penerbit

Script **CREATE TABLE penerbit_tb** adalah sebagai berikut:

```

DROP TABLE IF EXISTS `penerbit_tb`;
CREATE TABLE `penerbit_tb` (
  `kode_penerbit` char(10) NOT NULL DEFAULT '',
  `nama_penerbit` varchar(30) NOT NULL,
  `alamat` varchar(50) DEFAULT NULL,
  `kota` varchar(25) DEFAULT NULL,
  `kode_pos` varchar(6) DEFAULT NULL,
  `telp` varchar(15) DEFAULT NULL,
  `email` tinytext,
  `website` tinytext,
  PRIMARY KEY (`kode_penerbit`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

```

- **Penulis_tb** – table yang berisi daftar penulis buku. Struktur table **penulis_tb** adalah sebagai berikut:

Table penulis_tb				
Nama Kolom	Tipe Data	Lebar	Null	Keterangan
kode_penulis	CHAR	10	Tidak	Primary Key bersifat unik
nama_penulis	VARCHAR	30	Tidak	Nama penulis
Email	TINYTEXT		Ya	Alamat email penulis
Website	TINYTEXT		Ya	Alamat website penulis
tentang_penulis	TEXT		Ya	Keterangan tentang biodata penulis

Script **CREATE TABLE penulis_tb** adalah sebagai berikut:

```
DROP TABLE IF EXISTS `penulis_tb`;
CREATE TABLE `penulis_tb` (
  `kode_penulis` char(10) NOT NULL DEFAULT '',
  `nama_penulis` varchar(30) NOT NULL,
  `email` tinytext,
  `website` tinytext,
  `tentang_penulis` text,
  PRIMARY KEY (`kode_penulis`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

- **Supplier_tb** – table yang berisi daftar supplier yang menjual atau menyuplai buku ke toko buku. Struktur table **supplier_tb** adalah sebagai berikut:

Table supplier_tb				
Nama Kolom	Tipe Data	Lebar	Null	Keterangan
kode_supplier	CHAR	10	Tidak	Primary Key bersifat unik
nama_supplier	VARCHAR	50	Tidak	Nama supplier
Alamat	VARCHAR	50	Ya	Alamat supplier
Kota	VARCHAR	25	Ya	Kota supplier
kode_pos	VARCHAR	6	Ya	Kode pos supplier
Telp	VARCHAR	15	Ya	
Fax	VARCHAR	15	Ya	
Email	TINYTEXT		Ya	Alamat email supplier

Website	TINYTEXT		Ya	Alamat website supplier
---------	----------	--	----	-------------------------

Script **CREATE TABLE supplier_tb** adalah sebagai berikut:

```
DROP TABLE IF EXISTS `supplier_tb`;
CREATE TABLE `supplier tb` (
  `kode_supplier` char(10) NOT NULL DEFAULT '',
  `nama_supplier` varchar(50) NOT NULL,
  `alamat` varchar(50) DEFAULT NULL,
  `kota` varchar(25) DEFAULT NULL,
  `kode_pos` varchar(6) DEFAULT NULL,
  `telp` varchar(15) DEFAULT NULL,
  `fax` varchar(15) DEFAULT NULL,
  `email` tinytext,
  `website` tinytext,
  PRIMARY KEY (`kode_supplier`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

- **Pembelian_header_tb** – table yang mencatat nota pembelian buku, tanggal pembelian, dan kode supplier. Table ini berhubungan secara *one-to-many* (satu ke banyak) dengan table **pembelian_detil_tb**. Struktur table **pembelian_header_tb** adalah sebagai berikut:

Table pembelian_header_tb				
Nama Kolom	Tipe Data	Lebar	Null	Keterangan
nota_beli	CHAR	10	Tidak	Primary Key, nomor nota beli bersifat unik
tgl_beli	DATE		Tidak	Tanggal nota pembelian
kode_supplier	CHAR	10	Tidak	Foreign Key terhadap kolom kode_supplier dari table supplier_tb

Script **CREATE TABLE pembelian_header_tb** adalah sebagai berikut:

```
DROP TABLE IF EXISTS `pembelian header tb`;
CREATE TABLE `pembelian header tb` (
  `nota_beli` char(10) NOT NULL DEFAULT '',
  `tgl_beli` date NOT NULL,
  `kode_supplier` char(10) NOT NULL,
  PRIMARY KEY (`nota_beli`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

- **Pembelian_detil_tb** – table yang mencatat detail pembelian buku pada setiap nota pembelian. Struktur table **pembelian_detil_tb** adalah sebagai berikut:

Table pembelian_detil_tb				
Nama Kolom	Tipe Data	Lebar	Null	Keterangan
nota_beli	CHAR	10	Tidak	Primary Key, juga menjadi Foreign Key terhadap kolom nota_beli dari table pembelian_header_tb
kode_buku	CHAR	13	Tidak	Primary Key, juga menjadi Foreign Key terhadap kolom kode_buku dari table buku_tb
Jumlah	TINYTEXT	4	Tidak	Jumlah buku yang dibeli
Harga	DOUBLE		Tidak	Harga satuan buku yang dibeli

Script **CREATE TABLE pembelian_detil_tb** adalah sebagai berikut:

```
DROP TABLE IF EXISTS `pembelian_detil_tb`;
CREATE TABLE `pembelian_detil_tb` (
  `nota_beli` char(10) NOT NULL DEFAULT '',
  `kode_buku` char(13) NOT NULL DEFAULT '',
  `jumlah` tinyint(4) unsigned NOT NULL DEFAULT '1',
  `harga` double unsigned NOT NULL,
  PRIMARY KEY (`nota_beli`,`kode_buku`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

- **Penjualan_header_tb** - table yang mencatat nota penjualan buku dan tanggal penjualan. Table ini berhubungan secara *one-to-many* (satu ke banyak) dengan table **penjualan_detil_tb**. Struktur table **penjualan_header_tb** adalah sebagai berikut:

Table penjualan_header_tb				
Nama Kolom	Tipe Data	Lebar	Null	Keterangan
nota_jual	CHAR	10	Tidak	Primary Key, nomor nota jual bersifat unik
tgl_jual	DATE		Tidak	Tanggal nota penjualan

Script **CREATE TABLE penjualan_header_tb** adalah sebagai berikut:

```
DROP TABLE IF EXISTS `penjualan_header_tb`;
CREATE TABLE `penjualan_header_tb` (
  `nota_jual` char(10) NOT NULL DEFAULT '',
  `tgl_jual` date NOT NULL,
  PRIMARY KEY (`nota_jual`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

- **Penjualan_detil_tb** – table yang mencatat detail penjualan buku pada setiap nota penjualan. Struktur table **penjualan_detil_tb** adalah:

Table penjualan_detil_tb				
Nama Kolom	Tipe Data	Lebar	Null	Keterangan
nota_jual	CHAR	10	Tidak	Primary Key, juga menjadi Foreign Key terhadap kolom nota_jual dari table penjualan_header_tb
kode_buku	CHAR	13	Tidak	Primary Key, juga menjadi Foreign Key terhadap kolom kode_buku dari table buku_tb
jumlah	SMALLINT	11	Tidak	Jumlah buku yang dijual
harga	DOUBLE		Tidak	Harga satuan buku yang dijual

Script **CREATE TABLE penjualan_detil_tb** adalah sebagai berikut:

```

DROP TABLE IF EXISTS `penjualan_detil_tb`;
CREATE TABLE `penjualan_detil_tb` (
  `nota_jual` char(10) NOT NULL DEFAULT '',
  `kode_buku` char(13) NOT NULL DEFAULT '',
  `jumlah` smallint(11) unsigned NOT NULL DEFAULT '1',
  `harga` double unsigned NOT NULL,
  PRIMARY KEY (`nota_jual`,`kode_buku`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

```

- **Retur_header_tb** - table yang mencatat pengembalian buku ke supplier. Table ini berhubungan secara *one-to-many* (satu ke banyak) dengan table **retur_detil_tb**. Struktur table **retur_header_tb** adalah sebagai berikut:

Table retur_header_tb				
Nama Kolom	Tipe Data	Lebar	Null	Keterangan
nota_retur	CHAR	10	Tidak	Primary Key, nomor nota retur bersifat unik
tgl_retur	DATE		Tidak	Tanggal nota retur
kode_supplier	CHAR	10	Tidak	Foreign Key terhadap kolom kode_supplier dari table supplier_tb

Script **CREATE TABLE retur_header_tb** adalah sebagai berikut:

```

DROP TABLE IF EXISTS `retur_header_tb`;
CREATE TABLE `retur_header_tb` (
  `nota_retur` char(10) NOT NULL,
  `tgl_retur` date NOT NULL,
  `kode_supplier` char(10) NOT NULL,
  PRIMARY KEY (`nota_retur`)
) ENGINE=MyISAM DEFAULT CHARSET=latin1;

```

- **Retur_detil_tb** - table yang mencatat detail pengembalian buku pada setiap nota retur. Struktur table **retur_detil_tb** adalah sebagai berikut:

Table retur_detil_tb				
Nama Kolom	Tipe Data	Lebar	Null	Keterangan
nota_retur	CHAR	10	Tidak	Primary Key, juga menjadi Foreign Key terhadap kolom nota_retur dari table retur_header_tb
kode_buku	CHAR	13	Tidak	Primary Key, juga menjadi Foreign Key terhadap kolom kode_buku dari table buku_tb
jumlah	SMALLINT	11	Tidak	Jumlah buku yang dijual
harga	DOUBLE		Tidak	Harga satuan buku yang dijual

Script **CREATE TABLE retur_detil_tb** adalah sebagai berikut:

```
DROP TABLE IF EXISTS `retur_detil_tb`;
CREATE TABLE `retur_detil_tb` (
  `nota_retur` char(10) NOT NULL,
  `kode_buku` char(13) NOT NULL,
  `jumlah` tinyint(4) NOT NULL DEFAULT '0',
  `harga` double NOT NULL DEFAULT '0',
  PRIMARY KEY (`nota_retur`,`kode_buku`)
) ENGINE=MyISAM DEFAULT CHARSET=latin1;
```

- **stok_buku_tb** – merupakan table yang menampung akumulasi stok buku di toko. Nilai stok buku diperoleh dari akumulasi semua transaksi yang pernah terjadi dalam satu bulan, antara lain transaksi pembelian, penjualan, dan retur. Nilai-nilai stok dalam table **stok_buku_tb** diperoleh secara otomatis atau *di-generate* oleh program dengan rumusan **Stok_Ahir = Stok_Awal + Stok_Beli – Stok_Retur – Stok_Jual** dan nilai stok dalam rupiah diperoleh dari perkalian rata-rata harga beli

dikalikan dengan masing-masing kolom stok buku. Struktur table **stok_buku_tb** adalah sebagai berikut:

Table stok_buku_tb				
Nama Kolom	Tipe Data	Lebar	Null	Keterangan
kode_buku	CHAR	13	Tidak	Primary Key, juga sebagai Foreign key terhadap kolom kode_buku dari table buku_tb
tahun	INT	4	Tidak	Primary Key, tahun stok
bulan	TINYINT	2	Tidak	Primary Key, bulan stok
stok_awal	INT	10	Tidak	Stok awal
stok_beli	INT	10	Tidak	Stok beli
stok_retur	INT	10	Tidak	Stok retur
stok_akhir	INT	10	Tidak	Stok akhir

Script **CREATE TABLE stok_buku_tb** adalah sebagai berikut:

```
DROP TABLE IF EXISTS `stok_buku_tb`;
CREATE TABLE `stok_buku_tb` (
  `kode_buku` char(13) NOT NULL DEFAULT '',
  `tahun` int(4) NOT NULL DEFAULT '0',
  `bulan` tinyint(2) NOT NULL DEFAULT '0',
  `stok_awal` int(10) unsigned NOT NULL DEFAULT '0',
  `stok_beli` int(10) unsigned NOT NULL DEFAULT '0',
  `stok_jual` int(10) unsigned NOT NULL DEFAULT '0',
  `stok_retur` int(10) unsigned NOT NULL DEFAULT '0',
  `stok_akhir` int(10) unsigned NOT NULL DEFAULT '0',
  PRIMARY KEY (`kode_buku`, `tahun`, `bulan`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

2.2 Isi Table

Untuk memudahkan kita dalam memahami program, ada baiknya jika table-table yang telah terbentuk diisi terlebih dahulu dengan beberapa data. Sebagai contoh adalah table **kelompok_buku_tb**.

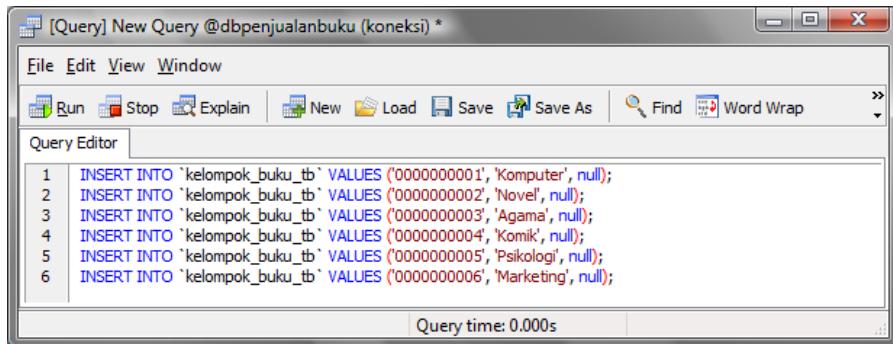
The screenshot shows a MySQL Workbench interface with the title bar '[Table] kelompok_buku_tb @dbpenjualanbuku (koneksi)'. Below the title bar is a menu bar with File, Edit, View, and Window. Under the View menu, there are options for Memo, Hex, and Image, along with Sort Ascending, Sort Descending, and Remove Sort buttons. The main area displays a table with three columns: kode_kelompok, nama_kelompok, and keterangan. The data consists of six rows:

kode_kelompok	nama_kelompok	keterangan
0000000001	Komputer	(Memo)
0000000002	Novel	(Memo)
0000000003	Agama	(Memo)
0000000004	Komik	(Memo)
0000000005	Psikologi	(Memo)
0000000006	Marketing	(Memo)

At the bottom of the interface, there is a SQL query: 'SELECT * FROM `kelompok_buku_tb` LIMIT 0, 1000' and a status message 'Record 6 of 6 in Page 1'.

Pemasukkan baris data table dapat dilakukan dengan dua cara, pertama melalui input secara langsung ke data grid seperti tampak gambar di atas (pilih salah satu table, kemudian klik ikon **Open Table**), kedua melalui penulisan script SQL INSERT, buka menu **Query->New Query** seperti berikut ini:

```
INSERT INTO `kelompok_buku_tb` VALUES ('0000000001', 'Komputer', null);
INSERT INTO `kelompok_buku_tb` VALUES ('0000000002', 'Novel', null);
INSERT INTO `kelompok_buku_tb` VALUES ('0000000003', 'Agama', null);
INSERT INTO `kelompok_buku_tb` VALUES ('0000000004', 'Komik', null);
INSERT INTO `kelompok_buku_tb` VALUES ('0000000005', 'Psikologi', null);
INSERT INTO `kelompok_buku_tb` VALUES ('0000000006', 'Marketing', null);
```



```
1 | INSERT INTO `kelompok_buku_tb` VALUES ('0000000001', 'Komputer', null);
2 | INSERT INTO `kelompok_buku_tb` VALUES ('0000000002', 'Novel', null);
3 | INSERT INTO `kelompok_buku_tb` VALUES ('0000000003', 'Agama', null);
4 | INSERT INTO `kelompok_buku_tb` VALUES ('0000000004', 'Komik', null);
5 | INSERT INTO `kelompok_buku_tb` VALUES ('0000000005', 'Psikologi', null);
6 | INSERT INTO `kelompok_buku_tb` VALUES ('0000000006', 'Marketing', null);
```

Query time: 0.000s

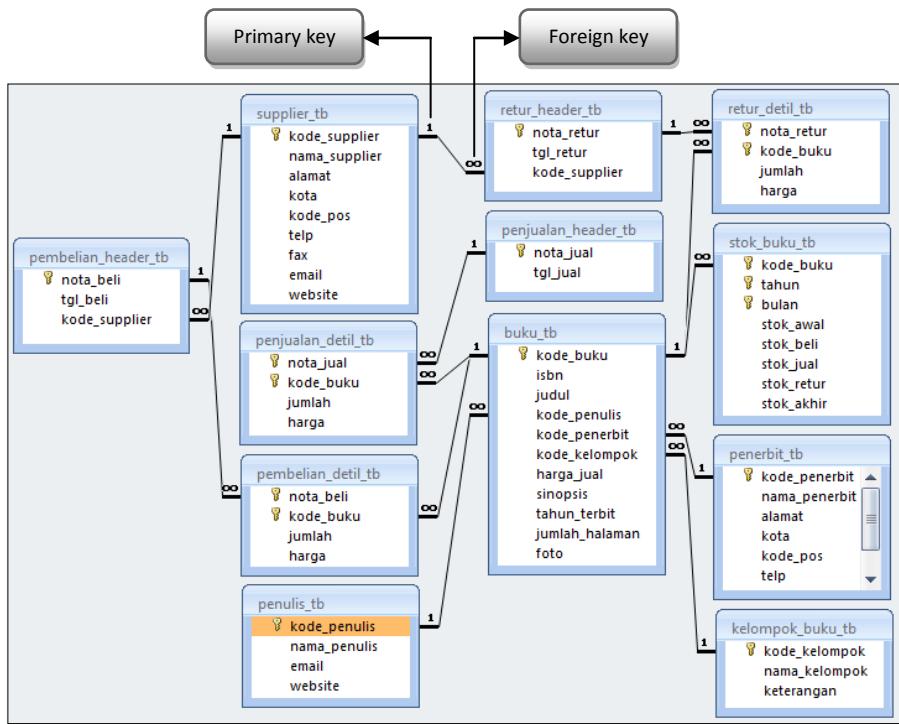
Pemasukkan baris data ke table seperti di atas berlaku bagi seluruh table yang ada.

2.3 Relasi Antar Table (Integrity Constraints)

Pengisian baris data dalam table harus memperhatikan batasan *integrity constraints* yaitu semacam aturan yang menjaga keutuhan data table dan menjadikan data benar adanya. Penerapan *integrity constraints* dapat berupa hubungan antar table (*table relation*), kolom kunci table yang bersifat unik (*primary key*), tipe data, lebar kolom, dan lain sebagainya.

Sebagai contoh, dua buku dengan judul berbeda tidak mungkin memiliki kode buku yang sama, satu nomor nota pembelian tidak mungkin memiliki dua atau lebih kode buku yang sama (yang bisa dilakukan adalah menambah jumlah buku yang dibeli pada kolom jumlah), tidak mungkin melakukan penjualan buku dengan **kode_buku** yang belum terdaftar pada table **buku_tb** dan masih banyak lagi.

Bagan di atas merupakan hubungan antar table-table secara konseptual di dalam database **dbpenjualanbuku**, dimana setiap table mempunyai paling tidak satu simbol kunci yang menunjukkan kolom *primary key*, dan hubungan antar table bersifat satu ke banyak (*one-to-many relation*).



Hubungan antara table satu dengan table lainnya didasarkan pada kolom *foreign key* dan kolom *primary key* pada setiap table. Kolom *primary key* ditandai oleh simbol kunci, terbentuk saat pembuatan table. Kolom *foreign key* adalah kolom yang terhubung dengan kolom *primary key* table lain.

Berikut ini adalah daftar kolom *primary key* dan *foreign key* seluruh table yang ada pada database **dbpenjualanbuku** beserta dua belas nama relasi yang akan dibuat:

No	Nama Relasi	Foreign Key	Primary Key
Table buku_tb			
1	Fk_buku_kelompok	buku_tb.kode_kelompok	kelompok_buku_tb.kode_kelompok

2	Fk_buku_penerbit	buku_tb.kode_penerbit	penerbit_tb.kode_penerbit
3	Fk_buku_penulis	buku_tb.kode_penulis	penulis_tb.kode_penulis

Table pembelian_detil_tb

4	Fk_pembeliand_buku	pembelian_detil_tb.kode_buku	buku_tb.kode_buku
5	Fk_pembelian_hd	pembelian_detil_tb.nota_beli	pembelian_header_tb.nota_beli

Table pembelian_header_tb

6	Fk_pembelian_supplier	pembelian_header_tb.kode_supplier	supplier_tb.kode_supplier
---	-----------------------	-----------------------------------	---------------------------

Table penjualan_detil_tb

7	Fk_penjualand_buku	penjualan_detil_tb.kode_buku	buku_tb.kode_buku
8	Fk_penjualanh_penjualand	penjualan_detil_tb.nota_jual	penjualan_header_tb.nota_jual

Table retur_detil_tb

9	Fk_returd_buku	retur_detil_tb.kode_buku	buku_tb.kode_buku
10	Fk_returd_returh	retur_detil_tb.kode_buku	retur_header_tb.kode_buku

Table retur_header_tb

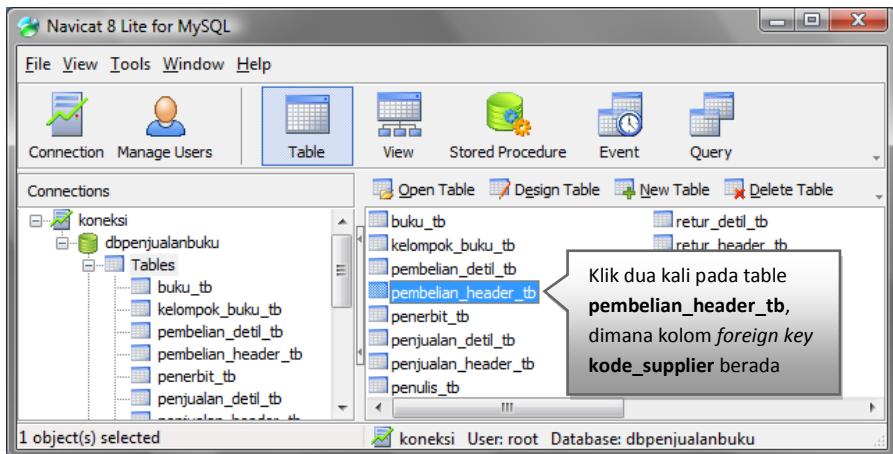
11	Fk_returh_suppplier	retur_header_tb.kode_supplier	supplier_tb.kode_supplier
----	---------------------	-------------------------------	---------------------------

Table stok_buku_tb

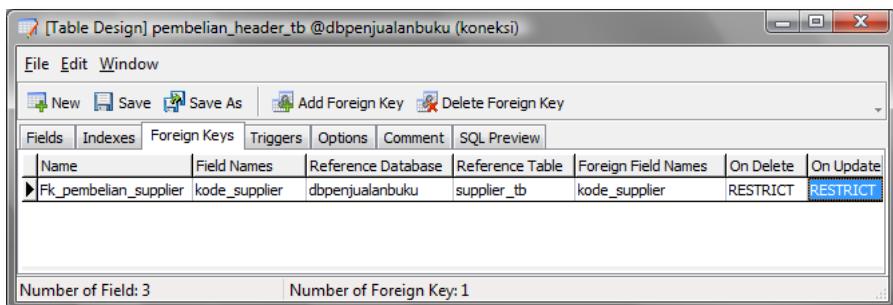
12	Fk_stok_buku	stok_buku_tb.kode_buku	buku_tb.kode_buku
----	--------------	------------------------	-------------------

Sebagai contoh kita akan menghubungkan kolom *foreign key* **pembelian_header_tb.kode_supplier** dengan kolom *primary key* **supplier_tb.kode_supplier** (lihat daftar table khusus nomor 6), caranya adalah sebagai berikut:

- Pilih kolom *foreign key* **kode_supplier** pada table **pembelian_header_tb**. Kemudian klik ikon **Design Table**.



- Pilih tab **Foreign Keys**, kemudian isikan parameternya sesuai dengan daftar table di atas.



Anda bisa membentuk relasi sisa yang belum terbentuk dengan cara yang sama.

Penulisan script SQL untuk menghubungkan antara table **pembelian_header_tb** dengan table **supplier_tb** pada kolom **kode_supplier** adalah sebagai berikut:

```
DROP TABLE IF EXISTS `pembelian_header_tb`;
CREATE TABLE `pembelian_header_tb` (
  `nota_beli` char(10) NOT NULL DEFAULT '',
  ...)
```

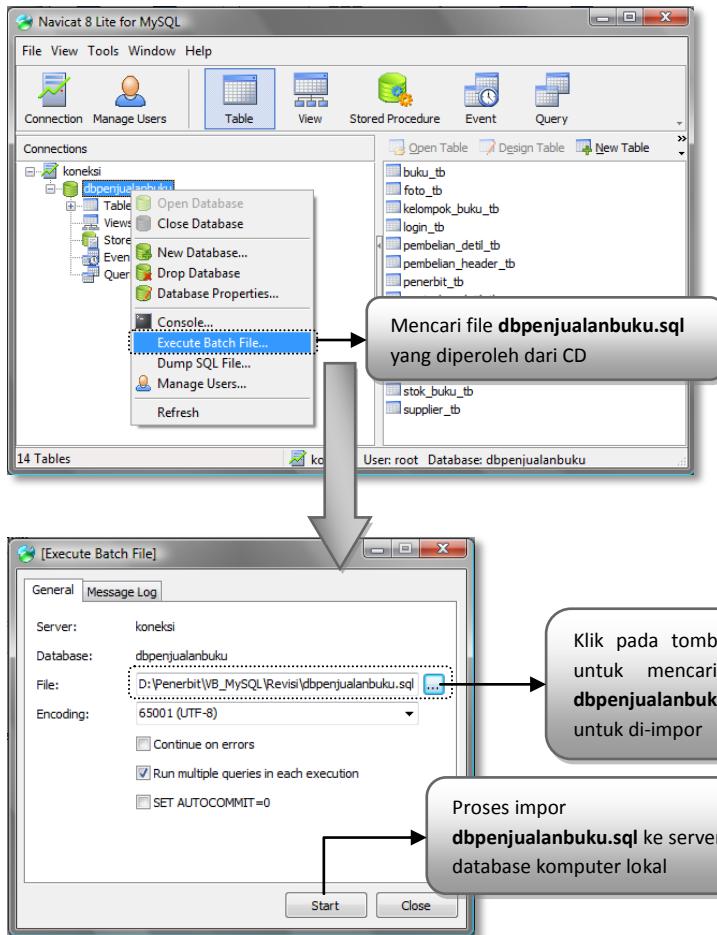
```
`tgl beli` date NOT NULL,  
`kode supplier` char(10) NOT NULL,  
PRIMARY KEY (`nota_beli`),  
KEY `Fk_pembelian_supplier` (`kode_supplier`),  
CONSTRAINT `Fk_pembelian_supplier` FOREIGN KEY (`kode_supplier`)  
REFERENCES `supplier_tb` (`kode_supplier`)  
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

Catatan: Relasi antar table sebagai salah satu fitur untuk menjaga keutuhan data (*integrity constraint*) hanya dapat dilakukan oleh table berjenis **InnoDB** dan tidak berlaku bagi table berjenis **MyIsam**.

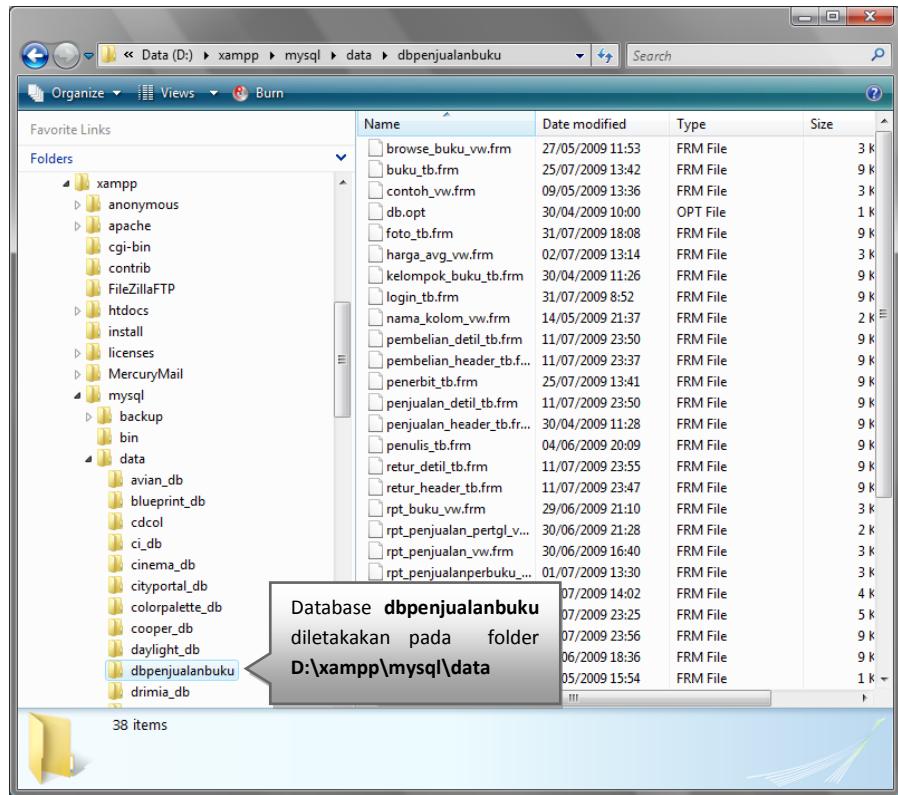
2.4 Impor SQL dbpenjualanbuku

Dalam CD bonus buku telah disertai script SQL, berisi struktur kode *table*, *view*, dan *store procedure* yang berkaitan dengan database **dbpenjualanbuku**. Ada dua cara untuk membentuk database **dbpenjualanbuku** ke server database komputer lokal Anda:

1. Melalui impor script SQL, caranya:
 - a. Buka program Navicat, bentuk koneksi baru (lihat BAB 1 tentang program Navicat).
 - b. Buat database baru bernama **dbpenjualanbuku** melalui perintah console. Buka menu **Tools->Console**, tuliskan pernyataan **CREATE DATABASE dbpenjualanbuku**, kemudian jalankan.
 - c. Klik kanan pada database **dbpenjualanbuku**, pilih menu **Execute Batch File**. Cari file **dbpenjualanbuku.sql** yang ada dalam CD penyerta buku, setelah itu klik tombol **start** untuk melakukan impor database.



2. Dengan menggandakan folder **dbpenjualanbuku** beserta seluruh file yang ada di dalamnya (diperoleh dari CD penyerta buku) ke folder dimana server database berada. Sebagai contoh, letakkan folder **dbpenjualanbuku** di folder **D:\xampp\mysql\data**, alamat drive hardisk tidak harus sama.



BAB 3

Store Program MySQL

Seorang programer database dituntut untuk memahami database dengan baik misalnya tentang perintah atau pernyataan SQL, konsep pemodelan table (*integrity constraint*), *aggregate* (pernyataan *max*, *min*, *count*, dan lain-lain), *filtering* (pernyataan *having*, *where*), *grouping* (pernyataan *group by*), function *built-in* dan fitur-fitur database MySQL lain yang bersifat mendasar. Apabila Anda belum pernah berkecimpung dengan MySQL, alangkah baiknya jika mempelajari MySQL lebih detail dari buku-buku lain yang khusus membahas tentang dasar-dasar MySQL.

Untuk saat ini kita mempelajari beberapa fitur MySQL sedikit ke tingkat lanjut antara lain **store program (store procedure, store function)** dan **view**. Meskipun masih banyak fitur-fitur MySQL lainnya yang tidak kalah penting untuk dipahami namun tidak berhubungan langsung dengan Visual Basic 2005, misalnya tentang **Trigger**, **Trasaction**, **Event**, dan **Locking** sehingga diharapkan pembaca terus meningkatkan kemampuannya di bidang database MySQL.

View, Store Procedure dan **Store Function** beroperasi pada sisi *backend*, ini berarti bahwa semua script SQL diletakkan pada sisi database sedangkan pada sisi *frontend* (Visual Basic .NET 2005, 2008) hanya berisi script pemanggil dari script SQL **store procedure**, **store function**, dan **view** yang telah dibuat dalam MySQL. Oleh karena itu tingkat pemrosesan data menjadi lebih cepat dan keamanannya pun lebih baik jika dibandingkan script SQL diletakkan pada sisi *frontend* (Visual Basic 2005/2008).

3.1 View

MySQL mendukung dua jenis table:

- **Table Sesungguhnya**, lebih dikenal sebagai **table dasar**.
- **Table Turunan**, lebih dikenal sebagai **view**. Table turunan adalah table yang diperoleh dari hasil turunan table dasar atau view lainnya.

Table dasar dibuat melalui pernyataan **CREATE TABLE** dengan syarat tidak ada kesamaan nama table dalam satu database. Contohnya adalah table **penulis_tb** dan table **penerbit_tb** dalam database **dbpenjualanbuku**.

Table turunan atau view secara fisik tidak menyimpan baris-baris data table. View hanya berfungsi sebagai pemberi layanan berupa rumusan yang mengkombinasikan data tertentu dari table dasar menjadi **table maya**. Dikatakan table maya karena baris-baris data dalam view terbentuk ketika view digunakan dalam pernyataan saja. Pada kondisi tersebut MySQL menelusuri rumusan view, menjalankan view, dan menyajikan baris-baris data ke user layaknya table dasar atau table sesungguhnya.

3.1.1 Membuat View

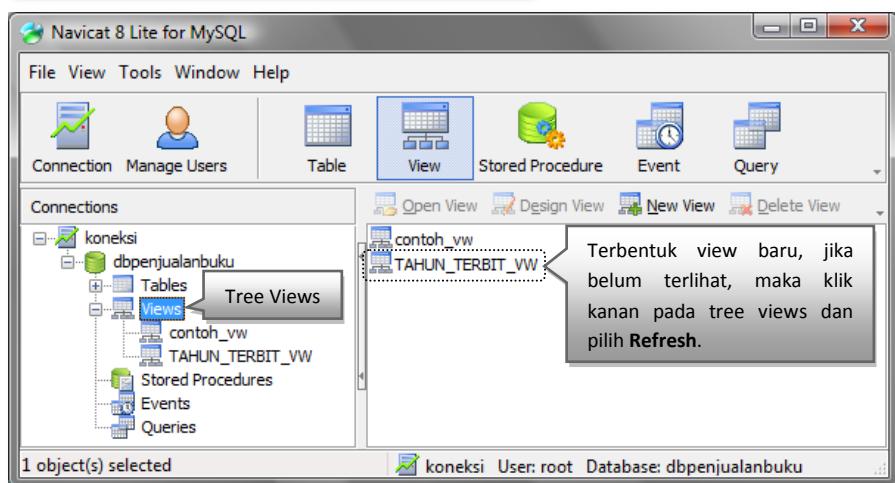
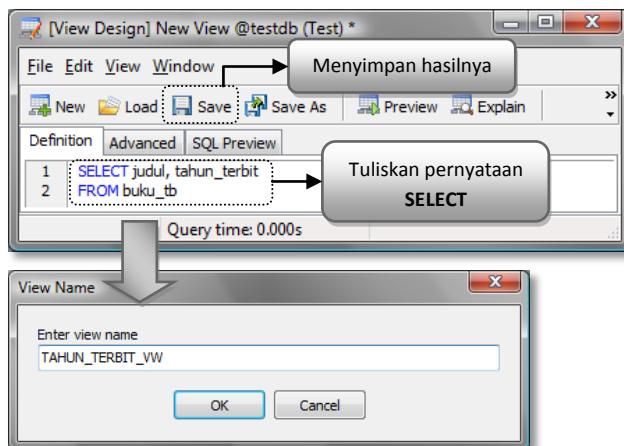
View dibuat melalui pernyataan CREATE VIEW

Definisi:

```
CREATE [ OR REPLACE ] VIEW <view name> [ <column list> ]
AS <table expression>
[ WITH [ CASCADED | LOCAL ] CHECK OPTION ]
```

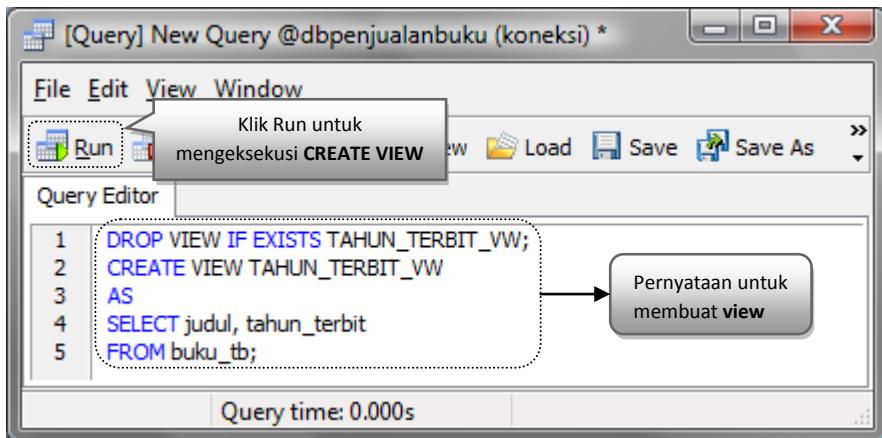
Penciptaan view dengan navicat dapat dilakukan dengan dua cara, pertama melalui item menu **View->New View** yang secara otomatis akan men-generate CREATE VIEW, cara kedua melalui item menu **Query->New Query**.

- Menu item **View->New View**, Anda cukup menuliskan pernyataan **SELECT** pada editor tab **Definition**, menyimpannya.



- Menu item **Query->New Query**, sebagai contoh kita akan menampilkan kolom judul dan tahun terbit dari table **buku_tb** dengan menggunakan view bernama **TAHUN_TERBIT_VW**. Script SQL yang harus dituliskan adalah sebagai berikut:

```
DROP VIEW IF EXISTS TAHUN_TERBIT_VW;  
CREATE VIEW TAHUN_TERBIT_VW  
AS  
SELECT judul, tahun_terbit  
FROM buku_tb;
```



Untuk memperoleh hasil view **TAHUN_TERBIT_VW** yang baru saja dibuat, klik item **Run**, untuk menyimpan script tersebut jika mungkin sewaktu-waktu mau diubah klik item **Save**.



3.1.2 Nama Kolom View

Nama kolom dalam view secara default akan mengikuti nama kolom yang tercantum dalam klausa SELECT. Sebagai contoh, terdapat empat kolom pada table **penulis_tb** antara lain: **kode_penulis**, **nama_penulis**, **email** dan **website**.

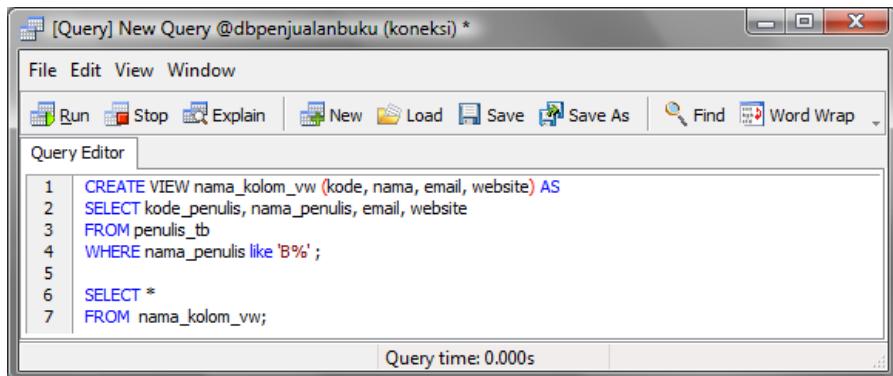
The screenshot shows the MySQL Workbench table editor for the 'penulis_tb' table. The table has four columns: 'kode_penulis', 'nama_penulis', 'email', and 'website'. The data rows are: 0000000001 (Handi Chandra, (Memo), (Memo)), 0000000002 (Dodit Suprianto, d0dit@yahoo.com, (Memo)), 0000000003 (Bunafit Nugroho, (Memo), (Memo)), 0000000004 (Bambang H, (Memo), (Memo)), 0000000005 (Hengky W P, (Memo), (Memo)), 0000000006 (Ahmad Sofy, (Memo), (Memo)), and 0000000007 (Rini Agustina, (Memo), (Memo)). A callout bubble points to the first row with the text 'Nama kolom asli bawaan table penulis_tb'.

kode_penulis	nama_penulis	email	website
0000000001	Handi Chandra	(Memo)	(Memo)
0000000002	Dodit Suprianto	d0dit@yahoo.com	doditsuprianto.com
0000000003	Bunafit Nugroho	(Memo)	(Memo)
0000000004	Bambang H	(Memo)	(Memo)
0000000005	Hengky W P	(Memo)	(Memo)
0000000006	Ahmad Sofy	(Memo)	(Memo)
0000000007	Rini Agustina	(Memo)	(Memo)

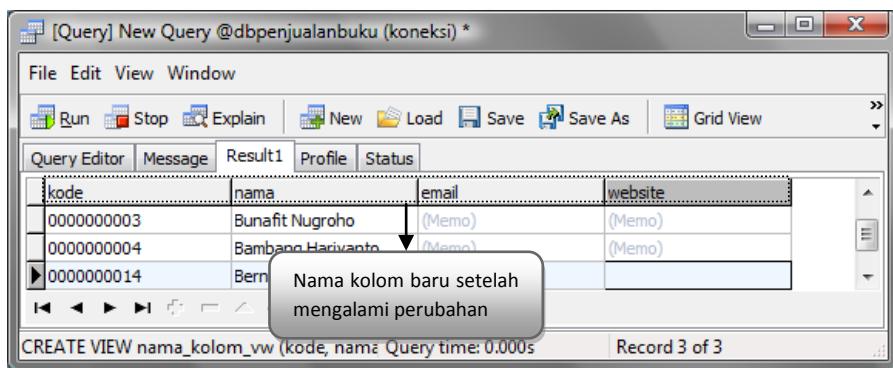
Kolom asli table dapat diubah dengan nama kolom baru yang Anda tentukan sendiri. Tuliskan script SQL di bawah ini pada editor query (menu **Query->New Query**):

```
CREATE VIEW nama_kolom_vw (kode, nama, email, website) AS
SELECT kode_penulis, nama_penulis, email, website
FROM penulis_tb
WHERE nama_penulis like 'B%' ;

SELECT *
FROM nama_kolom_vw;
```



Jika dijalankan dengan klik item **Run** hasilnya kurang lebih seperti ini:



3.1.3 WITH CHECK OPTION

Anda harus hati-hati dalam melakukan perubahan data melalui view karena bisa menghasilkan keluaran yang tidak diharapkan. Berikut ini view yang menampilkan baris data dari table **buku_tb** dimana kolom **harga_jual** kurang dari **40000**:

```
DROP VIEW IF EXISTS HARGA_JUAL_VW;
CREATE VIEW HARGA_JUAL_VW
AS
SELECT *
FROM buku_tb
WHERE harga_jual < 40000;
```

Jika dilaksanakan pernyataan **SELECT * FROM HARGA_JUAL_VW** maka hasilnya adalah:

kode_buku	isbn	judul	kode_penerbit	kode_kelempok	harga_jual	sinopsis	tahun_terbit	jumlah_halaman
9789791167123	9791167125	Buku Pintar Pemrograman PHP	0000000002	0000000001	29500	(Memo)	2008	340
9789792075830	9792075836	Aplikasi Manajemen Perekuratur	0000000005	0000000006	35000	(Memo)	2005	352
9789792405569	9792405569	Tips & Trik Baru Memembus Tes	0000000015	0000000005	15000	(Memo)	2006	175
9789793767079	9793767073	Erek Partikel 3ds max 6	0000000001	0000000001	35000	(Memo)	2004	175
9789795336730	9795336738	Internet Marketing	0000000013	0000000007	25000	(Memo)	2001	171
9789797633134	9797633136	Kumpulan Latihan Pemrograma	0000000009	0000000007	25000	(Memo)	2006	116
978999992260	9799992265	30 Menit Mendjadi Webmaster	0000000012	0000000002	29500	(Memo)	2007	127

Sekarang kita akan mengubah harga jual buku khusus **kode_buku = 9789792405569** yang semula **15000** menjadi **25000**.

```
UPDATE HARGA_JUAL_VW
SET harga_jual = 25000
WHERE kode_buku = 9789792405569
```

Pernyataan **UPDATE VIEW HARGA_JUAL_VW** di atas benar karena baris data dengan kode buku **9789792405569** tersedia dalam view

harga_jual_vw, namun akan bermasalah jika data tidak tersedia dan tidak ditemukan. Untuk mengatasi hasil keluaran yang tidak diharapkan maka sisipkan definisi tambahan yang disebut dengan **WITH CHECK OPTION**, sehingga definisi view berubah menjadi:

```
DROP VIEW IF EXISTS HARGA_JUAL_VW;
CREATE VIEW HARGA_JUAL_VW
AS
SELECT *
FROM buku_tb
WHERE harga_jual < 40000
WITH CHECK OPTION;
```

Jika dalam view terdapat klausula **WITH CHECK OPTION** maka semua perubahan pada view yang disebabkan oleh pernyataan **UPDATE**, **INSERT**, dan **DELETE** akan dicek terlebih dahulu kebenarannya. Karena view dapat dipanggil dari view lainnya maka pernyataan **WITH CHECK OPTION** akan membantu Anda untuk mengecek kebenaran data terhadap semua view yang terlibat. Jika view menspesifikasikan **WITH CASCDED CHECK OPTION** maka semua view yang berhubungan akan dicek kebenarannya. Jika **WITH LOCAL CHECK OPTION** digunakan maka pengecekan hanya dilakukan sekali pada view yang sedang diubah. **CASCDED** adalah pilihan default-nya.

```
DROP VIEW IF EXISTS HARGA_JUAL_VW;
CREATE VIEW HARGA_JUAL_VW
AS
SELECT *
FROM buku_tb
WHERE harga_jual < 40000
WITH CHECK OPTION;
```

Sama artinya dengan:

```
DROP VIEW IF EXISTS HARGA_JUAL_VW;
CREATE VIEW HARGA_JUAL_VW
AS
SELECT *
FROM buku_tb
WHERE harga_jual < 40000
WITH CASCDED CHECK OPTION;
```

3.1.4 Penghapusan View

Penghapusan view sangat mudah, contoh:

```
DROP VIEW harga_jual_vw;
```

Atau

```
DROP VIEW IF EXISTS harga_jual_vw;
```

Pernyataan **IF EXISTS** merupakan pernyataan tambahan yang memastikan bahwa view **harga_jual_vw** benar-benar ada sebelum dilakukan penghapusan view, bertujuan untuk mencegah keluarnya kesalahan saat pernyataan **DROP VIEW** dijalankan.

3.2 Store Procedure

Store procedure adalah beberapa potongan kode (procedure/routine) yang berisi deklarasi dan pernyataan-pernyataan SQL prosedural yang tersimpan di dalam katalog database dan dapat diaktifkan dengan cara memanggilnya dari program, trigger, atau store procedure lainnya.

Potongan kode store procedure bisa berupa deklarasi pernyataan SQL seperti **CREATE**, **UPDATE**, **SELECT** dan mungkin juga dilengkapi dengan pernyataan prosedural seperti **IF-THEN-ELSE** dan **WHILE-DO**.

Berikut ini contoh sederhana store procedure yang akan menghapus baris data penulis dari table **penulis_tb** berdasarkan kolom **kode_penulis**.

```
CREATE PROCEDURE hapus penulis sp (IN kode char(10))
BEGIN
    DELETE
    FROM penulis tb
    WHERE kode penulis = kode;
END
```

[Query] New Query @dbpenjualanbuku (koneksi) *

File Edit View Window

Klik Run untuk membentuk store procedure baru.

```

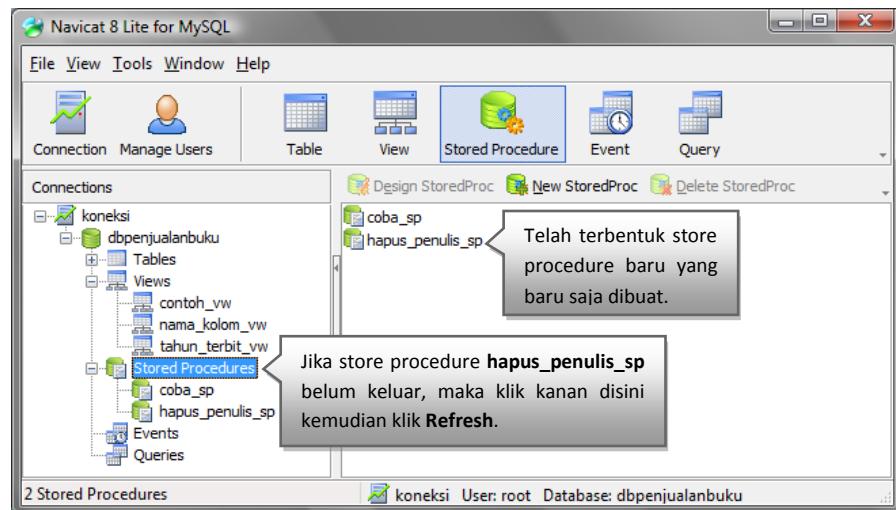
1 CREATE PROCEDURE hapus_penulis_sp(IN kode char(10))
2 BEGIN
3   DELETE
4   FROM penulis_tb
5   WHERE kode_penulis = kode;
6 END

```

parameter/argumen

Nama store procedure

Query time: 0.000s



Pemanggilan store procedure melalui pernyataan **CALL**, seperti berikut ini:

```
CALL hapus_penulis_sp ('0000000015');
```

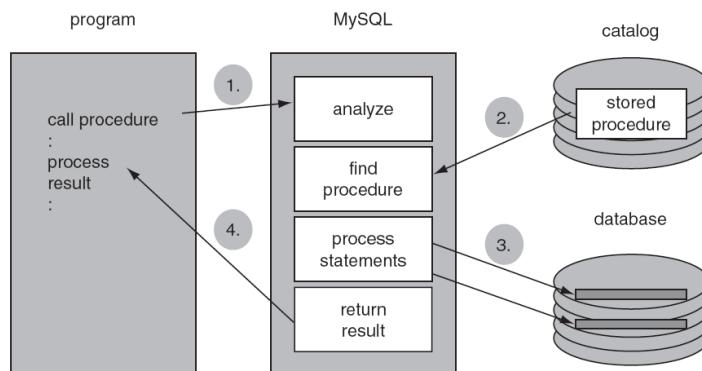
The screenshot shows the MySQL Workbench Query Editor window. The title bar reads "[Query] New Query @dbpenjualanbuku (koneksi) *". The menu bar includes File, Edit, View, Window. The toolbar has icons for Run, Stop, Explain, New, Load, Save, Save As, Find. The main area is labeled "Query Editor" and contains the following SQL code:

```
1 CALL hapus_penulis_sp ('0000000015');
```

Below the code, a status bar displays "Query time: 0.000s".

Catatan: penulisan store procedure dengan navicat dapat dilakukan dengan dua cara. Cara pertama adalah melalui item **Store Procedure->New StoreProc** seperti yang dijelaskan pada bab pertama. Cara kedua adalah melalui item **Query->New Query**.

Berikut ini bagan yang menggambarkan bagaimana tahapan proses store procedure:



Blok sebelah kiri mewakili program yang memanggil **procedure**, blok tengah mewakili server database, dan blok sebelah kanan mewakili database dan semua katalog. Proses berawal ketika procedure dipanggil dari program (langkah 1). Server database menerima panggilan ini dan mencari procedure yang sesuai dalam katalog (langkah 2). Setelah ketemu maka procedure dijalankan (langkah 3). Procedure selesai mengerjakan

tugasnya (bisa berupa penambahan baris baru, menghapus baris, dll), kemudian hasil procedure dikembalikan ke pemanggilnya (langkah 4). Di sini tidak ada komunikasi yang terjadi antara server database dengan program selama proses eksekusi procedure.

3.2.1 Variable

Variable adalah nama item data, nilai data yang ada dalam variable dapat berubah selama program dijalankan. Pendeklarasian variable adalah dengan pernyataan **DECLARE**, seperti berikut ini:

```
DECLARE variable_name [,variable_name...] datatype [DEFAULT value];
```

Pernyataan **DECLARE** dapat mewakili pendeklarasian banyak variable. Suatu variable bisa diberi nilai awal secara langsung melalui penambahan klausul **DEFAULT**, jika tidak disertai klausul **DEFAULT** maka variable akan bernilai awal **NULL**.

Tipe Data	Penjelasan	Contoh
INT, INTEGER	32 bit bilangan bulat. Nilai berkisar -2.1 miliar sampai +2.1 miliar. Jika <i>unsigned</i> (bernilai positif saja) maka nilai bisa berkisar 4.2 miliar.	123,345 -2,000,000,000
BIGINT	64 bit bilangan bulat. Nilai berkisar -9 juta miliar sampai +9 juta miliar atau 0 sampai 18 juta miliar nilai positif jika <i>unsigned</i>	9,000,000,000,000,000,000 -9,000,000,000,000,000,000
FLOAT	32 bit bilangan pecahan. Nilai berkisar antara -1.7e38 sampai +1.7e38. dan berkisar 0 sampai 3.4e38 jika bernilai positif	0.0000000000000002 17897.890790

	saja (<i>unsigned</i>).	-345.8908770
		1.7e21
DOUBLE	64 bit bilangan pecahan. Rentang nilai mendekati tak terbatas (1.7e308).	1.765e203 -1.765e100
DECIMAL <i>(presisi,skala)</i>	Angka yang mempertimbangkan penetapan angka desimal.	78979.00 -87.50
NUMERIC <i>(presisi,skala)</i>	Contoh untuk menyimpan nilai mata uang.	9.95
DATE	Penanggalan tanpa disertai waktu.	'1999-12-31'
DATETIME	Penanggalan yang disertai waktu.	'1999-12-31 23:59:59'
CHAR (lebar)	String character dengan lebar yang tetap.	'hello world '
VARCHAR (lebar)	String character dengan lebar tidak tetap.	'hello world'
BLOB, TEXT	Mampu menyimpan 64 ribu data, jika data-nya biner maka gunakan BLOB, jika data-nya teks maka gunakan TEXT.	
LONGBLOB, LONGTEXT	Serupa dengan BLOB dan TEXT namun mampu menyimpan data sampai 4GB.	

Contoh pendeklarasian variable:

```

DECLARE l_int1      int default -2000000;
DECLARE l_int2      INT unsigned default 4000000;
DECLARE l_bigint1   BIGINT DEFAULT 400000000000000000;
DECLARE l_float      FLOAT DEFAULT 1.8e8;
DECLARE l_double     DOUBLE DEFAULT 2e45;
DECLARE l_numeric    NUMERIC(8,2) DEFAULT 9.95;

```

```

DECLARE l_date      DATE DEFAULT '1999-12-31';
DECLARE l_datetime DATETIME DEFAULT '1999-12-31 23:59:59';
DECLARE l_char       CHAR(255) DEFAULT 'Karakter dengan alokasi tetap';
DECLARE l_varchar    VARCHAR(255) DEFAULT 'Karakter dengan alokasi
fleksibel';
DECLARE l_text       TEXT DEFAULT 'Digunakan untuk kalimat yang sangat
panjang.';

```

3.2.2 Literal

Literal adalah nilai data yang dituliskan ke dalam program. Penggunaan literal dalam variable bisa berupa pemberian nilai, perbandingan (misalnya pernyataan IF), sebagai argumen procedure atau function.

Sebuah literal adalah nilai tetap yang tidak berubah. Sebagai contoh saat menampilkan baris melalui pernyataan **SELECT** dari table **buku_tb** atau menambahkan baris melalui pernyataan **INSERT** dari table **penulis_tb**, seperti berikut ini:

```

SELECT judul
FROM buku_tb
WHERE tahun terbit > 2005

```

```

INSERT INTO penulis_tb (kode penulis, nama penulis, email, website)
VALUES ('0000000001', 'Handi Chandra', 'handi@yahoo.com',
        'www.handi.com')

```

Setiap literal memiliki tipe data tertentu yang mengacu pada kolom table sehingga antara tipe literal dengan tipe kolom table harus sesuai. Penentuan tipe kolom table dibuat saat pembuatan table melalui pernyataan **CREATE TABLE**.

Literal terbagi menjadi beberapa kelompok utama: literal **numerik**, **alphanumeric**, **temporal**, **boolean**, dan **heksadesimal**. Semua memiliki karakteristik dan batasannya sendiri-sendiri.

Definisi:

```

<literal> ::=

<numeric literal> |
<alphanumeric literal> |
<temporal literal> |
<boolean literal> |
<hexadecimal literal>

<numeric literal> ::=

<integer literal> |
<decimal literal> |
<float literal> |
<bit literal>

<integer literal> ::= [ + | - ] <whole number>

<decimal literal> ::=

[ + | - ] <whole number> [ .<whole number> ] |
[ + | - ] <whole number>. |
[ + | - ] .<whole number>

<float literal> ::=

<mantissa> { E | e } <exponent>

<bit literal> ::=

{ b | B } ' { 0 | 1 }... ' 

<alphanumeric literal> ::= <character list>

<temporal literal> ::=
<date literal> |
<time literal> |
<datetime literal> |
<timestampl literal> |
<year literal>

<date literal> ::=
{ ' <years> - <months> - <days> ' } |
{ <years> <months> <days> }

<time literal> ::=
{ ' <hours> : <minutes> [ : <seconds>
[ . <microseconds> ] ] ' } |
{ ' [ <hours> : <minutes> : ] <seconds> ' } |
{ <hours> <minutes> <seconds> } |
{ [ [ <hours> ] <minutes> ] <seconds> }

<datetime literal> ;

<timestampl literal> ::=
{ ' <years> - <months> - <days> <space>
[ <hours> [ : <minutes> [ : <seconds>
[ . <micro seconds> ] ] ] ' } |
{ <years> <months> <days> <hours> <minutes> <seconds> }

```

```

<year literal> ::= <year>

<hexadecimal literal> ::=
{ X | x } <hexadecimal character>... |
0x <hexadecimal character>...

<hexadecimal character> ::=
<digit> | A | B | C | D | E | F | a | b | c | d | e | f
<years> ;
<micro seconds> ;

<year> ::= <whole number>
<months> ;
<days> ;
<hours> ;
<minutes> ;

<seconds> ::= <digit> [ <digit> ]

<whole number> ::= <digit>...

<boolean literal> ::= TRUE | true | FALSE | false

<mantissa> ::= <decimal literal>

<exponent> ::= <integer literal>

<character list> ::= ' [ <character>... ] '

<character> ::= <digit> | <letter> | <special character> | ''

<special character> ::=
{ \ { 0 | ' | " | b | n | r | t | z | \ | % } } |
<any other character>

<whole number> ::= <digit>...

```

3.2.2.1 Literal Integer

MySQL mempunyai beberapa literal bertipe numerik, salah satu tipe numerik yang sering digunakan adalah tipe **integer**. Integer adalah semua angka tanpa disertai titik desimal, bisa diawali oleh tanda plus atau minus. Contohnya seperti berikut ini:

```

38
+12
-3404
016

```

Contoh literal integer yang salah:

```
342.16  
-14E5  
jan
```

3.2.2.2 Literal Desimal

Literal desimal adalah angka dengan atau tanpa titik desimal, bisa diawali oleh tanda plus atau minus. Contohnya seperti berikut ini:

```
49  
18.47  
-3400  
17.  
0.83459  
-.47
```

Semua digit angka yang tertera disebut dengan **presisi**, sedangkan angka digit setelah tanda titik adalah **skala**. Untuk literal desimal 123.45 berarti memiliki presisi 5 dengan skala 2. Skala untuk literal integer adalah 0. Rentang maksimal literal desimal diukur berdasarkan skala dan presisi. Presisi harus lebih besar dari 0, dan skala harus berada diantara 0 dan nilai presisinya. Sebagai contoh, sebuah angka desimal dengan presisi 8 dan skala 2 adalah benar, tetapi jika presisinya 6 dan skala 8 adalah salah.

3.2.2.3 Literal Float, Real, dan Double

Literal float adalah literal desimal yang diikuti oleh eksponen. Berikut contoh literal float:

Literal Float	Nilai
-34E2	-3400
0.16E4	1600
4E-3	0.004
4e-3	0.004

3.2.2.4 Literal Alphanumeric

Literal **alphanumeric** adalah semua character alphanumeric termasuk string kosong yang berada diantara tanda petik ganda ("...") atau tanda petik tunggal ('...'). Berikut ini character yang diperbolehkan dalam literal alphanumeric:

- Semua huruf kecil (a sampai z)
- Semua huruf besar (A sampai Z)
- Semua digit angka (0 sampai 9)
- Semua character sisanya (seperti: ', +, -, ?, =, and _)

Bisa jadi literal alphanumeric mengandung tanda petik. Untuk menyatakan tanda petik tunggal di dalam literal alphanumeric, dibutuhkan dua tanda petik. Berikut beberapa contoh penggunaan literal alphanumeric yang benar:

Literal Alphanumeric	Nilai
'Dodit'	Dodit
"Dodit"	Dodit
'Jum''at'	Jum'at
'!?-@'	!?-@
''	
'''	'
'''	"
'1234'	1234

Contoh salah dari penggunaan literal alphanumeric adalah:

```
'Surip  
''mbah  
'''
```

Literal alphanumeric mungkin juga mengandung character khusus (*special character*) misalnya *carriage return*, *tab*, *backspace*, dan lain sebagainya. Penulisan character khusus harus diawali dengan tanda *slash* (\). Tabel berikut ini menjelaskan daftar beberapa character khusus yang ada:

Character Khusus	Penjelasan
\0	Character ASCII 0 (nol)
\'	Tanda petik tunggal
\\"	Tanda petik ganda
\b	Backspace
\n	Baris baru
\r	Enter
\t	Character tab
\z	Character ASCII 26, atau Ctrl+Z
\\	slash

3.2.2.5 Literal Date/Tanggal

Literal date/tanggal terdiri dari tiga komponen, antara lain: **tahun**, **bulan**, dan **hari**. MySQL membolehkan literal tanggal untuk ditulis dalam bentuk literal **alphanumeric** atau literal **integer**.

Jika literal alphanumeric digunakan maka seluruh nilai harus berada diantara tanda petik, ketiga komponen (tahun, bulan, dan tanggal) harus dipisahkan oleh character khusus (biasanya tanda hubung '-') meskipun character lain seperti: /, @, dan % masih bisa digunakan. Contoh literal tanggal:

Literal Tanggal	Nilai
'1980-12-08'	December 8, 1980
'1991-6-19'	June 19, 1991
'1991@6@19'	June 19, 1991

Dalam banyak kasus komponen tahun dituliskan dalam empat digit angka. Jika menyimpang dari itu maka MySQL akan menggunakan beberapa aturan untuk menentukan apa maksudnya, antara lain :

- Bila terdapat tiga digit angka maka angka nol ditempatkan di depan.

- Bila tahun dituliskan dalam dua digit angka maka saat angkanya berada diantara 00 sampai 69 nilai akan ditambah dengan 2000, jika tidak memenuhi maka akan ditambah dengan 1900.
- Bila tahun hanya terdiri dari satu digit maka tiga angka nol akan ditempatkan didepannya.

Berikut contoh penerapan tiga aturan di atas:

Literal Tanggal	Nilai
'999-10-11'	October 11, 0999
551011	October 11, 2055
'99-10-11'	October 11, 1999
'9-10-11'	October 11, 0009

Pembaca disarankan untuk menggunakan alphanumeric tanda hubung ('-') saat memisahkan setiap komponen (tahun, bulan, tanggal) karena bentuk ini lebih mudah dibaca dan jelas. Gunakan pula empat digit angka pada komponen tahun untuk mengurangi tingkat kesalahan.

3.2.2.6 Literal Time/Waktu

Literal time/waktu terdiri dari empat komponen, antara lain: **jam**, **menit**, **detik**, dan **mikrodetik**. Literal waktu dapat dituliskan dalam bentuk **alphanumeric** maupun **integer** keduanya bisa diterima dengan baik.

Jika menggunakan alphanumeric maka keseluruhan nilai harus berada diantara tanda petik. Pembagian setiap komponen biasanya dipisahkan oleh tanda titik dua (':'), meskipun character khusus lainnya masih bisa digunakan seperti -, /, @, atau %. Di depan komponen mikrodetik secara normal disertai dengan tanda titik. Angka nol menyimpang yang berada diketiga komponen pertama akan diabaikan.

Beberapa aturan literal waktu:

- Jika terdapat hanya dua komponen yang ditetapkan maka MySQL menganggap komponen tersebut sebagai komponen jam dan menit.
- Jika terdapat hanya satu komponen yang ditetapkan maka akan dianggap sebagai komponen detik.

Lihat contoh berikut:

Literal Waktu	Nilai
'23:59:59'	1 detik sebelum tengah malam.
'12:10:00'	10 menit setelah jam 12 siang
'14:00'	Jam 2 siang atau 14:00:00
'14'	14 detik setelah tengah malam atau 00:00:14
'00:00:00.000013'	13 mikrodetik setelah tengah malam

Jika penulisan literal waktu/time menggunakan literal integer maka tiga komponen pertama harus ditulis tanpa character pemisah. MySQL akan menganggap :

- Dua digit terakhir adalah komponen detik.
- Dua digit di depannya adalah komponen menit.
- Apapun yang berada diposisi paling depan adalah jam.

Literal Waktu	Nilai
235959	1 detik sebelum tengah malam atau 23:59:59
121000	10 menit setelah jam 12 siang atau 12:10:00
1400	14 menit setelah tengah malam atau 00:14:00
14	14 detik setelah tengah malam atau 00:00:14
000000.000013	13 mikrodetik setelah tengah malam

3.2.2.7 Literal Datetime dan Timestamp

Nilai literal **datetime** dan **timestamp** merupakan kombinasi dari literal date, literal time, dan tambahan komponen mikrodetik.

Setiap literal datetime dan timestamp terdiri dari tujuh komponen, antara lain: **tahun**, **bulan**, **hari**, **jam**, **menit**, **detik**, dan **mikrodetik**, penulisannya bisa menggunakan **alphanumeric** maupun **integer**.

Jika menggunakan alphanumeric maka tiga komponen pertama menunjukkan tanggal yang dipisahkan oleh tanda hubung ('-'), tanda titik dua untuk memisahkan tiga komponen selanjutnya menunjukkan waktu, tanda spasi yang berada diantara tanggal dan waktu, titik desimal diletakkan di depan mikrodetik.

Literal Timestamp	Nilai
'1980-12-08 23:59:59.99999'	1 mikrodetik sebelum tengah malam 8 desember 1980
'1991-6-19 12:5:00'	5 menit setelah jam 12 siang 19 Juni 1991

3.2.2.8 Literal Boolean

Literal yang paling sederhana adalah **Boolean** karena hanya terdiri dari dua kemungkinan nilai: **TRUE** dan **FALSE**. Nilai numerik pengganti dari nilai FALSE diwakili oleh angka 0 sedangkan TRUE diwakili oleh angka 1.

Contoh untuk mendapatkan nilai literal TRUE dan FALSE.

```
SELECT TRUE, FALSE
```

Hasilnya:

TRUE	FALSE
-----	-----
1	0

3.2.2.9 Literal Hexadecimal

Literal ini dituliskan sebagai literal alphanumeric dimana posisi terdepan ditempati huruf X atau x. Diantara tanda petik hanya ada sepuluh digit angka dan huruf yang dapat diterima adalah A sampai F.

Format heksadesimal utamanya digunakan untuk menyimpan nilai-nilai khusus di database, misalnya format gambar (JPG atau BMP), film (AVI atau MPG).

Contoh penggunaan literal heksadesimal:

Literal Heksadesimal	Nilai
X'41'	A
X'6461746162617365'	database
X'3B'	;

3.2.2.10 Literal Bit

Literal Bit adalah literal numerik yang dituliskan sebagai literal alphanumeric dengan **b** huruf kecil atau **B** huruf besar di depannya. Diantara tanda petik hanya nilai **1** dan **0** yang diperbolehkan.

Contoh literal bit:

Literal Bit	Nilai
b'1001'	9
b'1111111'	127
b'0001'	1

3.3 Pemberian Nilai Ke Variable

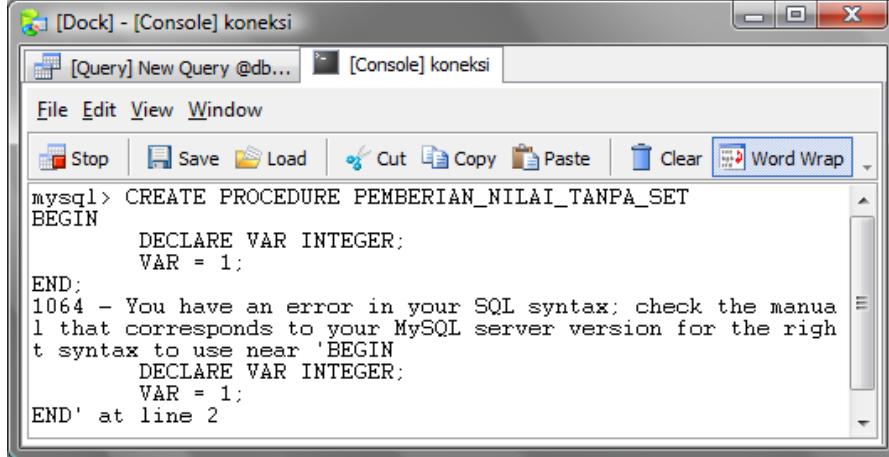
Anda dapat memanipulasi (mengisi, mengganti) nilai variable dengan pernyataan **SET**, seperti sintaks berikut ini:

```
SET variable_name = expression [,variable_name = expression ...]
```

Berikut ini contoh yang salah karena memberikan nilai pada variable tanpa menggunakan pernyataan **SET**:

```
CREATE PROCEDURE PEMBERIAN_NILAI_TANPA_SET
BEGIN
    DECLARE VAR INTEGER;
    VAR = 1; → Seharusnya SET VAR = 1;
END;
```

[Err] 1064 - You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near 'BEGIN
 DECLARE VAR INTEGER;
 VAR = 1;
END' at line 2



3.4 Parameter Store Procedure

Store procedure bisa memiliki satu parameter, banyak parameter, atau tanpa parameter. Melalui paramater tersebut store procedure mampu berkomunikasi dengan lingkungan luar. Terdapat tiga jenis parameter yang didukung oleh store procedure. Parameter adalah variable pada program store (stored procedure / store function) yang dapat dilewati oleh suatu nilai baik kedalam maupun keluar saat program store dipanggil. Parameter program store ditetapkan bersamaan dengan pernyataan CREATE FUNCTION atau CREATE PROCEDURE, seperti berikut ini:

```
CREATE PROCEDURE|FUNCTION (
[ [ IN
| OUT
| INOUT
] nama_parameter tipe_data...])
```

Parameter dapat diasosiasikan dengan atribut **IN**, **OUT** atau **INOUT**:

- Atribut **IN** – Jika parameter tidak diawali oleh atribut **IN** maka secara default parameter sudah dianggap beratribut **IN**. Parameter beratribut **IN** menyebabkan segala modifikasi parameter yang dibuat dalam program store tidak dapat diakses dari program pemanggilnya.
- **OUT** – Nilai parameter **OUT** yang dimodifikasi dalam program store dapat diakses dari program pemanggilnya. Program pemanggil harus menyediakan variable untuk menerima keluaran parameter **OUT** tetapi program store itu sendiri tidak mempunyai hak untuk memberi nilai inisialisasi ke variable.
- **INOUT** - Parameter **INOUT** bertindak sebagai parameter **IN** dan **OUT**. Program pemanggil boleh menyediakan nilai, program store itu sendiri boleh memodifikasi nilai parameter, dan program pemanggil bisa mengakses nilai setelah mengalami perubahan ketika program store dieksekusi.

Kata kunci IN, OUT dan INOUT hanya dapat diaplikasikan pada store procedure dan tidak berlaku untuk store function. Dalam stored function semua parameter berlaku sebagai parameter IN. Contoh penggunaan parameter IN adalah sebagai berikut, tuliskan script berikut melalui menu Tools->Console:

```
mysql> DROP PROCEDURE IF EXISTS DEMO_PARAMETER_IN;
CREATE PROCEDURE DEMO_PARAMETER_IN (IN PARAM_IN INT)
BEGIN
    /* Menampilkan nilai PARAM_IN */
    SELECT PARAM_IN;
    /* Mengubah nilai PARAM_IN dengan 2*/
    SET PARAM_IN = 2;
    /* Menunjukkan apakah PARAM_IN terpengaruh */
    SELECT PARAM_IN;
END;
Query OK, 0 rows affected

Query OK, 0 rows affected

mysql> SET @PARAM_IN=1;
Query OK, 0 rows affected

mysql> CALL DEMO_PARAMETER_IN (@PARAM_IN);
+-----+
| PARAM_IN |
+-----+
|      1   |
+-----+
1 row in set

+-----+
| PARAM_IN |
+-----+
|      2   |
+-----+
1 row in set

Query OK, 0 rows affected

mysql> SELECT @PARAM_IN, 'Kita tidak dapat melihat perubahan nilai
dari program pemanggilnya';
+-----+-----+
| @PARAM_IN | Kita tidak dapat melihat perubahan nilai dari program
pemanggilnya |
+-----+-----+
|      1   | Kita tidak dapat melihat perubahan nilai dari program
pemanggilnya |
+-----+
```

```
+-----+  
-----+  
1 row in set
```

Contoh penggunaan parameter **OUT**, tuliskan script berikut melalui menu **Tools->Console**:

```
mysql> DROP PROCEDURE IF EXISTS DEMO_PARAMETER_OUT;  
CREATE PROCEDURE DEMO PARAMETER OUT (OUT PARAM OUT INT)  
BEGIN  
    /* Kita tidak dapat melihat nilai dari parameter OUT */  
    SELECT PARAM OUT, 'Kita tidak dapat melihat nilai dari  
parameter OUT';  
    /* Mengubah nilai PARAM_OUT menjadi 2 */  
    SET PARAM_OUT = 2;  
    SELECT PARAM OUT, 'Nilai parameter OUT telah berubah';  
END;  
Query OK, 0 rows affected  
  
Query OK, 0 rows affected  
  
mysql> SET @PARAM_OUT=1;  
Query OK, 0 rows affected  
  
mysql> CALL DEMO_PARAMETER_OUT (@PARAM_OUT);  
+-----+-----+  
| PARAM_OUT | Kita tidak dapat melihat nilai dari parameter OUT |  
+-----+-----+  
| NULL      | Kita tidak dapat melihat nilai dari parameter OUT |  
+-----+-----+  
1 row in set  
  
+-----+-----+  
| PARAM_OUT | Nilai parameter OUT telah berubah |  
+-----+-----+  
|        2 | Nilai parameter OUT telah berubah |  
+-----+-----+  
1 row in set  
  
Query OK, 0 rows affected  
  
mysql> SELECT @PARAM_OUT, "Program pemanggil dapat melihat nilai dari  
parameter OUT setelah mengalami perubahan";  
+-----+-----+  
| @PARAM_OUT | Program pemanggil dapat melihat nilai dari parameter  
OUT setelah mengalami perubahan |  
+-----+-----+  
|          2 | Program pemanggil dapat melihat nilai dari parameter  
OUT setelah mengalami perubahan |
```

```
+-----+  
-----+  
1 row in set
```

Contoh penggunaan parameter **INOUT**, tuliskan script berikut melalui menu **Tools->Console**:

```
mysql> DROP PROCEDURE IF EXISTS DEMO_PARAMETER_INOUT;  
CREATE PROCEDURE DEMO_PARAMETER_INOUT (INOUT PARAM INOUT INT)  
BEGIN  
    SELECT PARAM INOUT, 'Kita dapat melihat nilai dari parameter  
INOUT di dalam store program';  
    SET PARAM INOUT = 2;  
    SELECT PARAM_INOUT, 'Nilai parameter INOUT telah berubah';  
END;  
Query OK, 0 rows affected  
  
Query OK, 0 rows affected  
  
mysql> SET @PARAM_INOUT = 1;  
Query OK, 0 rows affected  
  
mysql> CALL DEMO_PARAMETER_INOUT(@PARAM_INOUT);  
+-----+  
| PARAM_INOUT | Kita dapat melihat nilai dari parameter INOUT di  
dalam store program |  
+-----+  
-----+  
| 1 | Kita dapat melihat nilai dari parameter INOUT di  
dalam store program |  
+-----+  
-----+  
1 row in set  
  
+-----+-----+  
| PARAM_INOUT | Nilai parameter INOUT telah berubah |  
+-----+-----+  
| 2 | Nilai parameter INOUT telah berubah |  
+-----+-----+  
1 row in set  
  
Query OK, 0 rows affected  
  
mysql> SELECT @PARAM_INOUT, "Program pemanggil dapat melihat nilai  
dari parameter INOUT yang telah berubah";  
+-----+  
-----+  
| @PARAM_INOUT | Program pemanggil dapat melihat nilai dari parameter  
INOUT yang telah berubah |
```

```
+-----+  
|          2 | Program pemanggil dapat melihat nilai dari parameter  
INOUT yang telah berubah |  
+-----+  
-----+  
1 row in set
```

3.5 Komentar

Ada dua model pemberian komentar dalam program store yang didukung oleh MySQL:

- Dua tanda minus (--) kemudian dilanjutkan dengan komentar. Cara ini digunakan untuk komentar satu baris tunggal.
- Model bahasa C, diawali dengan /* dan diakhiri dengan */. Cara ini digunakan untuk komentar banyak baris.

Contoh penggunaan komentar tunggal:

```
-- -----  
-- Table structure for buku_tb  
-- -----
```

Contoh penggunaan komentar banyak baris:

```
/*  
Navicat MySQL Data Transfer  
Source Host      : localhost:3306  
Source Database  : dbpenjualanbuku  
Target Host     : localhost:3306  
Target Database : dbpenjualanbuku  
Date: 12/07/2009 0:00:12  
*/
```

3.6 Operator

Operator MySQL memiliki banyak kemiripan dengan operator pada bahasa pemrograman lainnya meskipun ada beberapa operator dari bahasa

pemrograman tersebut yang tidak dikenali oleh MySQL, contohnya operator model bahasa C misalnya operator: ++, +=, dan --. MySQL juga mengenal operator perbandingan, operator seleksi kondisi, operator pemberi nilai, operator khusus (SET), dan operator perulangan. Berikut ini contoh program store yang memanfaatkan beberapa operator:

```
mysql> DROP PROCEDURE IF EXISTS OPERATORS;
CREATE PROCEDURE OPERATORS()
BEGIN
    DECLARE a int default 2;
    declare b int default 3;
    declare c FLOAT;

    set c=a+b; select 'a+b=',c;
    SET c=a/b; select 'a/b=',c;
    SET c=a*b; Select 'a*b=',c;

    IF (a<b) THEN
        select 'a kurang dari b';
    END IF;
    IF NOT (a=b) THEN
        SELECT 'a tidak sama dengan b';
    END IF;
END;

Query OK, 0 rows affected

Query OK, 0 rows affected

mysql> CALL OPERATORS;
+-----+---+
| a+b= | c |
+-----+---+
| a+b= | 5 |
+-----+---+
1 row in set

+-----+-----+
| a/b= | c      |
+-----+-----+
| a/b= | 0,666667 |
+-----+-----+
1 row in set

+-----+---+
| a*b= | c |
+-----+---+
| a*b= | 6 |
+-----+---+
1 row in set
```

```

+-----+
| a kurang dari b |
+-----+
| a kurang dari b |
+-----+
1 row in set

+-----+
| a tidak sama dengan b |
+-----+
| a tidak sama dengan b |
+-----+
1 row in set

Query OK, 0 rows affected

```

3.6.1 Operator Matematika

MySQL mendukung operator matematika dasar, misalnya penambahan (+), pengurangan (-), perkalian (*), dan pembagian (/). Selain itu ada operator matematika tambahan seperti division (**DIV**) yaitu mengambil nilai hasil dari pembagian bilangan bulat, serta operator modulus (%) yaitu mengambil nilai sisa pembagian.

Operator	Penjelasan	Contoh
+	Penambahan	SET var1=2+2; →4
-	Pengurangan	SET var2=3-2; →1
*	Pengalian	SET var3=3*2; →6
/	Pembagian	SET var4=10/3; →3.3333
DIV	Pembagian bulat	SET var5=10 DIV 3; →3
%	Pembagian sisa, modulus	SET var6=10%3 ; →1

3.6.2 Operator Perbandingan

Operator perbandingan bertujuan untuk membandingkan nilai-nilai tertentu dan menghasilkan nilai kebenaran **TRUE**, **FALSE** atau **UNKNOWN** (biasanya jika salah satu nilai yang sedang dibandingkan bernilai

UNKNOWN atau **NULL**). Operator perbandingan pada umumnya digunakan pada ekspresi **IF**, **CASE** dan pernyataan perulangan.

Operator	Penjelasan	Contoh	Hasil
>	Lebih dari	1>2	False
<	Kurang dari	2<1	False
<=	Kurang dari atau sama dengan	2<=2	True
>=	Lebih dari atau sama dengan	3>=2	True
BETWEEN	Nilai berada diantara dua nilai	5 BETWEEN 1 AND 10	True
NOT BETWEEN	Nilai tidak berada diantara dua nilai	5 NOT BETWEEN 1 AND 10	False
IN	Nilai termasuk di dalam daftar	5 IN (1,2,3,4)	False
NOT IN	Nilai tidak termasuk di dalam daftar	5 NOT IN (1,2,3,4)	True
=	Sama dengan	2=3	False
<>, !=	Tidak sama dengan	2<>3	False
LIKE	Sesuai dengan pola yang diberikan	"Dodit Suprianto" LIKE "Dod%"	True
REGEXP	Sesuai dengan pola <i>extended regular expression</i>	"Guy Harrison" REGEXP "[Gg]reg"	False
IS NULL	Nilai adalah NULL	0 IS NULL	False
IS NOT NULL	Nilai adalah tidak NULL	0 IS NOT NULL	True

3.6.3 Operator Logika

Operator logika mengoperasikan tiga nilai logika, antara lain: **TRUE**, **FALSE**, dan **NULL**. Operator tersebut digunakan khusus pada operator

perbandingan untuk membuat ekspresi yang lebih rumit. Jika salah satu nilai yang sedang dibandingkan adalah **NULL** maka hasilnya juga **NULL**.

Tabel kebenaran pada operator **AND**:

AND	TRUE	FALSE	NULL
TRUE	TRUE	FALSE	NULL
FALSE	FALSE	FALSE	NULL
NULL	NULL	NULL	NULL

Tabel kebenaran pada operator **OR**:

OR	TRUE	FALSE	NULL
TRUE	TRUE	TRUE	TRUE
FALSE	TRUE	FALSE	NULL
NULL	TRUE	NULL	NULL

Tabel kebenaran pada operator **XOR**:

XOR	TRUE	FALSE	NULL
TRUE	FALSE	TRUE	NULL
FALSE	TRUE	FALSE	NULL
NULL	NULL	NULL	NULL

Contoh penggunaan operator logika:

```
DROP FUNCTION IF EXISTS PANGGILAN_FN;
CREATE FUNCTION PANGGILAN FN
(
    in_gender CHAR(1),
    in age INT,
    in marital status VARCHAR(7)
)
RETURNS VARCHAR(6)
BEGIN
    DECLARE title VARCHAR(6);
    IF in_gender='F' AND in_age<16 THEN
        SET title='Miss';
```

```

ELSEIF in gender='F' AND in age>=16 AND in marital status='Married'
THEN
    SET title='Mrs';
ELSEIF in_gender='F' AND in_age>=16 AND in_marital_status='Single'
THEN
    SET title='Ms';
ELSEIF in gender='M' AND in age<16 THEN
    SET title='Master';
ELSEIF in_gender='M' AND in_age>=16 THEN
    SET title='Mr';
END IF;
RETURN(title);
END;

```

3.6.4 Operator Bitwise

Operator bitwise digunakan untuk mengoperasikan bilangan biner. Tabel operator bitwise:

Operator	Use
	OR
&	AND
<<	Geser bit ke kiri
>>	Geser bit ke kanan
~	NOT atau membalik bit

Operator bitwise serupa dengan operator logika, kecuali bahwa operasi berlaku pada nilai bit (biner) di dalam variable.

Sebagai contoh, integer 5 (dalam biner 101) dan 4 (dalam biner 010). Operator OR akan mengubah dengan sendirinya angka tersebut menjadi format biner; jadi **5|2=7** karena **101|010=111** yang artinya 7 dalam desimal. Ekspresi **5&6=7** karena **101&110=100** yang artinya 4 dalam desimal.

3.7 Ekspresi

Ekspresi merupakan kombinasi dari **literal**, **variable**, dan **operator** untuk menghasilkan suatu nilai. Contoh ekspresi:

```
Myvariable_name  
Myvariable name+1  
ABS (Myvariable name)  
3.14159  
IF(Myvariable='M','Male','Female')  
2+4) /12
```

3.8 Function Built-In

Function Built-In adalah fungsi-fungsi yang telah disediakan oleh MySQL. Anda tinggal menggunakannya tanpa harus membuat *function* itu sendiri. Semua function built-in telah terdokumentasi dengan baik pada **MySQL reference manual**. Function MySQL terbagi menjadi beberapa kategori:

- **Function String**

Function ini beroperasi pada string. Sebagai contoh, Anda dapat menggabungkan string, mencari character dalam string, mencari bagian string dalam string (substring), dan lain sebagainya.

- **Function Matematika**

Function ini beroperasi pada angka. Sebagai contoh, Anda dapat mencari nilai eksponensial, fungsi trigonometri, fungsi nilai acak (random), logaritma, dan lain sebagainya.

- **Function Tanggal dan Waktu**

Fungsi-fungsi ini beroperasi pada tanggal dan waktu. Sebagai contoh, Anda bisa mendapatkan tanggal saat ini, menambahkan atau

mengurangi antar tanggal sebagai rentang waktu, menguraikan tanggal menjadi hari, bulan, tahun, dan lain sebagainya.

- **Function Lain-lain**

Fungsi ini meliputi semua fungsi yang tidak mudah untuk digolongkan seperti kategori di atas.

Berikut contoh Function MySQL yang sering digunakan:

Function	Description
ABS(<i>number</i>)	Mengembalikan nilai absolute. Contoh: ABS(-2.3)=2.3.
CEILING(<i>number</i>)	Membulatkan ke atas suatu nilai pecahan. Contoh: CEILING(2.3)=3.
CONCAT(<i>string1[,string2,string3,...]</i>)	Menggabungkan beberapa string menjadi satu string saja.
CURDATE	Mengembalikan nilai tanggal saat ini (tanpa disertai waktu).
DATE_ADD(<i>date</i>,INTERVAL <i>amount_type</i>)	Menambahkan rentang waktu (<i>interval</i>) terhitung dari tanggal (<i>date</i>), dimana tipe (<i>amount_type</i>) yang berlaku adalah SECOND, MINUTE, HOUR, DAY, MONTH, dan YEAR. Menghasilkan tanggal baru
DATE_SUB(<i>date</i>,INTERVAL <i>interval_type</i>)	Pengurangan tanggal (<i>date</i>) dengan rentang waktu <i>interval</i> , dimana tipe (<i>interval_type</i>) yang berlaku adalah SECOND, MINUTE, HOUR, DAY, MONTH, and YEAR.
FORMAT(<i>number,decimals</i>)	memformat angka (<i>number</i>) dalam bentuk desimal dengan jumlah digit sebanyak <i>decimals</i> .
GREATEST(<i>num1,num2[,num3, ...]</i>)	Mengembalikan nilai terbesar dari

	seluruh nilai argumen yang tersedia.
IF(<i>test, value1,value2</i>)	Menguji kondisi logika. Jika test bernilai TRUE, maka mengembalikan <i>value1</i> ; sebaliknya mengembalikan <i>value2</i> .
INSERT(<i>string,position,length,new</i>)	Menyisipkan string <i>new</i> ke <i>string</i> , diawali pada posisi ke <i>position</i> dan sebanyak <i>length</i> character.
INSTR(<i>string,substring</i>)	Mencari lokasi <i>substring</i> diantara <i>string</i> jika ada.
ISNULL(<i>expression</i>)	Mengembalikan 1 jika argumen (<i>expression</i>) bernilai NULL, dan sebaliknya akan bernilai 0.
LEAST(<i>num1,num2[,num3, ...]</i>)	Mengembalikan nilai terkecil dari angka-angka yang terdapat pada argumen (<i>num1, num2, num3,...</i>)
LEFT(<i>string,length</i>)	Mengembalikan string baru yang diperoleh dari <i>string</i> dimulai dari kiri sebanyak <i>length</i> character.
LENGTH(<i>string</i>)	Menghitung lebar/jumlah character <i>string</i> .
LOCATE(<i>substring,string[,number]</i>)	Returns the location of the substring within the string, optionally starting the search at the position given by the third argument.
LOWER(<i>string</i>)	Mengubah <i>string</i> menjadi huruf kecil semua.
LTRIM(<i>string</i>)	Membuang spasi kosong sebelah kiri dari <i>string</i> .
MOD(<i>num1,num2</i>)	Memperoleh nilai sisa bagi (modulus) hasil dari pembagian <i>num1</i> dengan <i>num2</i> .

NOW	Mengembalikan tanggal dan waktu saat ini.
POWER(<i>num1,num2</i>)	Pemangkatan. Mengembalikan bilangan <i>num1</i> pangkat <i>num2</i> .
RAND([<i>seed</i>])	Mengembalikan nilai acak dengan <i>seed</i> sebagai awalan pembangkit bilangan acak.
REPEAT(<i>string,number</i>)	Returns a string consisting of <i>number</i> repetitions of the given <i>string</i> .
REPLACE(<i>string,old,new</i>)	Menimpa string <i>old</i> dengan string <i>new</i> dari string <i>string</i> yang disediakan.
ROUND(<i>number[,decimal</i>])	Membulatkan nilai <i>number</i> menjadi bilangan pecahan dengan digit sebanyak <i>decimal</i> .
RTRIM(<i>string</i>)	Membuang semua spasi kosong sebelah kanan dari <i>string</i> .
SIGN(<i>number</i>)	Mengembalikan -1 jika <i>number</i> kurang dari 0, 1 jika <i>number</i> lebih dari 0, atau 0 jika <i>number</i> sama dengan 0.
SQRT(<i>number</i>)	Mengembalikan nilai akar kwadrat dua dari <i>number</i> .
strcmp(<i>string1,string2</i>)	Mengembalikan 0 jika <i>string1</i> dan <i>string2</i> identik. Bernilai -1 jika <i>string1</i> lebih pendek dari Returns 0 if the two strings are identical, -1 if the first string would sort earlier than the second string, or 1 otherwise.
SUBSTRING(<i>string,position,length</i>)	Mengambil sebagian nilai <i>string</i> , dimulai dari <i>position</i> , sebanyak <i>length</i> character.

UPPER(string)	Mengubah string menjadi huruf besar semua.
VERSION	Mengembalikan string berisi informasi MySQL server yang digunakan saat ini.

Contoh penggunaan function:

```

mysql> DROP PROCEDURE IF EXISTS CONTOH FUNCTION SP;
CREATE PROCEDURE CONTOH FUNCTION SP()
BEGIN
    DECLARE dua_puluhan_tahun_yang_lalu_dari_sekarang DATE;
    DECLARE mystring VARCHAR(250);

    SET dua_puluhan_tahun_yang_lalu_dari_sekarang = DATE_SUB(CURDATE(),
INTERVAL 20 YEAR);

    SET mystring = CONCAT('Saat itu ',
dua_puluhan_tahun_yang_lalu_dari_sekarang, ' aku masih di sekolah
dasar');

    SELECT mystring;

    IF (CAST(SUBSTR(VERSION(),1,3) AS DECIMAL(2,1)) <5.0) THEN
        SELECT 'MySQL sebelum versi 5.0 tidak dapat menjalankan program
store';
    ELSE
        SELECT 'selamat Anda telah menggunakan MySQL versi 5.0 atau
lebih!';
    END IF;

END;
Query OK, 0 rows affected

Query OK, 0 rows affected

mysql> CALL CONTOH FUNCTION SP ();
+-----+
| mystring |
+-----+
| Saat itu 1989-07-13 aku masih di sekolah dasar |
+-----+
1 row in set

+-----+
| selamat Anda telah menggunakan MySQL versi 5.0 atau lebih! |
+-----+
| selamat Anda telah menggunakan MySQL versi 5.0 atau lebih! |
+-----+

```

```
1 row in set  
Query OK, 0 rows affected
```

3.9 Blok, Pernyataan Kondisi, dan Perulangan

Struktur blok pada MySQL bertujuan untuk mengelompokkan dan menjalankan pernyataan-pernyataan secara bersama dalam satu blok. Program store MySQL mendukung dua jenis pernyataan kontrol, antara lain: pernyataan kontrol kondisi dan pernyataan perulangan.

Kontrol kondisi memiliki kemampuan untuk mengendalikan aliran eksekusi melalui program yang telah terkondisi tersebut. Seperti **IF-THEN-ELSE** dan pernyataan **CASE**.

Kontrol perulangan bertujuan untuk menjalankan satu atau lebih kode program secara berulang-ulang. MySQL menyediakan tiga jenis bentuk perulangan berbeda, antara lain:

- **Perulangan Sederhana**, Berlanjut sampai bertemu dengan pernyataan LEAVE untuk menghentikan perulangan.
- **Perulangan REPEAT...UNTIL**, Berlanjut selama ekspresi yang mengevaluasi bernilai benar (pengecekan kebenaran berada diakhir).
- **Perulangan WHILE...DO**, Berlanjut selama ekspresi yang mengevaluasi bernilai benar (pengecekan kebenaran berada diawal).

3.9.1 Struktur Blok Program Store

Setiap blok diawali oleh pernyataan **BEGIN** dan diakhiri oleh pernyataan **END**. Kasus sederhana penggunaan blok adalah pada program store seperti **CREATE PROCEDURE**, **CREATE FUNCTION** atau **CREATE TRIGGER**,

kemudian diikuti oleh blok tunggal yang berisi kode program untuk dijalankan, seperti berikut ini:

```
CREATE {PROCEDURE|FUNCTION|TRIGGER} nama program  
BEGIN  
    Pernyataan_program  
END;
```

Blok mempunyai dua tujuan utama:

- Untuk mengelompokkan beberapa pernyataan untuk dijalankan sekaligus. Semua pernyataan yang terlingkupi oleh blok dianggap sebagai satu kesatuan untuk dijalankan.
- Dengan blok maka pendefinisian variable yang diletakkan diantara blok tidak akan dikenali di luar blok, sehingga blok akan mempengaruhi jangkauan siklus hidup variable (*scope*).

3.9.2 Struktur Blok

Dalam blok bisa berisi berbagai jenis deklarasi misalnya: *variable*, *cursor*, *handler* dan beberapa baris kode program misalnya: pemberian nilai, pernyataan kondisi, perulangan. Urutan penulisan antara deklarasi dan kode program yang memungkinkan adalah sebagai berikut:

1. Deklarasi variable dan kondisi.
2. Deklarasi *Cursor*.
3. Deklarasi *Handlers*.
4. Kode Program.

Jika Anda melanggar urutan penulisan perintah misalnya meletakkan pernyataan **DECLARE** setelah pernyataan **SET** maka MySQL akan membangkitkan pesan kesalahan ketika Anda membuat kode program store.

Anda juga dapat memberi nama blok dengan label. Label diletakkan sebelum pernyataan **BEGIN** dan setelah pernyataan **END**. Pemberian label blok mempunyai keuntungan:

- Kode mudah dibaca karena mengetahui awal dan akhir blok berdasarkan labelnya.
- Membolehkan Anda untuk menghentikan eksekusi blok dengan pernyataan LEAVE.

Contoh sederhana struktur blok:

```
[label:] BEGIN
    Deklarasi variable dan kondisi
    Deklarasi cursor
    Deklarasi handler
    Kode program
END[label];
```

3.9.3 Blok Berulang

Jika semua program store hanya terdiri dari satu blok maka struktur blok akan sulit dipahami sehingga memungkinkan jika banyak kode program akan melibatkan beberapa blok, minimal terdapat blok utama yang melingkupi kode program seluruhnya.

Contoh bagaimana menciptakan variable dalam blok. Variable tidak akan dikenali diluar blok sehingga contoh berikut ini akan menghasilkan pesan kesalahan:

```
mysql> DROP PROCEDURE IF EXISTS BLOK_BERULANG_SP;
CREATE PROCEDURE BLOK_BERULANG_SP( )
BEGIN
    DECLARE variable_luar VARCHAR(20);
    BEGIN
        DECLARE variable_dalam VARCHAR(20);
        SET variable_dalam='Variable yang berada di blok lebih dalam';
    END;
    SELECT variable_dalam, ' menyebabkan kesalahan karena variable
dibaca dari luar blok dalam';
END;
```

```
Query OK, 0 rows affected
Query OK, 0 rows affected

mysql> CALL BLOK_BERULANG_SP();
1054 - Unknown column 'variable_dalam' in 'field list'
```

Variable yang dideklarasikan di luar blok (berada pada blok utama) juga akan dikenali pada blok yang lebih dalam.

```
mysql> DROP PROCEDURE IF EXISTS BLOK_BERULANG2_SP;
CREATE PROCEDURE BLOK_BERULANG2_SP( )
BEGIN
    DECLARE my_variable varchar(50);
    SET my_variable='pemberian nilai berada diblok utama';
    BEGIN
        SET my_variable='pemberian nilai berada diblok lebih dalam';
    END;
    SELECT my_variable, ' perubahan nilai masih tampak di luar blok ';
END;

Query OK, 0 rows affected
Query OK, 0 rows affected

mysql> CALL BLOK_BERULANG2_SP();
+-----+-----+
| my_variable | perubahan nilai masih tampak di luar blok |
+-----+-----+
| pemberian nilai berada diblok lebih dalam | perubahan nilai masih tampak di luar blok |
+-----+-----+
1 row in set

Query OK, 0 rows affected
```

Sekarang kita akan membuat variable dengan nama sama, baik yang berada di luar blok maupun di dalam blok. Ketika nilai variable yang berada di dalam blok berubah maka perubahannya tidak akan mempengaruhi variable di luar blok meskipun terdapat dua nama variable yang sama.

Pemberian nama variable yang sama di dalam blok berbeda akan membingungkan kita sehingga kode program sulit dibaca, sebaiknya jangan gunakan nama variable yang sama pada setiap blok.

```
mysql> DROP PROCEDURE IF EXISTS BLOK_BERULANG3_SP;
CREATE PROCEDURE BLOK_BERULANG3_SP( )
```

```

BEGIN
    DECLARE my_variable varchar(50);
    SET my_variable='pemberian nilai berada diblok utama';
    BEGIN
        DECLARE my_variable VARCHAR(50);
        SET my_variable='pemberian nilai berada diblok lebih dalam';
    END;
    SELECT my_variable, ' tidak melihat perubahan di blok lebih
dalam';
END;

Query OK, 0 rows affected
Query OK, 0 rows affected

mysql> CALL BLOK_BERULANG3_SP();
+-----+
-----+
| my variable | tidak melihat perubahan di
blok lebih dalam |
+-----+
-----+
| pemberian nilai berada diblok utama | tidak melihat perubahan di
blok lebih dalam |
+-----+
-----+
1 row in set

Query OK, 0 rows affected

```

Penggunaan label blok dan pernyataan LEAVE untuk menghentikan eksekusi blok. Pernyataan LEAVE akan dibahas lebih lanjut pada halaman berikutnya, tapi untuk saat ini inti dari pernyataan LEAVE adalah untuk menghentikan eksekusi blok kapan pun.

```

mysql> DROP PROCEDURE IF EXISTS BLOK_BERULANG4_SP;
CREATE PROCEDURE BLOK_BERULANG4_SP()
blok_luar: BEGIN
    DECLARE l_status int;
    SET l_status=1;
    blok_dalam: BEGIN
        IF (l_status=1) THEN
            LEAVE blok_dalam;
        END IF;
        SELECT 'Pernyataan ini tidak pernah di-eksekusi';
    END blok_dalam;
    SELECT 'Akhir program';
END blok_luar;

Query OK, 0 rows affected
Query OK, 0 rows affected

```

```
mysql> CALL BLOK_BERULANG4 SP();
+-----+
| Akhir program |
+-----+
| Akhir program |
+-----+
1 row in set (0.00 sec)

Query OK, 0 rows affected (0.00 sec)
```

3.10 Kontrol Bersyarat

Kontrol bersyarat “**kontrol alir**” adalah suatu pernyataan yang mengizinkan Anda untuk melakukan eksekusi terhadap kode-kode program dari beberapa nilai ekspresi.

Catatan: Seperti yang dijelaskan sebelumnya bahwa ekspresi bisa berupa kombinasi dari literal MySQL, variable, operator, dan.

Bahasa program store MySQL mendukung dua pernyataan kontrol syarat yaitu **IF** dan **CASE**. Kedua kontrol bersyarat tersebut biasanya digunakan salah satu karena memiliki kesamaan tujuan.

3.10.1 Pernyataan IF

Pernyataan IF dalam MySQL hampir sama dengan pernyataan IF dalam bahasa pemrograman lainnya, seperti contoh berikut ini:

```
IF ekspresi THEN perintah
    [ELSEIF ekspresi THEN perintah ....]
    [ELSE perintah]
END IF;
```

3.10.1.1 Kombinasi IF-THEN

Pernyataan **IF-THEN** akan menjalankan perintah berikutnya jika kondisi **ekspresi** bernilai TRUE

```
IF ekspresi THEN  
    Perintah/pernyataan  
END IF;
```

Contoh IF-THEN sederhana yang menjalankan satu pernyataan:

```
IF harga_jual > 25000 THEN  
    CALL buku_discount(kode_buku, 10);  
END IF;
```

Contoh pernyataan **IF-THEN** yang menjalankan lebih dari satu pernyataan jika IF bernilai benar:

```
IF harga_jual > 25000 THEN  
    CALL buku_discount(kode_buku, 10);  
    SELECT * FROM buku_tb WHERE kode_buku = '9789792075830';  
END IF;
```

Contoh pernyataan **IF-THEN** berulang:

```
IF harga_jual > 25000 THEN  
    CALL buku_discount(kode_buku, 10);  
    IF harga_jual > 35000 THEN  
        CALL buku_discount(kode_buku, 5);  
    END IF;  
END IF;
```

3.10.1.2 Pernyataan IF-THEN-ELSE

Penambahan kondisi ELSE pada pernyataan IF mengizinkan Anda untuk menentukan pernyataan baru jika eksekusi kondisi IF bernilai tidak TRUE (tidak TRUE tidak selalu bernilai FALSE, bisa jadi bernilai NULL).

Blok sintaks **IF-THEN-ELSE** adalah:

```
IF ekspresi THEN
    Pernyataan dilaksanakan jika ekspresi bernilai TRUE
ELSE
    Pernyataan dilaksanakan jika ekspresi bernilai FALSE atau NULL
END IF;
```

Contoh penggunaan **IF-THEN-ELSE** sederhana:

```
IF harga_jual < 25000 THEN
    CALL buku_discount(kode_buku, 10);
ELSE
    CALL buku_discount(kode_buku, 15);
END IF;
```

3.10.1.3 Pernyataan IF-THEN-ELSEIF-ELSE

Sintaks pernyataan IF yang lengkap mengizinkan Anda untuk menetapkan kondisi lebih dari satu. Kondisi pertama dijalankan jika hasil evaluasi bernilai TRUE. Jika tidak bernilai TRUE (False, Null), maka klausula ELSE (jika ada) akan dijalankan. Contoh blok **IF-THEN-ELSEIF-ELSE** adalah sebagai berikut (kondisi **ELSEIF** dapat ditambah sesuai kebutuhan):

```
IF ekspresi THEN
    Pernyataan dilaksanakan jika ekspresi bernilai TRUE
ELSEIF ekspresi1 THEN
    Pernyataan dilaksanakan jika ekspresi1 bernilai TRUE
ELSE
    Pernyataan dilaksanakan jika seluruh ekspresi sebelumnya
bernilai FALSE atau NULL
END IF;
```

Contoh blok **IF-ELSEIF** :

```
IF (harga_jual > 45000) THEN
    CALL buku_discount(kode_buku, 10);
ELSEIF (harga_jual>45000 AND kode_buku='9789792075830') THEN
    CALL buku_discount(kode_buku, 15);
    SELECT * FROM buku_tb WHERE kode_buku='9789792075830'
END IF;
```

Contoh blok **IF-ELSEIF** dengan banyak kondisi:

```
IF (harga_jual > 45000) THEN
    CALL buku_discount(kode_buku, 10);
```

```

ELSEIF (harga_jual > 45000 AND kode_buku='9789792075830') THEN
    CALL buku_discount(kode_buku, 15);
    SELECT * FROM buku_tb WHERE kode_buku='9789792075830'
ELSEIF (harga_jual > 45000 AND kode_buku='9789792405569') THEN
    CALL buku_discount(kode_buku, 15);
    SELECT * FROM buku_tb WHERE kode_buku='9789792405569'
ELSEIF (harga_jual > 45000 AND kode_buku='9789793338125') THEN
    CALL buku_discount(kode_buku, 15);
    SELECT * FROM buku_tb WHERE kode_buku='9789793338125'
END IF;

```

3.10.2 Pernyataan CASE

Pernyataan CASE merupakan alternatif dari pernyataan kondisi IF, namun CASE lebih mudah dibaca dan efisien ketika banyak kondisi membutuhkan evaluasi (pengecekan) terutama ketika semua kondisi membandingkan suatu nilai yang berasal dari satu ekspresi.

3.10.2.1 Pernyataan CASE Sederhana

Pernyataan CASE mempunyai dua bentuk. Pertama, disebut sebagai pernyataan **CASE sederhana** yang membandingkan nilai ekspresi dengan banyak kondisi:

```

CASE ekspresi
    WHEN nilai THEN
        pernyataan
    [WHEN nilai THEN
        pernyataan ...]
    [ELSE
        pernyataan]
END CASE;

```

Contoh, kita akan melaksanakan pernyataan INSERT jika ekspresi pilihan=’INSERT’, melaksanakan pernyataan UPDATE jika ekspresi pilihan=’UPDATE’ dan melaksanakan pernyataan DELETE jika ekspresi pilihan=’DELETE’.

```

CASE pilihan
    WHEN 'INSERT' THEN

```

```

    INSERT INTO penulis_tb (kode_penulis, nama_penulis, email,
website, tentang_penulis)
    VALUES (vkode_penulis, vnama_penulis, vemail, vwebsite,
vtentang_penulis);
    WHEN 'UPDATE' THEN
        UPDATE penulis_tb
        SET nama_penulis=vnama_penulis, email=vemail, website=vwebsite,
tentang_penulis=vtentang_penulis
        WHERE kode_penulis = vkode_penulis;
    WHEN 'DELETE' THEN
        DELETE FROM penulis_tb
        WHERE kode_penulis = vkode_penulis;
END CASE;

```

Sama dengan perintah IF Anda dapat menetapkan banyak pernyataan **WHEN**. Anda dapat menambahkan klausa **ELSE** yang akan dilaksanakan pernyataan-pernyataannya jika tidak ada satu ekspresi dan nilai kondisi yang memenuhi syarat.

Catatan: Pernyataan CASE akan menampilkan pesan kesalahan jika tak satu pun dari kondisi yang ada memenuhi syarat. Alangkah baiknya jika pernyataan **ELSE** dicantumkan saat kondisi tidak terpenuhi. Adapun pesan kesalahannya adalah: “**ERROR 1339 (20000) : Case not found for CASE statement”**

3.10.2.2 Pernyataan CASE Dengan Kondisi

Pernyataan **CASE** dengan kondisi secara fungsional serupa dengan **blok IF-ELSEIF-ELSE-END IF**. Sintaks CASE kondisi:

```

CASE
    WHEN kondisi THEN
        pernyataan
    [WHEN kondisi THEN
        pernyataan...]
    [ELSE
        pernyataan]
END CASE;

```

Contoh CASE dengan kondisi:

```

CASE
    WHEN (harga_jual > 45000 AND kode_buku='9789792075830') THEN
        SELECT * FROM buku_tb WHERE kode_buku='9789792075830';
    WHEN (harga_jual > 45000 AND kode_buku='9789792405569') THEN
        SELECT * FROM buku_tb WHERE kode_buku='9789792405569';
    WHEN (harga_jual > 45000 AND kode_buku='9789793338125') THEN
        SELECT * FROM buku_tb WHERE kode_buku='9789793338125';
END CASE;

```

Untuk mencegah kesalahan ketika kondisi tidak terpenuhi maka tambahkan pernyataan ELSE seperti contoh berikut ini:

```

CASE
    WHEN (harga_jual > 45000 AND kode_buku='9789792075830') THEN
        SELECT * FROM buku_tb WHERE kode_buku='9789792075830';
    WHEN (harga_jual > 45000 AND kode_buku='9789792405569') THEN
        SELECT * FROM buku_tb WHERE kode_buku='9789792405569';
    WHEN (harga_jual > 45000 AND kode_buku='9789793338125') THEN
        SELECT * FROM buku_tb WHERE kode_buku='9789793338125';
    ELSE
        SELECT 'Data tidak ditemukan';
END CASE;

```

Contoh penggunaan CASE berulang dengan lingkup blok penanganan kesalahan “**error 1339**” yang berarti tidak satupun kondisi terpenuhi.

```

BEGIN
    DECLARE not_found INT DEFAULT 0;
    DECLARE CONTINUE HANDLER FOR 1339 SET not_found=1;

    CASE
        WHEN (harga_jual > 45000 AND kode_buku='9789792075830') THEN
            SELECT * FROM buku_tb WHERE kode_buku='9789792075830';
        WHEN (harga_jual > 45000 AND kode_buku='9789792405569') THEN
            SELECT * FROM buku_tb WHERE kode_buku='9789792405569';
        WHEN (harga_jual > 45000 AND kode_buku='9789793338125') THEN
            SELECT * FROM buku_tb WHERE kode_buku='9789793338125';
    END CASE;
END;

```

3.11 Perihal Proses Perulangan

Ada dua alasan utama mengapa program membutuhkan perulangan:

- Banyak algoritma matematika yang dapat diterapkan melalui perulangan program.
- Program database terkadang membutuhkan perulangan untuk menelusuri baris/record yang dihasilkan dari pernyataan SELECT, kemudian melakukan manipulasi terhadapnya.

3.11.1 Pernyataan LOOP

Bentuk dari pernyataan LOOP adalah sebagai berikut:

```
[label:] LOOP
  pernyataan
END LOOP [label];
```

Pernyataan-pernyataan yang berada diantara **LOOP** dan **END LOOP** akan diulang secara terus-menerus sampai LOOP dihentikan. Penghentian LOOP melalui pernyataan **LEAVE**.

Anda dapat memberi **label** (penamaan) loop pada bagian awal dan akhir blok. Label akan membantu Anda mengenali akhir dari pernyataan END LOOP yang berarti akhir dari suatu blok perulangan, selain itu label LOOP juga dapat digunakan untuk mengendalikan aliran eksekusi.

Contoh di bawah ini merupakan perulangan sederhana tapi berbahaya, karena perulangan akan dilakukan secara terus-menerus tidak pernah berhenti atau sampai Anda atur penghentinya. Perulangan yang tidak pernah berhenti menyebabkan sumberdaya CPU terkuras dan komputer menjadi *hang*.

```
perluangan_tak_terhingga: LOOP
  SELECT 'Pernyataan ini tak ada habisnya!!!';
END LOOP perluangan_tak_terhingga;
```

3.11.2 Pernyataan LEAVE

Pernyataan **LEAVE** bertujuan untuk menghentikan perulangan yang sedang aktif saat ini. Pernyataan LEAVE tampak seperti berikut:

```
LEAVE label;
```

Contoh ini akan menjalankan pernyataan **LEAVE** saat variable **i** bernilai 10:

```
SET i=1;
myloop: LOOP
    SET i=i+1;
    IF i=10 then
        LEAVE myloop;
    END IF;
END LOOP myloop;

SELECT 'Saya menghitung sampai 10';
```

3.11.3 Pernyataan ITERATE

Pernyataan **ITERATE** digunakan untuk mengulangi kembali eksekusi mulai dari awal perulangan, tanpa melakukan eksekusi pernyataan-pernyataan yang tersisa di dalam perulangan. Adapun sintaks **ITERATE** adalah sebagai berikut ini:

```
ITERATE label;
```

Ketika MySQL menjumpai pernyataan **ITERATE**, maka MySQL mengeksekusi kembali dari awal perulangan/loop yang telah ditunjuk. Contoh berikut ini akan mencetak seluruh angka ganjil kurang dari 10. **ITERATE** digunakan untuk mengulangi loop jika angka yang diperoleh bukan ganjil. **LEAVE** digunakan untuk menghentikan loop ketika variable **i** sudah mencapai angka lebih dari sama dengan 10.

```
SET i=0;
loop1: LOOP
    SET i=i+1;
    IF i>=10 THEN
```

```
    LEAVE loop1;
ELSEIF MOD(i,2)=0 THEN
    ITERATE loop1;
END IF;

    SELECT CONCAT(i," adalah bilangan ganjil");
END LOOP loop1;
```

3.11.4 Perulangan REPEAT..UNTIL

Pernyataan REPEAT dan UNTIL digunakan untuk menciptakan perulangan secara berkelanjutan sampai suatu logika kondisi ditemukan. Sintaks REPEAT...UNTIL adalah seperti berikut ini:

```
[label:] REPEAT
    pernyataan
UNTIL ekspresi END REPEAT [label]
```

Perulangan dengan REPEAT akan terus dilakukan sampai ekspresi yang telah ditetapkan dalam klausa UNTIL bernilai TRUE (pengecekan dilakukan di akhir baris perulangan sehingga minimal satu kali proses eksekusi akan dilakukan). Intinya sama dengan logika blok LOOP-LEAVE-END LOOP, seperti berikut ini:

```
some_label:LOOP
    pernyataan
    IF ekspresi THEN LEAVE some_label; END IF;
END LOOP;
```

Perulangan REPEAT sedikit lebih mudah untuk ditangani karena lebih jelas, mana kondisi-kondisi yang menyebabkan perulangan berhenti. Kalau pernyataan LEAVE bisa diletakkan dimana pun pada bagian loop sedang pernyataan UNTIL selalu dihubungkan dengan klausa END REPEAT pada posisi terakhir dari loop.

Kita tidak harus menyertakan **label** pada perulangan REPEAT, selama kondisi UNTIL tetap dicantumkan pada loop yang aktif saat ini. Penggunaan label dalam perulangan REPEAT disarankan karena akan

mempermudah pembacaan kode program khususnya jika ada perulangan bersarang.

Berikut ini contoh REPEAT yang akan mencetak bilangan ganjil yang kurang dari 10. Bandingkan sintaksnya dengan contoh LOOP sebelumnya yang mengandung pernyataan LEAVE.

```
SET i=0;
loop1: REPEAT
    SET i=i+1;
    IF MOD(i,2)<>0 THEN
        Select concat(i," adalah bilangan ganjil");
    END IF;
UNTIL i >= 10 → Pengecekan berada diakhir
END REPEAT;
```

Catatan penting mengenai perulangan REPEAT:

- Perulangan REPEAT pasti menjalankan setidaknya satu kali eksekusi sampai kondisi UNTIL untuk kali pertama dievaluasi.
- Penggunaan ITERATE dalam perulangan REPEAT bisa menghasilkan sesuatu yang tidak diharapkan sehingga tidak disarankan penggunaannya dalam perulangan REPEAT.

3.11.5 Perulangan WHILE

Perulangan WHILE melakukan eksekusi selama kondisi bernilai TRUE. Jika kondisi awal bukan TRUE maka perulangan tidak akan pernah dilaksanakan, tidak seperti perulangan REPEAT yang akan melaksanakan proses minimal satu kali. Penulisan perulangan WHILE seperti berikut ini:

```
[label:] WHILE ekspresi DO
    pernyataan
END WHILE [label]
```

Secara fungsional perulangan WHILE serupa dengan bentuk **LOOP-LEAVE-END** yang mengandung klausa LEAVE pada baris pertama pernyataan.

```
myloop: LOOP
    IF ekspresi THEN LEAVE myloop; END IF;
    Pernyataan lainnya;
END LOOP myloop;
```

Berikut ini contoh penggunaan WHILE yang menampilkan angka ganjil kurang dari 10:

```
SET i=1;
loop1: WHILE i<=10 DO → Pengecekan berada diawal
    IF MOD(i,2)<>0 THEN
        SELECT CONCAT(i," adalah bilangan ganjil");
    END IF;
    SET i=i+1;
END WHILE loop1;
```

3.11.6 Perulangan Bersarang

Berikut ini contoh bagaimana membuat *time table* dengan cara perulangan di dalam perulangan lain atau sering disebut dengan perulangan bersarang.

```
DECLARE i,j INT DEFAULT 1;
loop_luar: LOOP
    SET j=1;
    loop_dalam: LOOP
        SELECT concat(i," times ", j," is ",i*j);
        SET j=j+1;
        IF j>12 THEN
            LEAVE loop_dalam;
        END IF;
    END LOOP loop_dalam;
    SET i=i+1;
    IF i>12 THEN
        LEAVE loop_luar;
    END IF;
END LOOP loop_luar;
```

3.12 Pernyataan SQL Dalam Program Store

Kita akan membahas penggunaan SQL dalam program store yang sebagian besar berhubungan dengan database.

3.12.1 SQL Selain SELECT

Beberapa pernyataan SQL selain SELECT adalah INSERT, UPDATE, DELETE, SET dan lain-lain. Dalam *program stored MySQL* (store procedure, store function) pernyataan-pernyataan tersebut memiliki konteks yang sama jika dijalankan pada PHP, Visual Basic 2005 atau MySQL *Command Line*. Pernyataan SQL dalam program stored mengikuti aturan penulisan yang sama dengan SQL yang berada di luar program store. Anda dapat menggunakan semua kategori umum dari pernyataan-pernyataan SQL tanpa pembatasan, seperti **DML** (*Data Modelling Language*) dan **DDL** (*Data Definition Language*).

Berikut ini contoh pernyataan SQL hasil kombinasi dari **DDL** dan **DML** bertujuan untuk membuat dan memanipulasi data dalam table.

```
CREATE PROCEDURE CONTOH_SQL_SP();
BEGIN
    DECLARE i INT DEFAULT 1;

    DROP TABLE IF EXISTS `kelompok_buku_tb`;
    CREATE TABLE `kelompok_buku_tb` (
        `kode_kelompok` char(10) NOT NULL DEFAULT '',
        `nama_kelompok` varchar(30) NOT NULL,
        `keterangan` tinytext,
        PRIMARY KEY (`kode_kelompok`)
    ) ENGINE=InnoDB DEFAULT CHARSET=latin1;

    INSERT INTO `kelompok_buku_tb` VALUES ('0000000002', 'Novel',
    null);
    INSERT INTO `kelompok_buku_tb` VALUES ('0000000003', 'Agama',
    null);
    INSERT INTO `kelompok_buku_tb` VALUES ('0000000004', 'Komik',
    null);

    DELETE FROM kelompok_tb
    WHERE kode_kelompok='0000000001';
END;
```

3.12.2 SELECT Dengan Klausa INTO

Pernyataan **SELECT INTO** bertujuan untuk mengembalikan satu baris data saja ke dalam variable program stored. Bentuk pernyataan **SELECT INTO** adalah:

```
SELECT ekspresi1 [, ekspresi2 ...]
INTO variable1 [, variable2 ...]
    klausa pernyataan SELECT lainnya
```

Berikut ini contoh untuk mendapatkan detail data table **buku_tb**, dimana kolom **kode_buku** dilewatkan sebagai parameter.

```
DROP PROCEDURE IF EXISTS MENDAPATKAN_DETIL_BUKU;
CREATE PROCEDURE MENDAPATKAN_DETIL_BUKU (in_kode_buku CHAR(10))
BEGIN
    DECLARE v_kode_buku CHAR(10);
    DECLARE v_judul TINYTEXT;
    DECLARE v_harga DOUBLE;
    SELECT kode_buku, judul, harga_jual
    INTO v_kode_buku, v_judul, v_harga
    FROM buku_tb
    WHERE kode_buku = in_kode_buku;
END;
```

Parameter **kode_buku**

Variable program stored

Pelimpahan kolom table ke variable program stored

Jika klausa **WHERE** dihilangkan maka akan dihasilkan baris data lebih dari satu yang menyebabkan kesalahan ketika store procedure dijalankan.

```
mysql> CALL MENDAPATKAN_DETIL_BUKU('9789792075830');
1172 - Result consisted of more than one row
```

3.12.3 Membuat Dan Menggunakan Cursor

Cursor adalah object untuk menangani pernyataan **SELECT** yang mengembalikan baris data lebih dari satu secara terprogram. Cursor melakukan penelusuran secara berkelanjutan mulai dari awal sampai akhir terhadap rangkaian hasil baris data dan mengambil tindakan tertentu pada setiap baris data yang ditemui.

3.12.3.1 Deklarasi Cursor

Pendeklarasian cursor dilakukan melalui pernyataan **DECLARE** seperti penulisan berikut ini:

```
DECLARE nama_cursor CURSOR FOR pernyataan_SELECT;
```

Deklarasi cursor harus diletakkan setelah semua pendeklarasian variable. Mendeklarasikan cursor sebelum pendeklarasian variable akan menyebabkan kesalahan dengan pesan **error 1337**.

```
mysql> DROP PROCEDURE IF EXISTS CURSOR_BURUK;  
CREATE PROCEDURE CURSOR_BURUK()
```

```
BEGIN
```

```
    DECLARE c CURSOR FOR SELECT * from buku_tb;
```

```
    DECLARE i INT;
```

```
END;
```

```
Query OK, 0 rows affected
```

Pernyataan DECLARE

seharusnya diletakkan di sini

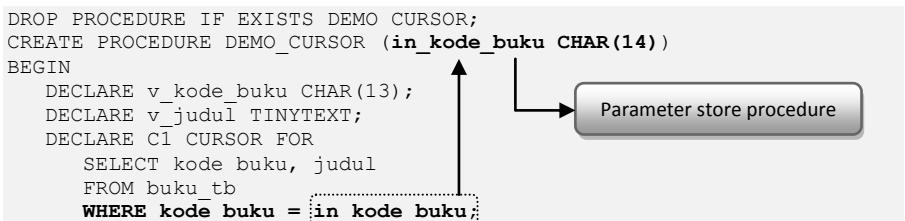
```
1337 - Variable or condition declaration after cursor or handler  
declaration
```

Cursor selalu dihubungkan dengan pernyataan SELECT, contoh berikut ini menunjukkan sebuah deklarasi cursor sederhana yang akan menelusuri kolom tertentu dari table **buku_tb**.

```
DECLARE cursor1 CURSOR FOR  
    SELECT kode_buku, judul FROM customers;
```

Sebuah cursor dapat dipengaruhi oleh variable parameter program stored melalui penggunaan klausa WHERE. Contoh di bawah ini adalah cursor yang mengandung parameter stored procedure, diletakkan pada klausa WHERE. Ketika cursor dibuka, parameter akan menjadi penentu baris data mana yang akan dikembalikan.

```
DROP PROCEDURE IF EXISTS DEMO_CURSOR;
CREATE PROCEDURE DEMO_CURSOR (in_kode_buku CHAR(14))
BEGIN
    DECLARE v_kode_buku CHAR(13);
    DECLARE v_judul TINYTEXT;
    DECLARE C1 CURSOR FOR
        SELECT kode_buku, judul
        FROM buku_tb
        WHERE kode_buku = in_kode_buku
```



3.12.3.2 Pernyataan-pernyataan Cursor

Bahasa program store MySQL mendukung tiga pernyataan yang berhubungan dengan operasi cursor:

- **OPEN**

Untuk membuka cursor sebelum mengambil baris data apapun dari cursor tersebut.

```
OPEN nama_cursor;
```

- **FETCH**

Penelusuran baris data berikutnya dari cursor dan memindahkan penunjuk cursor (*pointer*) di dalam *result set* (hasil berupa sekumpulan baris data).

```
FETCH nama_cursor INTO daftar_variable;
```

daftar_variable adalah variable yang memiliki kesamaan tipe dengan tipe kolom yang diperoleh dari pernyataan SELECT deklarasi cursor dan menjadi tempat tampungan nilai data kolom (*result set*).

- **CLOSE**

Untuk mematikan dan membebaskan memori yang telah dipakai oleh cursor.

```
CLOSE cursor_name;
```

3.12.3.3 Mengambil Satu Baris Cursor

Secara mendasar proses pengambilan satu baris cursor adalah: membuka cursor, mengambil satu baris data, dan menutup *result set*. Secara logika serupa dengan pernyataan SELECT disertai klausa INTO.

```
DROP PROCEDURE IF EXISTS DEMO_CURSOR;
CREATE PROCEDURE DEMO_CURSOR()
BEGIN
    DECLARE v_kode_buku CHAR(13);
    DECLARE v_judul TINYTEXT;
    DECLARE cursor1 CURSOR FOR
        SELECT kode_buku, judul
        FROM buku tb;

    OPEN cursor1;
    FETCH cursor1 INTO v_kode_buku, v_judul;
    CLOSE cursor1;
END;
```

3.12.3.4 Mengambil Keseluruhan *Result Set*

Pemrosesan cursor yang paling sering digunakan adalah mengambil setiap baris yang diperoleh dari pernyataan SELECT cursor, melakukan satu atau lebih operasi terhadap data yang telah ditelusuri, kemudian menutup cursor setelah mencapai baris terakhir. Contoh berikut ini menunjukkan bagaimana mendeklarasikan dan membuka cursor, mengambil baris-baris data dari cursor dengan cara perulangan dan menutup cursor.

```
DROP PROCEDURE IF EXISTS DEMO_CURSOR;
CREATE PROCEDURE DEMO_CURSOR()
BEGIN
    DECLARE v_kode_buku CHAR(13);
    DECLARE v_judul TINYTEXT;
    DECLARE cursor1 CURSOR FOR
        SELECT kode_buku, judul
        FROM buku tb;

    OPEN cursor1;
    label_cursor: LOOP
        FETCH cursor1 INTO v_kode_buku, v_judul;
    END LOOP label_cursor;
    CLOSE cursor1;
```

```
END;
```

Kode program di atas kelihatannya benar dan lengkap tetapi sebenarnya masih ada masalah. Jika penelusuran telah mencapai pada baris terakhir pengambilan, maka MySQL akan menampilkan pesan kesalahan "**no data to fetch**" (MySQL error 1329; SQLSTATE 02000) seperti berikut ini:

```
mysql> CALL DEMO_CURSOR();
1329 - No data - zero rows fetched, selected, or processed
```

Cara mengatasinya adalah melalui pendeklarasian *error handler* (penanganan kesalahan). *Error handler* akan menangkap kesalahan "**no data to fetch**" sehingga perulangan yang sedang terjadi dapat dihentikan dan cursor ditutup. Penambahan kode program *error handler* menjadi seperti berikut ini:

```
DECLARE CONTINUE HANDLER FOR NOT FOUND SET baris_akhir=1;
```

Penanganan kesalahan MySQL melakukan dua hal saat terjadi "**no data to fetch**":

1. Memberi nilai pada variable **baris_akhir** dengan 1 jika *pointer* (penunjuk cursor) sudah mencapai baris akhir *result set* yang diperoleh dari perintah SELECT cursor.
2. Mengizinkan program untuk melanjutkan eksekusi jika belum mencapai baris akhir *result set*.

Program sekarang dapat mengecek nilai dari variable **baris_akhir**. Jika bernilai 1 berarti baris terakhir telah diambil dan perulangan harus dihentikan dan cursor ditutup. Kita harus melakukan penataan ulang (*reset*) terhadap penunjuk baris akhir *result set* setelah cursor ditutup dengan cara memberi nilai 0 pada variable **baris_akhir**. Contoh berikut ini menunjukkan langkah-langkahnya, yaitu mendeklarasikan *handler CONTINUE*, melakukan perulangan baris *result set*, meninggalkan

perulangan jika variable **baris_akhir** telah diberi nilai, kemudian membersihkannya.

```
DROP PROCEDURE IF EXISTS DEMO_CURSOR;
CREATE PROCEDURE DEMO_CURSOR()
BEGIN
    DECLARE v_kode_buku CHAR(13);
    DECLARE v_judul TINYTEXT;
    DECLARE baris_akhir TINYTEXT;
    DECLARE cursor1 CURSOR FOR
        SELECT kode_buku, judul
        FROM buku_tb;
    DECLARE CONTINUE HANDLER FOR NOT FOUND SET baris_akhir=1;
    SET baris_akhir=0;

    OPEN cursor1;
    label_cursor: LOOP
        FETCH cursor1 INTO v_kode_buku, v_judul;
        IF baris_akhir=1 THEN
            LEAVE label_cursor;
        END IF;
        /* Lakukan sesuatu disini */
    END LOOP label_cursor;
    CLOSE cursor1;
    SET baris_akhir=0;
END;
```

3.12.3.5 Jenis Perulangan Cursor

Masih ada dua jenis perulangan selain **LOOP END LOOP** yaitu perulangan **WHILE** dan **REPEAT UNTIL**. Berikut ini contoh cursor dengan perulangan **WHILE**:

```
DROP PROCEDURE IF EXISTS DEMO_CURSOR WHILE;
CREATE PROCEDURE DEMO_CURSOR WHILE()
BEGIN
    DECLARE v_kode_buku CHAR(13);
    DECLARE v_judul TINYTEXT;
    DECLARE baris_akhir TINYTEXT;
    DECLARE cursor1 CURSOR FOR
        SELECT kode_buku, judul
        FROM buku_tb;
    DECLARE CONTINUE HANDLER FOR NOT FOUND SET baris_akhir=1;
    SET baris_akhir=0;

    OPEN cursor1;
```

```

label_cursor: WHILE (baris_akhir=0) DO
    FETCH cursor1 INTO v_kode_buku, v_judul;
    /* Lakukan sesuatu disini*/
    IF baris_akhir=1 THEN
        LEAVE label_cursor;
    END IF;
END WHILE label_cursor;
CLOSE cursor1;
SET baris_akhir=0;
END;

```

Contoh cursor dengan perulangan **REPEAT UNTIL**:

```

DROP PROCEDURE IF EXISTS DEMO_CURSOR_REPEAT;
CREATE PROCEDURE DEMO_CURSOR_REPEAT()
BEGIN
    DECLARE v_kode_buku CHAR(13);
    DECLARE v_judul TINYTEXT;
    DECLARE baris_akhir TINYTEXT;
    DECLARE cursor1 CURSOR FOR
        SELECT kode_buku, judul
        FROM buku tb;
    DECLARE CONTINUE HANDLER FOR NOT FOUND SET baris_akhir=1;
    SET baris_akhir=0;

    OPEN cursor1;
    label_cursor: REPEAT
        FETCH cursor1 INTO v_kode_buku, v_judul;
        /* Lakukan sesuatu disini*/
        IF baris_akhir THEN
            LEAVE label_cursor;
        END IF;
    UNTIL baris_akhir END REPEAT label_cursor;
    CLOSE cursor1;
    SET baris_akhir=0;
END;

```

3.4 Store Function

Store function hampir sama dengan store procedure yaitu potongan kode yang berisi perintah/pernyataan SQL yang disimpan di dalam katalog, dan dapat dipanggil dari aplikasi atau pernyataan SQL lainnya. Adapun perbedaannya antara lain:

- Store function hanya memiliki parameter input dan tidak memiliki parameter output karena store function sendiri adalah parameter output.
- Setelah store function dibuat maka store function dapat dioperasikan dengan semua jenis ekspresi. Oleh karena itu store function tidak dapat dipanggil melalui pernyataan **CALL**.
- Stored function harus mengandung pernyataan **RETURNS**. RETURNS merupakan pernyataan SQL khusus yang tidak dapat digunakan oleh store procedure. RETURN bertujuan untuk mengembalikan nilai ke store function itu sendiri.

Aturan penulisan CREATE FUNCTION hampir sama dengan store procedure, yaitu diawali nama function, diikuti oleh parameter dan diakhiri oleh badan function yang berisi kode. Store function hanya mempunyai parameter input sehingga parameter tidak perlu ditetapkan sebagai IN, OUT, dan INOUT karena secara tak tampak semua parameter dianggap sebagai parameter input. Pernyataan **RETURNS** diikuti oleh tipe data yang mengindikasikan jenis data yang akan dikembalikan ke store function. Definisi store function mengikuti aturan penulisan seperti berikut ini:

```
CREATE FUNCTION nama_function (parameter[,...])
RETURNS tipedata
[LANGUAGE SQL]
[ [NOT] DETERMINISTIC ]
[ {CONTAINS SQL | NO SQL | MODIFIES SQL DATA | READS SQL DATA} ]
[ SQL SECURITY {DEFINER|INVOKER} ]
[ COMMENT string_komentar ]
pernyataan_function
```

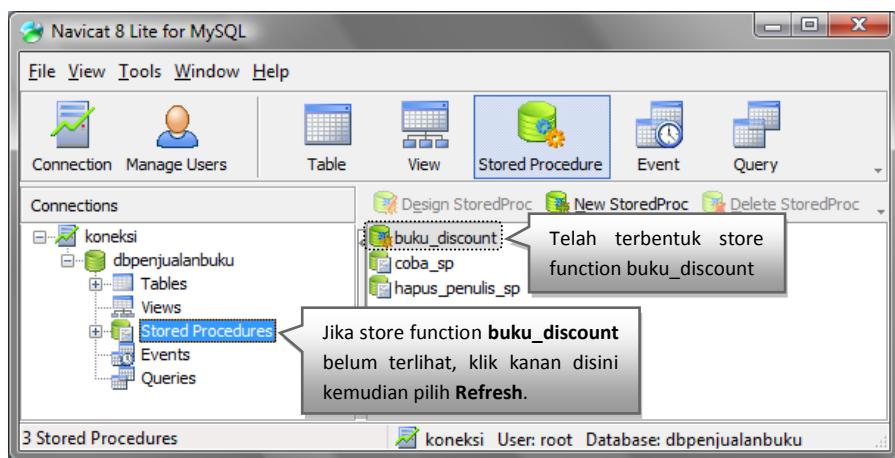
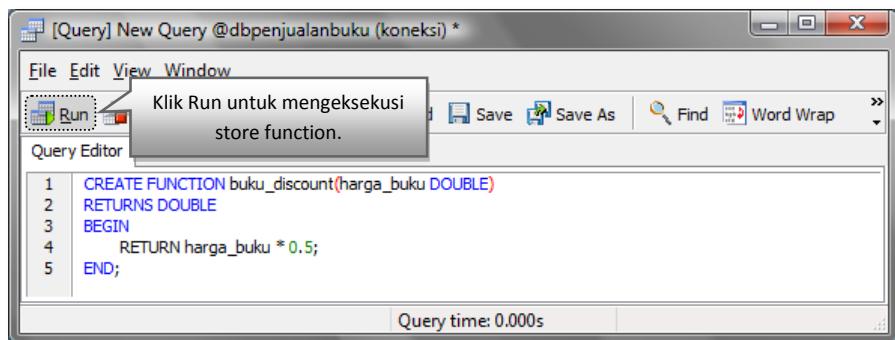
3.4.1 Contoh Store Function

Contoh store function yang memberi potongan harga sebesar 5 persen (0.5) dari harga jual buku (kolom **harga_jual**).

```

CREATE FUNCTION buku_discount(harga_buku DOUBLE)
RETURN DOUBLE
BEGIN
    RETURN harga_buku * 0.5;
END;

```

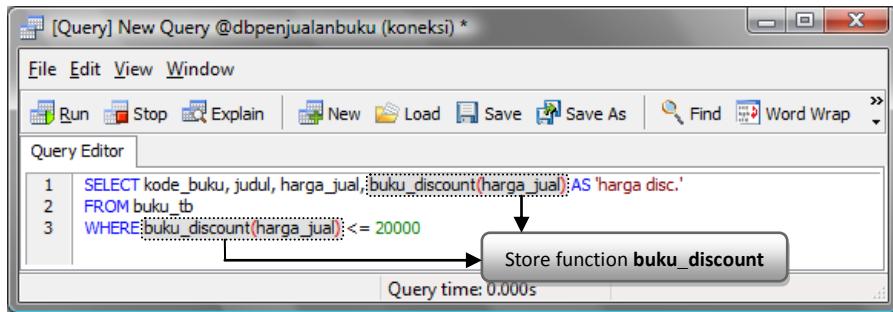


Contoh pemanggilan store function **buku_discount**.

```

SELECT kode_buku, judul, harga_jual, buku_discount(harga_jual) AS
'harga disc.'
FROM buku_tb
WHERE buku_discount(harga_jual) <= 20000

```



The screenshot shows the MySQL Workbench interface with a result grid window titled "[Query] New Query @dbpenjualanbuku (koneksi) *". The result grid displays a table with columns: kode_buku, judul, harga_jual, and harga disc. A row for the book "Internet Marketing" is highlighted.

kode_buku	judul	harga_jual	harga disc.
9789791167123	Buku Pintar Pemrograman PHP	29500	14750
9789792075830	Aplikasi Manajemen Perekutran Berbasis Access 97/2000/XP	35000	17500
9789792405569	Tips & Trik Baru Menembus Tes Bakat Skolastik	25000	12500
9789793767079	Efek Partikel 3ds max 6	35000	17500
9789793767543	7 Jam Belajar Visual Basic .NET Untuk Orang Awam	40000	20000
9789795336730	Internet Marketing	25000	12500
9789797633134	Kumpulan Latihan Pemrograman Delphi	25000	12500
97897999992260	30 Menit Menjadi Webmaster	29500	14750

A callout bubble points to the value "12500" in the "harga disc." column of the highlighted row with the text "Harga discount <= 20000". Below the grid, it says "SELECT kode_buku, judul, harga_jual, buku.discount(harga_jual) AS 'harga disc.'". At the bottom, it says "Query time: 0.003s" and "Record 6 of 8".

Dapat dilihat bahwa store function `buku_discount()` dapat dipanggil layaknya function *scalar* yang telah disediakan oleh MySQL. Tidak ada perbedaan antara pemanggilan function scalar, seperti **SUBSTR** dan **COS** dengan pemanggilan store function.

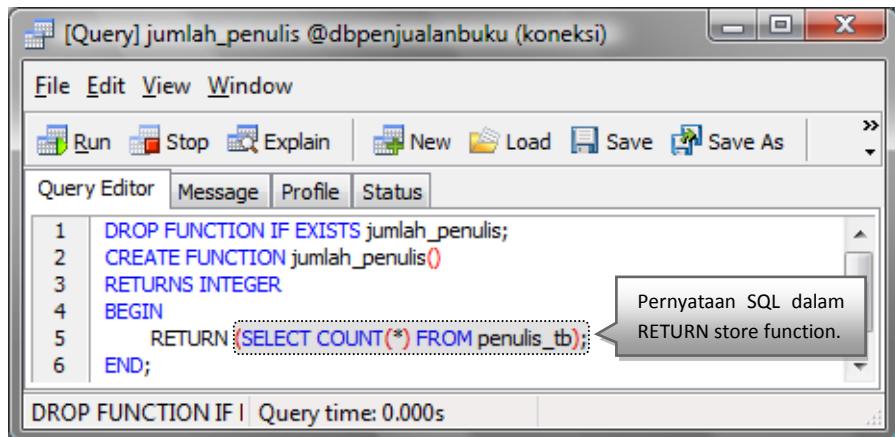
Contoh store function yang mengembalikan data jumlah penulis dari table **penulis_tb**.

```

DROP FUNCTION IF EXISTS jumlah_penulis;
CREATE FUNCTION jumlah_penulis()
RETURNS INTEGER

```

```
BEGIN
    RETURN (SELECT COUNT(*) FROM penulis_tb);
END;
```



Contoh pemanggilan store function **jumlah_penulis** pada fitur console, buka menu **Tools->Console**:

```
mysql> SELECT jumlah_penulis(), jumlah_penulis()+10 as 'Penulis + 10';
+-----+-----+
| jumlah_penulis() | Penulis + 10 |
+-----+-----+
|           15 |          25 |
+-----+-----+
1 row in set
```

Contoh di atas menunjukkan bahwa:

- Pernyataan SQL boleh digunakan dalam store function.
- Pernyataan RETURNS bisa berisi ekspresi gabungan yang rumit.

Contoh store function yang menghitung jumlah hari dari selisih antara dua hari.

```
DROP FUNCTION IF EXISTS NUMBER_OF_DAYS;
CREATE FUNCTION NUMBER_OF_DAYS (START_DATE DATE, END_DATE DATE)
RETURNS INTEGER
```

```

BEGIN
    DECLARE DAYS INTEGER;
    DECLARE NEXT_DATE, PREVIOUS_DATE DATE;

    SET DAYS = 0;
    SET NEXT_DATE = START_DATE + INTERVAL 1 DAY;

    WHILE NEXT_DATE <= END_DATE DO
        SET DAYS = DAYS + 1;
        SET PREVIOUS_DATE = NEXT_DATE;
        SET NEXT_DATE = NEXT_DATE + INTERVAL 1 DAY;
    END WHILE;

    RETURN DAYS;
END;

```

The screenshot shows the MySQL Workbench Query Editor window. The title bar reads "[Query] NUMBER_OF_DAYS @dbpenjualanbuku (koneksi) *". The menu bar includes File, Edit, View, and Window. The toolbar contains Run, Stop, Explain, New, Load, Save, Save As, and Find buttons. The main area is labeled "Query Editor" and contains the following SQL code:

```

1 DROP FUNCTION IF EXISTS NUMBER_OF_DAYS;
2 CREATE FUNCTION NUMBER_OF_DAYS (START_DATE DATE, END_DATE DATE)
3 RETURNS INTEGER
4 BEGIN
5     DECLARE DAYS INTEGER;
6     DECLARE NEXT_DATE, PREVIOUS_DATE DATE;
7
8     SET DAYS = 0;
9     SET NEXT_DATE = START_DATE + INTERVAL 1 DAY;
10
11    WHILE NEXT_DATE <= END_DATE DO
12        SET DAYS = DAYS + 1;
13        SET PREVIOUS_DATE = NEXT_DATE;
14        SET NEXT_DATE = NEXT_DATE + INTERVAL 1 DAY;
15    END WHILE;
16
17    RETURN DAYS;
18 END;

```

Below the code, a message in the status bar says "DROP FUNCTION IF EXISTS NL Query time: 0.000s".

3.4.2 Menghapus Store Function

Pernyataan DROP berlaku pula pada penghapusan store function. Definisi:

```

DROP FUNCTION [ IF EXISTS ]
[ <nama_database> . ] <nama_function >

```

Contoh penghapusan program function:

```
DROP FUNCTION buku_discount
```

Atau

```
DROP FUNCTION IF EXISTS buku_discount
```

BAB 4

Pemrograman Database

4.1 Connection

Anda perlu menetapkan sebuah sesi (*session*) ke database server sebelum melakukan sesuatu apapun terhadap database. Dibutuhkan sebuah object yang disebut *connection*, yaitu berupa *instance class* yang menerapkan antarmuka **System.Data.IDbConnection**, tergantung dari *data provider database* yang digunakan. Berikut ini tabel ringkasan class *connection* yang menerapkan antarmuka **System.Data.IDbConnection**:

Data Provider	Namespace	Connection Class
ODBC	System.Data.Odbc	OdbcConnection
OLE DB	System.Data.OleDb	OleDbConnection
Oracle	System.Data.OracleClient	OracleConnection
SQL Server	System.Data.SqlClient	SqlConnection
MySQL	MySQL.Data.MySQLClient	MySQLConnection

Pembahasan *connection* hanya mengacu pada database MySQL sedangkan untuk *data provider database* lain caranya kurang lebih sama.

4.1.1 Connection MySQLConnection

Cara pembentukan Connection yang menghubungkan antara Visual Basic .NET 2005/2008 dengan database MySQL adalah seperti berikut:

- Buat Project baru bernama **Connection** dengan mengeklik menu **File->New->Project**.
 - Pilih template **Console Application** hingga terbentuk file **Module1.vb**.
-

- Tambahkan *reference MySQL.Data* di menu **Project->Add Reference->.NET->MySQL.Data**.

Connection: Module1.vb

```

Imports MySQL.Data.MySQLClient

Module Connection

    Sub Main()
        Dim connString, db, user, pwd As String

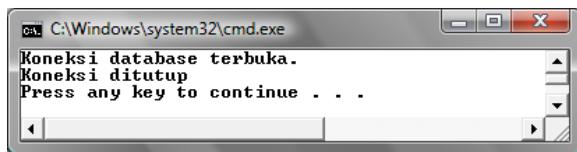
        db = "dbpenjualanbuku" 'database
        user = "root" 'user default
        pwd = "" 'password kosong
        connString = "Database=" & db & ";Data Source=localhost;User
Id=" & user & ";Password=" & pwd

        'membuat connection
        Dim conn As MySqlConnection = New MySqlConnection(connString)

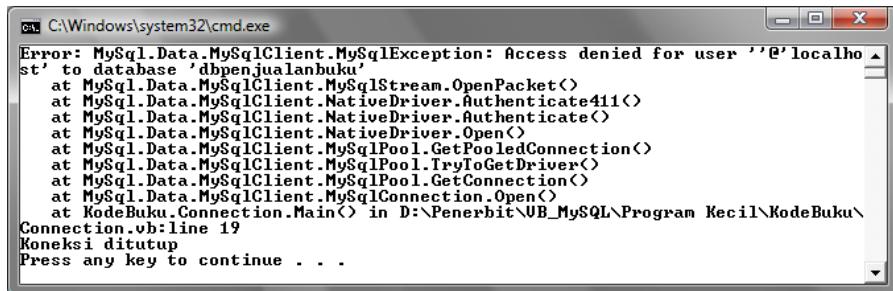
        Try
            'membuka connection
            conn.Open()
            Console.WriteLine("Koneksi database terbuka.")
        Catch ex As Exception
            'menampilkan pesan kesalahan
            Console.WriteLine("Error: " & ex.ToString)
        Finally
            'menutup connection
            conn.Close()
            Console.WriteLine("Koneksi ditutup")
        End Try
    End Sub
End Module

```

Sekarang jalankan program **Connection.vb** dengan menekan tombol keyboard **Ctrl+F5**, jika connection sukses maka tampilannya kurang lebih seperti ini:



Sekarang ubah nilai variable **user** yang semula bernilai **root** menjadi **root1**, maka connection akan gagal dan menampilkan pesan seperti berikut:



```
C:\Windows\system32\cmd.exe
Error: MySql.Data.MySqlClient.MySqlException: Access denied for user ''P' localho...
  at MySql.Data.MySqlClient.MySqlStream.OpenPacket()
  at MySql.Data.MySqlClient.NativeDriver.Authenticate411()
  at MySql.Data.MySqlClient.NativeDriver.Authenticate()
  at MySql.Data.MySqlClient.NativeDriver.Open()
  at MySql.Data.MySqlClient.MySqlPool.GetPooledConnection()
  at MySql.Data.MySqlClient.MySqlPool.TryToGetDriver()
  at MySql.Data.MySqlClient.MySqlPool.GetConnection()
  at MySql.Data.MySqlClient.MySqlConnection.Open()
  at KodeBuku.Connection.Main() in D:\Penerbit\VB_MySQL\Program Kecil\KodeBuku\Conne...
Connection.vb:line 19
Koneksi ditutup
Press any key to continue . . .
```

4.1.2 Bagaimana Connection Bekerja

Kita perlu menentukan ADO.NET dan *namespace* data provider untuk MySQL, seperti kode berikut ini:

```
Imports MySQL.Data.MySQLClient
```

Selanjutnya adalah membuat *connection string*. Sebuah connection string terdiri atas beberapa parameter, setiap pasangan **key=value** dipisahkan oleh tanda titik koma, setiap parameter mengandung informasi connection. Pada umumnya aturan penentuan parameter-parameter berlaku sama bagi semua *data provider* namun cara penulisannya saja yang berbeda.

```
Dim connString, db, user, pwd As String

db = "dbpenjualanbuku" 'database
user = "root" 'user default
pwd = "" 'password kosong
connString = "Database=" & db & ";Data Source=localhost;User Id=" &
user & ";Password=" & pwd
```

Parameter di atas terbagi menjadi empat bagian, antara lain: **Database**, **Data Source**, **User Id**, dan **Password**. Penjelasannya adalah sebagai berikut:

- **Database = “dbpenjualanbuku”**, adalah penentuan database mana yang akan dikoneksikan. Disini adalah “**dbpenjualanbuku**” dengan anggapan bahwa database tersebut sudah ada.
- **Data Source = “localhost”**, data source mengacu pada alamat IP dimana database berada. Karena database berada di komputer lokal, maka harus diisi dengan “localhost” atau “127.0.0.1”.
- **User Id = “root”**, adalah penentuan user yang berhak mengakses database “**dbpenjualanbuku**” sehingga dapat membuka koneksi tersebut. Secara default, MySQL yang diinstall melalui XAMPP mempunyai User Id **root**.
- **Password = “”**, adalah kode sandi yang dimiliki oleh user root. Secara default, MySQL yang diinstall melalui XAMPP mempunyai password kosong (tak ber-password) untuk user **root**. Dengan adanya user dan password maka connection menjadi lebih aman, karena tidak semua orang bisa membuat sesi dengan database.

Setelah semua parameter terisi lengkap dan benar, selanjutnya adalah membuat object connection yang dapat melewaskan connection string.

```
'membuat connection
Dim conn As MySqlConnection = New MySqlConnection(connString)
```

Sekarang Anda telah memiliki connection, tetapi masih perlu dibangun sesi (*session*) dengan database melalui pemanggilan method **Open** di dalam connection. Jika pembentukan *session* gagal maka akan dilempar ke *exception* sehingga digunakanlah pernyataan **Try** sebagai cara untuk menangani exception.

```
Try
    'membuka connection
    conn.Open()
    Console.WriteLine("Koneksi database terbuka.")
```

Jika terjadi kegagalan maka exception akan menjalankan pernyataan-pernyataan yang berada diantara pernyataan **catch** serta menampilkan pesan kesalahannya:

```
Catch ex As Exception  
    'menampilkan pesan kesalahan  
    Console.WriteLine("Error: " & ex.ToString)
```

Setiap *data provider* mempunyai class exception-nya sendiri-sendiri. Sebagai contoh, data provider MySQL mempunyai class exception **MySqlException** yang menyediakan informasi tentang *error* database MySQL lebih rinci, sehingga pernyataan **Catch ex As Exception** dapat diganti menjadi **Catch ex As MySqlException**.

Anda harus memanggil pernyataan **Close()** untuk menghentikan *session* dan mencetak pesan yang menunjukkan bahwa **Close()** telah dipanggil. Pernyataan yang berada di bawah **Finally** selalu dilaksanakan meskipun pernyataan di bawah **Try** sukses atau gagal dijalankan.

```
Finally  
    'menutup connection  
    conn.Close()  
    Console.WriteLine("Koneksi ditutup")
```

4.1.3 Menampilkan Informasi Connection

Connection memiliki beberapa *property* yang menyediakan informasi tentang connection. Hampir seluruh properties bersifat *read-only* (hanya dapat dibaca), karena hanya bertujuan menampilkan serangkaian informasi connection. Property tersebut sangat berguna saat proses *debugging* (penelusuran program) untuk memastikan bahwa memang property connection tersebut yang dikehendaki.

Buatlah module baru bernama **ConnetionDisplay.vb**:

- Buat Project baru bernama **ConnectionDisplay** dengan mengeklik menu **File->New->Project**.
- Pilih template **Console Application** hingga terbentuk file **Module1.vb**.
- Tambahkan reference **MySQL.Data** pada menu **Project->Add Reference->.NET->MySQL.Data**.

ConnetionDisplay: Module1.vb

```

Imports MySQL.Data.MySQLClient

Module Module1

    Sub Main()
        Dim connString, db, user, pwd As String

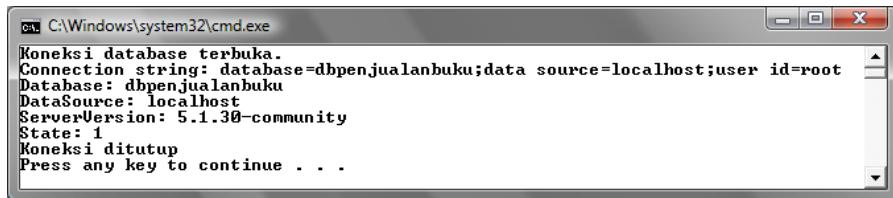
        db = "dbpenjualanbuku" 'database
        user = "root" 'user default
        pwd = "" 'password kosong
        connString = "Database=" & db & ";Data Source=localhost;User
Id=" & user & ";Password=" & pwd

        'membuat connection
        Dim conn As MySqlConnection = New MySqlConnection(connString)

        Try
            'membuka connection
            conn.Open()
            Console.WriteLine("Koneksi database terbuka.")
            Console.WriteLine("Connection string: " &
conn.ConnectionString)
            Console.WriteLine("Database: " & conn.Database)
            Console.WriteLine("DataSource: " & conn.DataSource)
            Console.WriteLine("ServerVersion: " & conn.ServerVersion)
            Console.WriteLine("State: " & conn.State)
        Catch ex As Exception
            'menampilkan pesan kesalahan
            Console.WriteLine("Error: " & ex.ToString)
        Finally
            'menutup connection
            conn.Close()
            Console.WriteLine("Koneksi ditutup")
        End Try
    End Sub
End Module

```

Jalankan programnya dengan menekan tombol keyboard Ctrl+F5. Jika connection sukses maka akan terlihat keluaran seperti berikut ini:



```
C:\Windows\system32\cmd.exe
Koneksi database terbuka.
Connection string: database=dbpenjualanbuku;data source=localhost;user id=root
Database: dbpenjualanbuku
DataSource: localhost
ServerVersion: 5.1.30-community
State: 1
Koneksi ditutup
Press any key to continue . . .
```

Kode program hampir sama dengan contoh sebelumnya yang berbeda adalah:

```
Console.WriteLine("Connection string: " & conn.ConnectionString)
Console.WriteLine("Database: " & conn.Database)
Console.WriteLine("DataSource: " & conn.DataSource)
Console.WriteLine("ServerVersion: " & conn.ServerVersion)
Console.WriteLine("State: " & conn.State)
```

Kode program memberi informasi berkenaan dengan **ConnectionString**, antara lain:

```
Connection string: database=dbpenjualanbuku;data
source=localhost;user id=root
```

Informasi nama database yang digunakan:

```
Database: dbpenjualanbuku
```

Informasi DataSource, dimana database berada:

```
DataSource: localhost
```

Informasi versi database MySQL yang digunakan

```
ServerVersion: 5.1.30-community
```

Informasi status connection, jika bernilai 1 maka connection terbuka, sebaliknya jika bernilai 0 maka artinya connection tertutup:

```
State: 1
```

4.2 Command

Interaksi dengan database dapat dilakukan setelah connection terbentuk misalnya: menambah data, mengubah data, menghapus data, bahkan memodifikasi database sekalipun. Apapun perintah yang diberikan dapat dipastikan akan melibatkan object *Command*.

4.2.1 Membuat Command

Untuk membentuk *command* bisa digunakan method *constructor MySQLCommand* atau menggunakan method lain yang secara tidak langsung akan membentuk object *Command* untuk Anda.

- Buat Project baru bernama **CommandSQL** dengan mengeklik menu **File->New->Project**.
- Pilih template **Console Application** hingga terbentuk file **Module1.vb**.
- Tambahkan reference **MySQL.Data** pada menu **Project->Add Reference->.NET->MySQL.Data**.

CommandSQL: Module1.vb

```
Imports MySQL.Data.MySQLClient

Module Module1

    Sub Main()
        Dim connString, db, user, pwd As String

        db = "dbpenjualanbuku" 'database
        user = "root" 'user default
        pwd = "" 'password kosong
        connString = "Database=" & db & ";Data Source=localhost;User
Id=" & user & ";Password=" & pwd

        'membuat connection
        Dim conn As MySqlConnection = New MySqlConnection(connString)

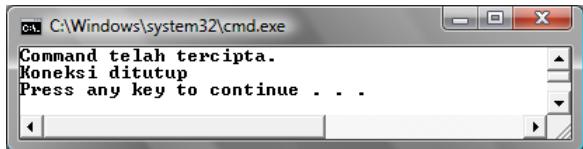
        'membuat command
        Dim CMD As MySqlCommand = New MySqlCommand()
        Console.WriteLine("Command telah tercipta.")
```

```

Try
    'membuka connection
    conn.Open()
Catch ex As Exception
    'menampilkan pesan kesalahan
    Console.WriteLine("Error: " & ex.ToString())
Finally
    'menutup connection
    conn.Close()
    Console.WriteLine("Koneksi ditutup")
End Try
End Sub
End Module

```

Sekarang jalankan programnya dengan menekan tombol keyboard **Ctrl+F5**, tampilannya kurang lebih seperti ini:



Anda telah membuat object **MySQLCommand** melalui default *constructor*, serta mencetak pesan yang menandakan command telah terbentuk.

```

'membuat command
Dim CMD As MySqlCommand = New MySqlCommand()
Console.WriteLine("Command telah tercipta.")

```

Contoh command di atas masih kosong belum dihubungkan dengan connection dan perintah SQL sama sekali, berarti command belum membawa manfaat apa-apa.

4.2.2 Menghubungkan Command dan Connection

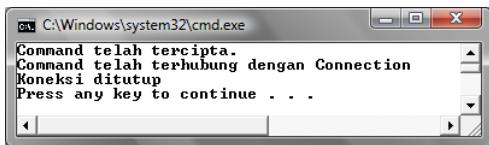
Jika command bekerja memanfaatkan database maka setiap command harus dihubungkan dengan *connection database* melalui pengaturan *property connection command* yang memberi perintah untuk menyimpan sumberdaya yang dibutuhkan. Satu connection dapat dimanfaatkan oleh

lebih dari satu command. Ada beberapa cara untuk menghubungkan command dengan connection, seperti contoh-contoh di bawah ini:

```
Try
    'membuka connection
    conn.Open()

    'menghubungkan command ke connection
    CMD.Connection = conn
    Console.WriteLine("Command telah terhubung dengan Connection")
```

Jalankan programnya dengan menekan tombol keyboard **Ctrl+F5**, seharusnya tampilannya seperti berikut:



Seperti yang Anda lihat pada contoh sebelumnya, kode program diawali dengan pembentukan connection dan command:

```
'membuat connection
Dim conn As MySqlConnection = New MySqlConnection(connString)

'membuat command
Dim CMD As MySqlCommand = New MySqlCommand()
Console.WriteLine("Command telah tercipta.")
```

Pada fase tersebut connection dan command telah ada tetapi belum terhubungkan satu sama lain. Untuk menghubungkannya perlu adanya pemberian nilai *property command connection* dengan *connection* itu sendiri.



Berikut ini adalah pilihan kedua bagaimana menghubungkan connection dan command, yaitu melalui pemanggilan method connection **CreateCommand**:

```
Dim CMD As MySqlCommand = conn.CreateCommand
```

Kode di atas identik dengan kode berikut ini:

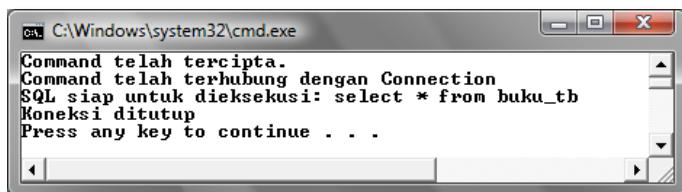
```
Dim CMD As MySqlCommand = New MySqlCommand()  
CMD.Connection = conn
```

4.2.3 Memasukkan Text SQL Ke Command

Salah satu property Command adalah **CommandText**, yaitu sebagai tempat untuk menyimpan pernyataan-pernyataan SQL yang akan dijalankan. Anda dapat memberi nilai property ini secara langsung atau melalui *constructor* command.

```
Try  
    'membuka connection  
    conn.Open()  
  
    'menghubungkan command ke connection  
    CMD.Connection = conn  
    Console.WriteLine("Command telah terhubung dengan Connection")  
  
    'menghubungkan SQL dengan command  
    Dim SQL As String  
    SQL = "select count(*) from buku_tb"  
    CMD.CommandText = SQL  
    Console.WriteLine("SQL siap untuk dieksekusi: " & CMD.CommandText)
```

Jalankan programnya dengan menekan tombol keyboard **Ctrl+F5**, seharusnya tampilannya seperti berikut ini:



CommandText merupakan string biasa saja sehingga Anda dapat mencetak CommandText melalui pernyataan **Console.WriteLine()** seperti string lainnya. Berikut ini contoh bagaimana SQL akan mengembalikan jumlah buku yang diperoleh dari table **buku_tb**. Sebagai alternatif Anda dapat mengatur property ini saat membuat *constructor* command seperti yang terlihat berikut ini:

```
Try
    'membuka connection
    conn.Open()

    'menghubungkan command ke connection
    CMD.Connection = conn
    Console.WriteLine("Command telah terhubung dengan Connection")

    'menghubungkan SQL dengan command
    Dim SQL As String
    SQL = "select count(*) from buku_tb"
    CMD = New MySqlCommand(SQL, conn)
    Console.WriteLine("SQL siap untuk dieksekusi: " & CMD.CommandText)
```

4.2.4 Eksekusi Command

Command tidak akan berguna sebelum di-eksekusi. Command mempunyai beberapa method berbeda dalam mengeksekusi SQL. Perbedaan method tergantung dari hasil yang diharapkan terhadap SQL. Adapun method eksekusi command yang tersedia antara lain:

Item	Keterangan
ExecuteReader	Mengeksekusi command yang menghasilkan baris-baris <i>record</i> (baris data).
ExecuteNonQuery	Mengeksekusi command tanpa menghasilkan baris-baris record, seperti pernyataan SQL INSERT, DELETE, dan UPDATE.
ExecuteScalar	Mengeksekusi command yang menghasilkan satu nilai saja, terutama nilai <i>aggregate</i> dari database (contoh pernyataan SUM, COUNT, MIN, MAX, dll.)

Contoh berikut ini memanfaatkan method **ExecuteScalar**:

- Buat Project baru bernama **CommandScalar** dengan mengeklik menu **File->New->Project**.
- Pilih template **Console Application** hingga terbentuk file **Module1.vb**.
- Tambahkan reference **MySQL.Data** pada menu **Project->Add Reference->.NET->MySQL.Data**.

CommandScalar: Module1.vb

```
Imports MySql.Data.MySqlClient

Module Module1

    Sub Main()
        Dim connString, db, user, pwd As String

        db = "dbpenjualanbuku" 'database
        user = "root" 'user default
        pwd = "" 'password kosong
        connString = "Database=" & db & ";Data Source=localhost;User
Id=" & user & ";Password=" & pwd

        'membuat string query
        Dim SQL As String = "select count(*) from buku_tb"

        'membuat connection
        Dim conn As MySqlConnection = New MySqlConnection(connString)

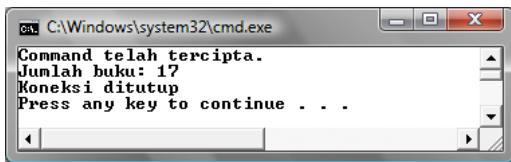
        'membuat command
        Dim CMD As MySqlCommand = New MySqlCommand(SQL, conn)
        Console.WriteLine("Command telah tercipta.")

        Try
            'membuka connection
            conn.Open()

            Dim jumlah As Integer
            'eksekusi query
            jumlah = CType(CMD.ExecuteScalar, Integer)
            Console.WriteLine("Jumlah buku: " & jumlah)
        Catch ex As Exception
            'menampilkan pesan kesalahan
            Console.WriteLine("Error: " & ex.ToString())
        Finally
            'menutup connection
            conn.Close()
            Console.WriteLine("Koneksi ditutup")
        End Try
    End Sub
End Module
```

```
End Sub  
End Module
```

Jalankan programnya dengan menekan tombol keyboard **Ctrl+F5**, keluarannya kurang lebih seperti berikut ini:



Pemanggilan **ExecuteScalar** sangatlah serderhana, yaitu:

```
'eksekusi query  
jumlah = CType(CMD.ExecuteScalar, Integer)  
Console.WriteLine("Jumlah buku: " & jumlah)
```

Karena **ExecuteScalar** adalah object, maka sebaiknya Anda mengubah object tersebut menjadi variable yang lebih spesifik, misalnya integer. Oleh karena itu diperlukan method **CType**.

4.2.5 Menjalankan Command Yang Menghasilkan Banyak Baris

Jika query menghasilkan banyak baris (*record*) dan kolom (*field*), maka Anda harus menggunakan method command **ExecuteReader()**. **ExecuteReader()** mengembalikan sebuah *data reader* yaitu object dari class **MySQLDataReader** (akan dipelajari lebih detil pada bab selanjutnya). *Data reader* mempunyai method-method yang mengizinkan Anda untuk membaca baris secara berturut-turut di dalam *result set* serta melakukan pencarian nilai kolomnya masing-masing.

Contoh program yang memanfaatkan *data reader*:

- Buat Project baru bernama **CommandReader** dengan mengeklik menu **File->New->Project**.
- Pilih template **Console Application** hingga terbentuk file **Module1.vb**.
- Tambahkan reference **MySQL.Data** pada menu **Project->Add Reference->.NET->MySQL.Data**.

CommandReader: Module1.vb

```

Imports MySql.Data.MySqlClient

Module Module1

    Sub Main()
        Dim connString, db, user, pwd As String

        db = "dbpenjualanbuku" 'database
        user = "root" 'user default
        pwd = "" 'password kosong
        connString = "Database=" & db & ";Data Source=localhost;User
Id=" & user & ";Password=" & pwd

        'membuat string query
        Dim SQL As String = "select kode_penulis, nama_penulis from
penulis_tb"

        'membuat connection
        Dim conn As MySqlConnection = New MySqlConnection(connString)

        'membuat command
        Dim CMD As MySqlCommand = New MySqlCommand(SQL, conn)
        Console.WriteLine("Command telah tercipta dan telah
terhubung.")

        Try
            'membuka connection
            conn.Open()

            'eksekusi query
            Dim rdr As MySqlDataReader = CMD.ExecuteReader
            While rdr.Read
                Console.WriteLine("Kode: {0} Penulis: {1}",
rdr.GetValue(0), rdr.GetValue(1))
            End While
            Catch ex As Exception
                'menampilkan pesan kesalahan
                Console.WriteLine("Error: " & ex.ToString)
            Finally
                'menutup connection
                conn.Close()
                Console.WriteLine("Koneksi ditutup")
        End Try
    End Sub
End Module

```

```
    End Try  
End Sub  
End Module
```

Jalankan programnya dengan menekan tombol keyboard **Ctrl+F5**, hasilnya kurang lebih seperti berikut ini:



```
ca C:\Windows\system32\cmd.exe  
Command telah tercipta dan telah terhubung.  
Kode: 000000001 Penulis: Handi Chandra  
Kode: 000000002 Penulis: Dodit Suprianto  
Kode: 000000003 Penulis: Bunafit Nugroho  
Kode: 000000004 Penulis: Banbang Hariyanto  
Kode: 000000005 Penulis: Hengky W Pranana  
Kode: 000000006 Penulis: Ahmad Sofyan  
Kode: 000000007 Penulis: Rini Agustina  
Kode: 000000008 Penulis: Tiffany Azhar Izzuddin  
Kode: 000000009 Penulis: Jaja Jamaludin Malik  
Kode: 000000010 Penulis: Firdaus  
Kode: 000000011 Penulis: Errima Oneto  
Kode: 000000012 Penulis: Haris Supriansyah  
Kode: 000000013 Penulis: Riveke Ustdiyanto  
Kode: 000000014 Penulis: Bernard Renaldi Suteja  
Kode: 000000015 Penulis: Tjahjanto Pudji Juwono  
Koneksi ditutup  
Press any key to continue . . .
```

Contoh di atas menggunakan method **ExecuteReader()** untuk melakukan penelusuran dan mengambil baris berdasarkan kolom **kode_penulis** dan **nama_penulis** dari table **penulis_tb**. Jika sebelumnya method **ExecuteReader()** mengembalikan satu nilai scalar maka dengan method **ExecuteReader()** akan dikembalikan sebuah object **MySQLDataReader()**.



Object **MySQLDataReader** mempunyai method **Read** yang menangkap setiap baris secara bergilir dan method **GetValue** yang mendapatkan nilai berdasar kolom pada setiap baris. Penentuan nilai kolom berdasarkan parameter integer sebagai tanda daftar urutan (*index*) kolom dari pernyataan Select, index dimulai dari angka 0, sehingga kolom **kode_penulis** menempati index ke-0 dan kolom **nama_penulis** menempati index ke-1.

4.2.6 Eksekusi Pernyataan

Method **ExecuteNonQuery** digunakan untuk mengeksekusi pernyataan query yang tidak membutuhkan hasil balik, misalnya pernyataan Insert, Update, dan Delete.

- Buat Project baru bernama **CommandNonQuery** dengan mengeklik menu **File->New->Project**
- Template pilih dengan **Console Application** hingga terbentuk file **Module1.vb**.
- Tambahkan reference **MySQL.Data** pada menu **Project->Add Reference->.NET->MySQL.Data**.

CommandNonQuery

```
Imports MySql.Data.MySqlClient

Module Module1

    Sub Main()
        Dim connString, db, user, pwd As String

        db = "dbpenjualanbuku" 'database
        user = "root" 'user default
        pwd = "" 'password kosong
        connString = "Database=" & db & ";Data Source=localhost;User
Id=" & user & ";Password=" & pwd

        'membuat string query
        Dim SqlQry As String = "select count(*) from penulis_tb"
        'membuat pernyataan insert
```

```

Dim SqlIns As String
SqlIns = "insert into penulis_tb(kode_penulis, nama_penulis)
"
SqlIns = SqlIns & "values('0000000016', 'Rony Sianturi')"

'membuat pernyataan delete
Dim SqlDel As String
SqlDel = "delete from penulis_tb where "
SqlDel = SqlDel & "kode_penulis='0000000016' "
SqlDel = SqlDel & "and nama_penulis='Rony Sianturi'"

'membuat connection
Dim conn As MySqlConnection = New MySqlConnection(connString)

'membuat command
Dim CmdQry As MySqlCommand = New MySqlCommand(SqlQry, conn)
Dim CmdNon As MySqlCommand = New MySqlCommand(SqlIns, conn)
Console.WriteLine("Command telah tercipta dan terkoneksi.")

Try
    'membuka connection
    conn.Open()

    'eksekusi query untuk mendapatkan jumlah penulis
    Console.WriteLine("Sebelum INSERT: Jumlah penulis = " &
CmdQry.ExecuteScalar)

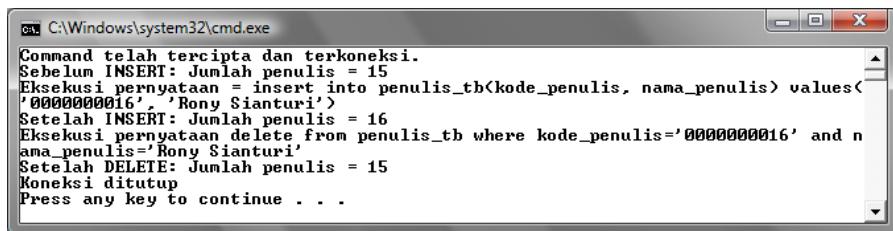
    'eksekusi nonquery untuk memasukkan penulis baru
    Console.WriteLine("Eksekusi pernyataan = " &
CmdNon.CommandText)
    CmdNon.ExecuteNonQuery()
    Console.WriteLine("Setelah INSERT: Jumlah penulis = " &
CmdQry.ExecuteScalar)

    'eksekusi nonquery untuk menghapus penulis
    CmdNon.CommandText = SqlDel
    Console.WriteLine("Eksekusi pernyataan " &
CmdNon.CommandText)
    CmdNon.ExecuteNonQuery()
    Console.WriteLine("Setelah DELETE: Jumlah penulis = " &
CmdQry.ExecuteScalar)

Catch ex As Exception
    'menampilkan pesan kesalahan
    Console.WriteLine("Error: " & ex.ToString)
Finally
    'menutup connection
    conn.Close()
    Console.WriteLine("Koneksi ditutup")
End Try
End Sub
End Module

```

Jalankan programnya dengan menekan tombol keyboard **Ctrl+F5**, hasilnya kurang lebih seperti berikut ini:



```
C:\> C:\Windows\system32\cmd.exe
Command telah tercipta dan terkoneksi.
Sebelum INSERT: Jumlah penulis = 15
Eksekusi pernyataan = insert into penulis_tb(kode_penulis, nama_penulis) values('0000000016', 'Rony Sianturi')
Setelah INSERT: Jumlah penulis = 16
Eksekusi pernyataan delete from penulis_tb where kode_penulis='0000000016' and nama_penulis='Rony Sianturi'
Setelah DELETE: Jumlah penulis = 15
Koneksi ditutup
Press any key to continue . . .
```

Program di bawah ini menggunakan satu **query scalar** dan dua **query jenis pernyataan** yang tersimpan pada tiga variable string.

Query Scalar

```
'membuat string query
Dim SqlQry As String = "select count(*) from penulis_tb"
```

Query jenis pernyataan

```
'membuat pernyataan insert
Dim SqlIns As String
SqlIns = "insert into penulis_tb(kode_penulis, nama_penulis) "
SqlIns = SqlIns & "values('0000000016', 'Rony Sianturi')"
```

```
'membuat pernyataan delete
Dim SqlDel As String
SqlDel = "delete from penulis_tb where "
SqlDel = SqlDel & "kode_penulis='0000000016' "
SqlDel = SqlDel & "and nama_penulis='Rony Sianturi'"
```

Di sini tercipta dua command, pertama **CmdQry** yang menghitung jumlah baris dari table **penulis_tb**. Anda menggunakan command ini beberapa kali yang bertujuan untuk memonitor perubahan jumlah baris saat terjadi penambahan dan penghapusan penulis. Kedua adalah command **CmdNon** yang digunakan dua kali, pertama untuk menambah baris (insert) dan kemudian menghapus baris (delete) pada baris yang sama. Anda perlu meng-inisialisasi pengaturan CommandText ke pernyataan SQL INSERT.

```
Dim CmdNon As MySqlCommand = New MySqlCommand(SqlIns, conn)
```

```
...  
CmdNon.ExecuteNonQuery()
```

Lakukan hal yang sama namun dengan cara berbeda untuk SQL DELETE. Command **CmdNon** digunakan kembali.

```
CmdNon.CommandText = SqlDel  
...  
CmdNon.ExecuteNonQuery()
```

Eksekusi pernyataan SQL dipanggil dua kali

```
CmdNon.ExecuteNonQuery()
```

Karena INSERT dan DELETE tidak mengembalikan hasil apapun, maka **ExecuteNonQuery** digunakan, sebaliknya untuk menampilkan hanya satu nilai, gunakan **ExecuteScalar**.

4.2.7 Command Parameter

Saat Anda menambahkan baris baru ke dalam table penulis (contoh sebelumnya), Anda memasukkan nilai-nilainya secara *hard-coded* tanpa melewati parameter, cara seperti itu jarang sekali terjadi karena bersifat statis. Terutama jika pemasukkan data diperoleh dari variable lain atau **form-textbox** yang bersifat dinamis (data yang dimasukkan bisa berganti-ganti). Sehingga command parameter sangat diperlukan. Beberapa keuntungan penggunaan command parameter:

- Memetakan antara variable dan dimana variable tersebut digunakan dalam SQL menjadi jelas.
- Dengan parameter dapat ditentukan tipe datanya secara spesifik, sesuai dengan data provider ADO.NET. Untuk memastikan bahwa variable yang terpetakan adalah tipe data SQL yang benar.
- Parameter digunakan lebih luas dalam teknik pemrograman lainnya, misalnya store procedure dan store function.

Sekarang mari kita modifikasi kode program sebelumnya dengan penambahan fitur command parameter.

- Buat Project baru bernama **CommandParameter** dengan mengeklik menu **File->New->Project**.
- Pilih template dengan **Console Application** hingga terbentuk file **Module1.vb**.
- Tambahkan reference **MySQL.Data** di menu **Project->Add Reference->.NET->MySQL.Data**.

CommandParameter: Module1.vb

```
Imports MySql.Data.MySqlClient

Module Module1

    Sub Main()
        Dim connString, db, user, pwd As String

        db = "dbpenjualanbuku" 'database
        user = "root" 'user default
        pwd = "" 'password kosong
        connString = "Database=" & db & ";Data Source=localhost;User
Id=" & user & ";Password=" & pwd

        'isi variable sebagai data mentah
        Dim Kode As String = "0000000016"
        Dim Penulis As String = "Rony Sianturi"

        'membuat string query
        Dim SqlQry As String = "select count(*) from penulis_tb"

        'membuat pernyataan insert
        Dim SqlIns As String
        SqlIns = "insert into penulis_tb(kode_penulis, nama_penulis)"

        SqlIns = SqlIns & "values('@kode', '@penulis')"

        'membuat pernyataan delete
        Dim SqlDel As String
        SqlDel = "delete from penulis_tb where "
        SqlDel = SqlDel & "kode_penulis='@kode' "
        SqlDel = SqlDel & "and nama_penulis='@penulis'"

        'membuat connection
        Dim conn As MySqlConnection = New MySqlConnection(connString)
```

```

'membuat command
Dim CmdQry As MySqlCommand = New MySqlCommand(SqlQry, conn)
Dim CmdNon As MySqlCommand = New MySqlCommand(SqlIns, conn)
Console.WriteLine("Command telah tercipta dan terkoneksi.")

'menambahkan parameter ke command untuk mengeksekusi
pernyataan
CmdNon.Parameters.Add("@kode", MySqlDbType.String, 10)
CmdNon.Parameters.Add("@penulis", MySqlDbType.VarChar, 30)

Try
    'membuka connection
    conn.Open()

    'eksekusi query untuk mendapatkan jumlah penulis
    Console.WriteLine("Sebelum INSERT: Jumlah penulis = " &
CmdQry.ExecuteScalar()

    'eksekusi nonquery untuk memasukkan penulis baru
    CmdNon.Parameters("@kode").Value = Kode
    CmdNon.Parameters("@penulis").Value = Penulis

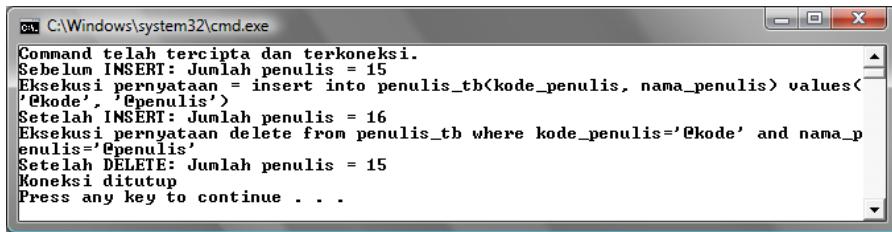
    Console.WriteLine("Eksekusi pernyataan = " &
CmdNon.CommandText)
    CmdNon.ExecuteNonQuery()
    Console.WriteLine("Setelah INSERT: Jumlah penulis = " &
CmdQry.ExecuteScalar()

    'eksekusi nonquery untuk menghapus penulis
    CmdNon.CommandText = SqlDel
    Console.WriteLine("Eksekusi pernyataan " &
CmdNon.CommandText)
    CmdNon.ExecuteNonQuery()
    Console.WriteLine("Setelah DELETE: Jumlah penulis = " &
CmdQry.ExecuteScalar())

    Catch ex As Exception
        'menampilkan pesan kesalahan
        Console.WriteLine("Error: " & ex.ToString)
    Finally
        'menutup connection
        conn.Close()
        Console.WriteLine("Koneksi ditutup")
    End Try
End Sub
End Module

```

Jalankan program dengan menekan tombol keyboard **Ctrl+F5**, hasilnya kurang lebih seperti berikut ini:



```
C:\Windows\system32\cmd.exe
Command telah tercipta dan terkoneksi.
Sebelum INSERT: Jumlah penulis = 15
Esekusi pernyataan = insert into penulis_tb(kode_penulis, nama_penulis) values(
'@kode', '@penulis')
Setelah INSERT: Jumlah penulis = 16
Esekusi pernyataan delete from penulis_tb where kode_penulis='@kode' and nama_p
enulis='@penulis'
Setelah DELETE: Jumlah penulis = 15
Koneksi ditutup
Press any key to continue . . .
```

Pertama adalah menyiapkan contoh data:

```
'isi variable sebagai data mentah
Dim Kode As String = "0000000016"
Dim Penulis As String = "Rony Sianturi"
```

Kemudian menambahkan dua parameter, **@kode** dan **@penulis**:

```
'membuat pernyataan insert
Dim SqlIns As String
SqlIns = "insert into penulis_tb(kode_penulis, nama_penulis) "
SqlIns = SqlIns & "values('@kode', '@penulis')"

'membuat pernyataan delete
Dim SqlDel As String
SqlDel = "delete from penulis_tb where "
SqlDel = SqlDel & "kode_penulis='@kode' "
SqlDel = SqlDel & "and nama_penulis='@penulis'"
```

Dan juga ke *parameter collection property* dari command yang Anda inginkan untuk diberi parameter.

```
'menambahkan parameter ke command untuk pengeksekusian pernyataan
CmdNon.Parameters.Add("@kode", MySqlDbType.String, 10)
CmdNon.Parameters.Add("@penulis", MySqlDbType.VarChar, 30)
```



Anda telah menyediakan nama parameter dan menentukan tipe data kolom yang diharapkan. Penentuan tipe data kolom mengacu pada anggota **MySqlDbType**, setara dengan tipe data database MySQL. Terakhir

adalah pemberian nilai terhadap parameter sebelum eksekusi command dilakukan:

```
'eksekusi nonquery untuk memasukkan penulis baru  
CmdNon.Parameters("@kode").Value = Kode  
CmdNon.Parameters("@penulis").Value = Penulis
```

4.3 Data Reader

Data reader memiliki kinerja yang cepat, *unbuffered* (hasil tidak ditempatkan pada memori khusus), *forward-only* (cursor penunjuk record hanya bisa maju), terhubung secara *read-only* (hanya bisa dibaca) pada setiap pembacaan data per baris. Pembacaan hanya dapat dilakukan satu kali pada setiap perulangan *result set*. Anda tidak dapat membuat object (*instantiate*) data reader secara langsung melainkan harus melalui method command **ExecuteReader**. Sebagai contoh, anggap saja **cmd** adalah object command maka object data reader-nya menjadi:

```
Dim rdr As MySqlDataReader = cmd.ExecuteReader()
```

Sekarang Anda dapat menggunakan data reader ini untuk mengakses query *result set*. Berikut ini contoh perulangan pada Result Set:

- Buat Project baru bernama **CommandParameter** dengan mengklik menu **File->New->Project**.
- Pilih template dengan **Console Application** hingga terbentuk file **Module1.vb**.
- Tambahkan reference **MySQL.Data** pada menu **Project->Add Reference->.NET->MySQL.Data**.

DataLooper: Module1.vb

```
Imports MySql.Data.MySqlClient  
  
Module Module1  
  
    Sub Main()  
        Dim connString, db, user, pwd As String
```

```

db = "dbpenjualanbuku" 'database
user = "root" 'user default
pwd = "" 'password kosong
connString = "Database=" & db & ";Data Source=localhost;User
Id=" & user & ";Password=" & pwd

'set up query
Dim sql As String = "select nama_penulis from penulis_tb"

'Membuat connection
Dim conn As MySqlConnection = New MySqlConnection(connString)

Try
    'membuka connection
    conn.Open()

    'membuat command
    Dim cmd As MySqlCommand = New MySqlCommand(sql, conn)

    'membuat data reader
    Dim rdr As MySqlDataReader = cmd.ExecuteReader

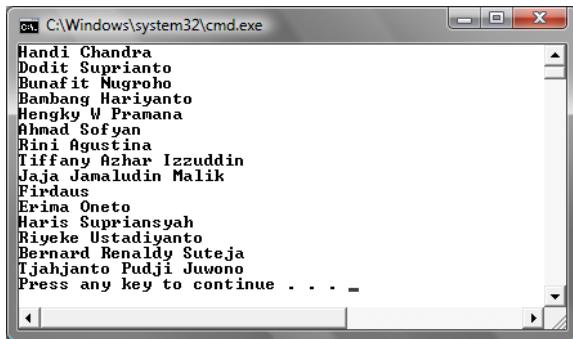
    'perulangan yang melalui result set
    While rdr.Read
        Console.WriteLine(rdr(0))
    End While

    'menutup data reader
    rdr.Close()

    Catch ex As Exception
        'menampilkan pesan kesalahan
        Console.WriteLine("Error: " & ex.ToString)
    Finally
        'menutup connection
        conn.Close()
    End Try
End Sub
End Module

```

Jalankan program dengan menekan tombol keyboard Ctrl+F5, hasilnya kurang lebih seperti berikut ini:



A screenshot of a Windows Command Prompt window titled 'C:\Windows\system32\cmd.exe'. The window contains a list of names, each on a new line, starting with 'Handi Chandra' and ending with 'Tjahjanto Pudji Juwono'. The text is in black font on a white background. At the bottom of the list, it says 'Press any key to continue . . . -'.

MySqlDataReader adalah class abstrak yang tidak dapat dibuat instansiasi-nya (penciptaan object) secara nyata, oleh karena itu Anda harus membuat *instance* **MySqlDataReader** dengan menjalankan method **ExecuteReader** dari **MySqlCommand**:

```
'membuat data reader
Dim rdr As MySqlDataReader = cmd.ExecuteReader
```

ExecuteReader tidak sekedar membuat data reader tetapi juga mengirimkan pernyataan SQL ke connection untuk dieksekusi, sehingga Anda dapat mencacah setiap baris result set yang dihasilkan, kemudian melakukan pencarian berdasarkan kolom yang dikehendaki. Untuk melakukan hal ini, Anda harus memanggil metod **Read** dari **MySqlDataReader** yang akan mengembalikan nilai **true** jika baris masih tersedia dan cursor masih berlanjut (pointer atau penunjuk ke baris selanjutnya di dalam *result set*) atau bernilai **false** jika baris sudah tidak tersedia atau tidak ditemukan lagi. Karena **Read** melakukan penelusuran berdasarkan baris berikutnya maka Anda dapat melakukan perulangan terhadap result set melalui pernyataan kondisi **While** seperti berikut ini:

```
'perulangan yang melalui result set
While rdr.Read
    Console.WriteLine(rdr(0))
End While
```

Sekali method **Read** dipanggil maka baris berikutnya dikembalikan sebagai *collection* dan disimpan dalam object **MySqlDataReader** itu sendiri. Untuk mengakses data dari kolom tertentu, Anda dapat menggunakan beberapa metode, tetapi untuk saat ini Anda hanya menggunakan nomor index yang menunjukkan urutan kolom untuk membaca nilai yang sedang dicari. Karena SQL hanya menghasilkan kolom tunggal yaitu **nama_penulis** dari table **penulis_tb** maka index kolom yang tersedia adalah index ke-0 saja, dan penulisannya adalah **rdr(0)**.

Setelah menggunakan data reader ada baiknya untuk ditutup secara eksplisit. Hal ini mencegah connection yang aktif saat ini melakukan *fetch* data (pengambilan data) secara terus menerus. Untuk melepaskan sumber daya data reader yang sudah terpakai dengan cara memanggil method **close**.

```
'menutup data reader  
rdr.Close()
```

4.3.1 Penggunaan Urutan Index

Urutan index berguna untuk mencari data kolom dari result set. Kita akan mempelajari lebih jauh mengenai urutan index, dimana kode tersebut merujuk pada property item data reader, dan mengembalikan nilai dalam kolom tertentu pada baris yang aktif saat itu. Karena **rdr(0)** mengembalikan nilai object maka sebaiknya dilakukan konversi data dari object ke tipe data yang sesuai.

- Buat Project baru bernama **OrdinalIndexer** dengan mengeklik menu **File->New->Project**.
- Pilih template dengan **Console Application** hingga terbentuk file **Module1.vb**.
- Tambahkan reference **MySQL.Data** pada menu **Project->Add Reference->.NET->MySQL.Data**.

OrdinalIndexer: Module1.vb

```
Imports MySql.Data.MySqlClient

Module Module1

    Sub Main()
        Dim connString, db, user, pwd As String

        db = "dbpenjualanbuku" 'database
        user = "root" 'user default
        pwd = "" 'password kosong
        connString = "Database=" & db & ";Data Source=localhost;User
Id=" & user & ";Password=" & pwd

        'set up query
        Dim sql As String
        sql = "select kode_penulis, nama_penulis from penulis_tb "
        sql = sql & "where nama_penulis like 'B%'"

        'Membuat connection
        Dim conn As MySqlConnection = New MySqlConnection(connString)

        Try
            'membuka connection
            conn.Open()

            'membuat command
            Dim cmd As MySqlCommand = New MySqlCommand(sql, conn)

            'membuat data reader
            Dim rdr As MySqlDataReader = cmd.ExecuteReader

            'cetak heading
            Console.WriteLine(" {0} {1}", "Kode".PadRight(10),
"Nama Penulis".PadRight(30))
            Console.WriteLine(" {0} {1}", "-----".
PadRight(10), "-----".PadRight(30))

            'perulangan melalui result set
            While rdr.Read
                Console.WriteLine(" {0} | {1}",
rdr(0).ToString.PadRight(10), rdr(1).ToString.PadRight(30))
            End While

            'menutup data reader
            rdr.Close()

        Catch ex As Exception
            'menampilkan pesan kesalahan
            Console.WriteLine("Error: " & ex.ToString)
        Finally
```

```

        'menutup connection
        conn.Close()
    End Try
End Sub
End Module

```

Jalankan programnya dengan menekan tombol keyboard Ctrl+F5, hasilnya kurang lebih seperti berikut ini:

Kode	Nama Penulis
0000000003	Bunafit Nugroho
0000000004	Bambang Hariyanto
0000000014	Bernard Renaldy Suteja

Anda membutuhkan table **penulis_tb** untuk menampilkan kolom **kode_penulis** dan **nama_penulis**, baris yang ditampilkan disaring berdasarkan nama penulis yang berawalan B (ditandai oleh '%B'):

```

'set up query
Dim sql As String
sql = "select kode_penulis, nama_penulis from penulis_tb "
sql = sql & "where nama_penulis like 'B%'"
```

Karena pada query terdapat dua kolom yang akan ditampilkan, maka urutan index-nya terdiri dari 0 dan 1. Pembacaan baris dilakukan melalui perulangan **While** dan pada setiap baris penelusuran diambil datanya melalui kolom indexnya masing-masing. Karena yang dikembalikan adalah object, maka perlu dikonversi nilainya terlebih dahulu menjadi string (ToString). Anda juga bisa memformat hasil keluarannya dengan method **PadLeft**:

```

'cetak heading
Console.WriteLine(" {0} {1}", "Kode".PadRight(10), "Nama
Penulis".PadRight(30))
Console.WriteLine(" {0} {1}", "-----".PadRight(10), "-----"
--".PadRight(30))

'perulangan melalui result set
While rdr.Read
```

```
    Console.WriteLine(" {0} | {1}", rdr(0).ToString().PadRight(10),
rdr(1).ToString().PadRight(30))
End While
```

Setelah proses selesai, lakukan penutupan terhadap data reader secara eksplisit, untuk memastikan bahwa sumber daya telah terbebas:

```
'menutup data reader
rdr.Close()
```

4.3.2 Penggunaan Urutan Nama Kolom

Jika melibatkan banyak kolom dalam query untuk ditampilkan maka akan memaksa kita untuk mengingat setiap nomor index kolom yang ada, apakah sesuai dengan yang diharapkan. Oleh karena itu kita diberi kemudahan untuk mengambil data berdasarkan nama kolomnya masing-masing. Dari contoh sebelumnya, maka index ke-0 mewakili nama kolom **kode_penulis** dan index ke-1 mewakili nama kolom **nama_penulis**, sehingga kode program dapat diubah menjadi:

```
'perulangan melalui result set
While rdr.Read
    Console.WriteLine(" {0} | {1}",
rdr("kode_penulis").ToString().PadRight(10),
rdr("nama_penulis").ToString().PadRight(30))
End While
```

Catatan: Proses pengambilan data berdasarkan index kolom akan jauh lebih cepat dibandingkan jika mengacu pada nama kolom, karena merujuk secara langsung nomor index-nya tanpa melalui penerjemahan nama kolom ke index terlebih dahulu.

4.3.3 Penggunaan Method Type Accessor

Ketika data reader mengembalikan suatu nilai yang berasal dari data source, maka hasil nilai tersebut dicari dan disimpan ke dalam tipe .NET, bukan tipe data asli yang bersumber dari data source itu sendiri (MySQL).

Proses pengkonversian secara internal tersebut berpengaruh pada konsistensi dan kecepatan, oleh karena itu data reader menyediakan fitur *typed accessor methods*. Anda dapat menentukan secara spesifik tipe dari nilai yang sedang dikembalikan dan tidak lagi mengikuti tipe data milik .NET.

Seluruh method **type accessor** pasti berawalan **Get**, kemudian disusul oleh urutan index data yang sedang dicari. Method ini menghasilkan keluaran jauh lebih cepat daripada pengambilan data berdasarkan nomor index maupun yang mengacu pada nama kolom. Menjadi lebih cepat karena menggunakan dua metode di atas sekaligus, yaitu pengambilan data berdasarkan nomor index dan bukan berdasarkan nama kolom yang memerlukan penterjemahan dari nama kolom ke nomor index. Tidak diperlukan pula pengkonversian data ke tipe .NET, karena kita sudah menentukannya secara spesifik. Table daftar **MySQL Typed Accessor**:

Tipe Data MySQL	Tipe .NET	Tipe Accessor .NET
Tinyint	Byte	GetByte
Smallint	Int16	GetInt16
Mediumint	Int16	GetInt16
Int	Int32	GetInt32
Integer	Int32	GetInt32
Bignum	Int64	GetInt64
Bit	Boolean	GetBoolean
Real	Single	GetFloat
Double	Double	GetDouble
Float	Double	GetDouble
Decimal	Decimal	GetDecimal
Numeric	Decimal	GetDecimal
Char	String atau Char[]	GetString atau GetChars
Varchar	String atau Char[]	GetString atau GetChars
Date	Int16	GetInt16
Time	Int16	GetInt16

Year	Int16	GetInt16
Timestamp	Byte[]	GetByte
Datetime	DateTime	GetDateTime
Tinyblob	Byte[]	GetBytes
Blob	Byte[]	GetBytes
Mediumblob	Byte[]	GetBytes
Longblob	Byte[]	GetBytes
Tinytext	String atau Char[]	GetString atau GetChars
Text	String atau Char[]	GetString atau GetChars
Mediumtext	String atau Char[]	GetString atau GetChars
Longtext	String atau Char[]	GetString atau GetChars
Binary	Byte[]	GetBytes
Varbinary	Byte[]	GetBytes

Berikut ini adalah struktur table **buku_tb** yang akan dibuat sebagai contoh programnya, kotak yang berwarna gelap adalah kolom yang hendak ditampilkan:

Name	Type	Length	Decimals	Allow Null	Primary Key
kode_buku	char	13	0	<input type="checkbox"/>	<input checked="" type="checkbox"/> 1
isbn	char	10	0	<input checked="" type="checkbox"/>	
judul	tinytext	0	0	<input type="checkbox"/>	
kode_penulis	char	10	0	<input type="checkbox"/>	
kode_penerbit	char	10	0	<input type="checkbox"/>	
kode_kelompok	char	10	0	<input type="checkbox"/>	
harga_jual	double	0	0	<input type="checkbox"/>	
sinopsis	text	0	0	<input checked="" type="checkbox"/>	
tahun_terbit	smallint	6	0	<input checked="" type="checkbox"/>	
jumlah_halaman	int	11	0	<input checked="" type="checkbox"/>	
foto	blob	0	0	<input checked="" type="checkbox"/>	
nama_foto	tinytext	0	0	<input checked="" type="checkbox"/>	

Default:

Comment:

Character set: latin1

Number of Field: 12

- Buat Project baru bernama **TypedAccessor** dengan mengeklik menu **File->New->Project**.
- Template pilih dengan **Console Application** hingga terbentuk file **Module1.vb**.
- Tambahkan reference **MySQL.Data** pada menu **Project->Add Reference->.NET->MySQL.Data**.

TypedAccessor: Module1.vb

```
Imports MySql.Data.MySqlClient

Module Module1

    Sub Main()
        Dim connString, db, user, pwd As String

        db = "dbpenjualanbuku" 'database
        user = "root" 'user default
        pwd = "" 'password kosong
        connString = "Database=" & db & ";Data Source=localhost;User
Id=" & user & ";Password=" & pwd

        'set up query
        Dim sql As String
        sql = "select judul, harga_jual, tahun_terbit, jumlah_halaman
"
        sql = sql & "from buku_tb "

        'Membuat connection
        Dim conn As MySqlConnection = New MySqlConnection(connString)

        Try
            'membuka connection
            conn.Open()

            'membuat command
            Dim cmd As MySqlCommand = New MySqlCommand(sql, conn)

            'membuat data reader
            Dim rdr As MySqlDataReader = cmd.ExecuteReader

            'perulangan melalui result set
            While rdr.Read
                Console.WriteLine("{0} {1} {2} {3}", _
rdr.GetString(0).PadRight(50), _
rdr.GetDouble(1), _
rdr.GetInt16(2), _
rdr.GetInt32(3))
            End While
        Catch ex As Exception
            MsgBox(ex.Message)
        End Try
    End Sub
End Module
```

```

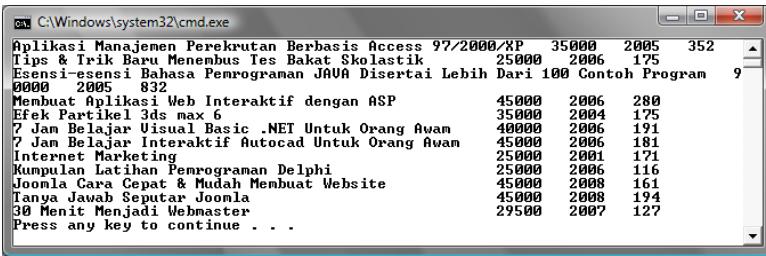
End While

    'menutup data reader
    rdr.Close()

Catch ex As Exception
    'menampilkan pesan kesalahan
    Console.WriteLine("Error: " & ex.ToString)
Finally
    'menutup connection
    conn.Close()
End Try
End Sub
End Module

```

Jalankan programnya dengan menekan tombol keyboard **Ctrl+F5**, hasilnya kurang lebih seperti berikut ini:



Aplikasi Manajemen Perekuturan Berbasis Access 97/2000/XP	35000	2005	352
Tips & Trik Baru Menembus Tes Bakat Skolastik	25000	2006	175
Esenensi-esensi Bahasa Pemrograman JAVA Disertai Lebih Dari 100 Contoh Program	9		
0000 2005 832			
Membuat Aplikasi Web Interaktif dengan ASP	45000	2006	280
Efek Partikel 3ds max 6	35000	2004	175
7 Jam Belajar Visual Basic .NET Untuk Orang Awam	40000	2006	191
7 Jam Belajar Interaktif Autocad Untuk Orang Awam	45000	2006	181
Internet Marketing	25000	2001	171
Kumpulan Latihan Pemrograman Delphi	25000	2006	116
Joomla Cara Cepat & Mudah Membuat Website	45000	2008	161
Tanya Jawab Sepertai Joomla	45000	2008	194
30 Menit Menjadi Webmaster	29500	2007	127

4.3.4 Mendapatkan Informasi Data

Berikut ini adalah beberapa method yang berguna untuk mengetahui informasi tentang scheme/database dan informasi yang berhubungan dengan *result set*. Tabel di bawah ini menjelaskan beberapa method metadata dan property dari *data reader*.

Method / Property	Keterangan
Depth	Property yang mendapatkan kedalaman perulangan baris saat ini.
FieldCount	Property yang menghitung jumlah kolom dalam baris saat ini.
GetDataTypeName	Method yang menerima index dan mengembalikan

	string berisi nama tipe data kolom.
GetFieldType	Method yang menerima index dan mengembalikan tipe Framework .NET object.
GetName	Method yang menerima index dan mengembalikan nama dari kolom yang telah ditentukan.
GetOrdinal	Method yang menerima nama kolom dan mengembalikan index kolom.
GetSchemaTable	Method yang mengembalikan kolom metadata.
HasRows	Property yang menandakan bahwa data reader mempunyai baris.
RecordsAffected	Property yang memperoleh sejumlah baris yang mengalami perubahan karena penambahan (INSERT) atau penghapusan (DELETE).

- Buat Project baru bernama **ResultSetInfo** dengan mengeklik menu **File->New->Project**.
- Template pilih dengan **Console Application** hingga terbentuk file **Module1.vb**.
- Tambahkan reference **MySQL.Data** pada menu **Project->Add Reference->.NET->MySQL.Data**.

ResultSetInfo: Module1.vb

```

Imports MySql.Data.MySqlClient

Module Module1

    Sub Main()
        Dim connString, db, user, pwd As String

        db = "dbpenjualanbuku" 'database
        user = "root" 'user default
        pwd = "" 'password kosong
        connString = "Database=" & db & ";Data Source=localhost;User
Id=" & user & ";Password=" & pwd

        'set up query
        Dim sql As String
        sql = "select kode_penulis, nama_penulis "
        sql = sql & "from penulis tb "
    End Sub
End Module

```

```

'sql = sql & "where nama_penulis = 'B%'" 

'Membuat connection
Dim conn As MySqlConnection = New MySqlConnection(connString)

Try
    'membuka connection
    conn.Open()

    'membuat command
    Dim cmd As MySqlCommand = New MySqlCommand(sql, conn)

    'membuat data reader
    Dim rdr As MySqlDataReader = cmd.ExecuteReader

    'mendapatkan nama kolom
    Console.WriteLine("Nama Kolom: {0} {1}", _
rdr.GetName(0).PadRight(10), _
rdr.GetName(1))

    'mendapatkan tipe data kolom
    Console.WriteLine("Tipe Data: {0} {1}", _
rdr.GetDataTypeName(0).PadRight(10), _
rdr.GetDataTypeName(1))

    Console.WriteLine()

    'perulangan melalui result set
    While rdr.Read
        Console.WriteLine(" {0} {1}", _
rdr.GetString(0).PadRight(10), _
rdr.GetString(1))
    End While

    'mendapatkan jumlah kolom
    Console.WriteLine()
    Console.WriteLine("Jumlah kolom pada setiap baris: {0}",

rdr.FieldCount)

    'mendapatkan informasi setiap kolom
    Console.WriteLine("{0}' berada di index {1} dan bertipe
{2}", _
rdr.GetName(0), _
rdr.GetOrdinal("kode_penulis"), _
rdr.GetFieldType(0))

    Console.WriteLine("{0}' berada di index {1} dan bertipe
{2}", _
rdr.GetName(1), _
rdr.GetOrdinal("nama_penulis"), _
rdr.GetFieldType(1))

```

```

'menutup data reader
rdr.Close()

Catch ex As Exception
    'menampilkan pesan kesalahan
    Console.WriteLine("Error: " & ex.ToString())
Finally
    'menutup connection
    conn.Close()
End Try
End Sub
End Module

```

Jalankan programnya dengan menekan tombol keyboard **Ctrl+F5**, hasilnya kurang lebih seperti berikut ini:

Nama Kolom: kode_penulis	Tipe Data: VARCHAR	Nama Penulis	Tipe Data: VARCHAR
0000000001		Handi Chandra	
0000000002		Dodit Suprianto	
0000000003		Bunafit Nugroho	
0000000004		Bambang Hariyanto	
0000000005		Hengky W Pramana	
0000000006		Ahmad Sofyan	
0000000007		Rini Agustina	
0000000008		Tiffany Azhar Izzuddin	
0000000009		Jaja Jamaludin Malik	
0000000010		Firdaus	
0000000011		Erima Oneto	
0000000012		Haris Supriansyah	
0000000013		Rieneke Ustadiyanto	
0000000014		Bernard Renaldy Suteja	
0000000015		Tjahjanto Pudji Juweono	

Jumlah kolom pada setiap baris: 2
'kode_penulis' berada di index 0 dan bertipe System.String
'nama_penulis' berada di index 1 dan bertipe System.String
Press any key to continue . . .

4.4 Dataset dan Data Adapter

Object lain yang digunakan untuk mengakses data adalah melalui **dataset**. Dataset secara penuh telah terbebas, maksudnya adalah pengaksesan data bisa dilakukan secara terhubung maupun terputus dari data source (database MySQL). Tujuan mendasar dari dataset adalah menyediakan *relational view* yang tersimpan dalam memori *chace* atau seolah-olah

terdapat database maupun termasuk relasi dan integrity constraint-nya pada memori *chace*.

Jika dataset tidak memerlukan database yang terhubung, lalu bagaimana kita dapat memanipulasi data seperti penyimpanan data kembali ke database? Oleh karenanya dibutuhkan *data adapter*. Data adapter seperti jembatan antara **dataset** dan **data source**. Tanpa data adapter maka dataset tidak dapat mengakses apapun dari data source. Data adapter mengurus semua detil *connection* ke dataset, memanipulasi data, dan memperbarui data source.

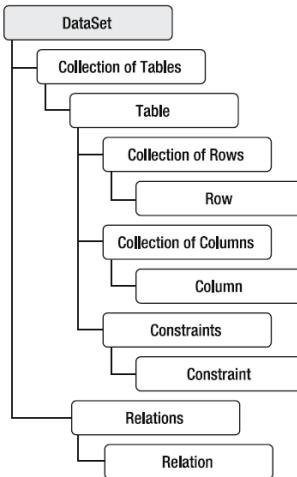
4.4.1 Perbedaan Dataset dan Data Reader

Jika Anda hendak membaca dan menampilkan data sederhana, sebaiknya gunakan data reader, seperti yang pernah dibahas pada bab sebelumnya khususnya jika pekerjaan melibatkan jumlah data sangat besar yang membutuhkan perulangan ribuan bahkan jutaan baris, akan lebih cepat pembacaan baris secara berurutan dengan menggunakan data reader (pembacaan baris-baris dari result set sekali dalam satu waktu) dan data reader mengerjakan pekerjaan ini secara efisien.

Jika Anda butuh manipulasi data dalam berbagai cara, kemudian meng-update database, maka gunakan dataset. Data adapter mengisi dataset melalui penggunaan data reader; sumberdaya tambahan dibutuhkan untuk menyimpan data yang penggunaannya bersifat terputus. Perlu dipikirkan apakah Anda benar-benar memerlukan dataset; karena hal ini bisa membuang sumberdaya yang ada. Kecuali Anda butuh meng-update data source (database mysql) atau menggunakan fitur dataset lainnya seperti pembacaan dan penulisan file XML, ekspor schema database, atau penciptaan view XML database, maka silahkan gunakan data reader.

4.4.2 Mengenal Dataset

Dalam ADO.NET dataset merupakan langkah besar di dunia pengembangan aplikasi database banyak tingkat (multitier). Ketika Anda melakukan pencarian atau memodifikasi sejumlah data yang sangat besar, maka penjagaan terhadap *open connection* ke data source (database mysql) selama proses tunggu oleh user saat meminta untuk dilayani (*request*) akan membuang sumberdaya yang sangat besar. Dataset sangat membantu disini, karena dataset mampu menyediakan dan memodifikasi jumlah data yang sangat besar namun berada pada sisi memori *chace* (memori lokal komputer), menampilkan data sebagai table, dan memproses data dalam mode *offline* dengan kata lain terputus dengan database. Bayangkan Anda sedang berusaha berhubungan ke server database jarak jauh melalui internet untuk mencari tahu informasi detil tentang transaksi bisnis. Anda mencari tanggal tertentu dari transaksi yang tersedia dan menampilkan hasilnya. Di belakang layar, aplikasi Anda membuat connection ke data source, menghubungkan beberapa table, dan menelusuri hasilnya. Kemudian Anda hendak mengubah informasi ini dan menambahkan atau menghapus detil informasinya. Apapun alasannya, aplikasi akan melakukan hal yang sama, menjadi suatu siklus yang berulang-ulang: membuat connection baru, menghubungkan table-table, menelusuri data. Ini menyebabkan beban berat bagi connection karena setiap saat harus membuat connection baru. Bukankah lebih baik jika Anda berhubungan dengan data source sekali saja, menyimpan datanya secara lokal dalam struktur yang disusun ulang menjadi database relasional, menutup connection, memodifikasi data secara lokal, dan kemudian menyimpan hasil perubahannya ke data source pada waktu yang tepat!. Karena kasus seperti inilah dataset dirancang, dataset menyimpan data relasional sebagai sekumpulan data table. Data table terdiri atas metadata yang menggambarkan struktur data dan data itu sendiri. Berikut ini bagan arsitektur dataset:



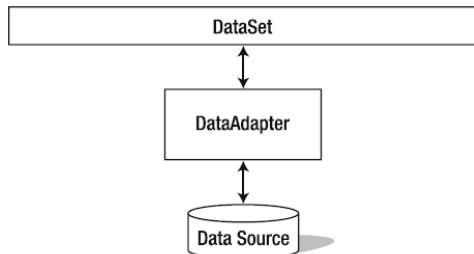
Arsitektur tersebut mencerminkan logika desain *relational database*. Anda akan melihat bagaimana cara menggunakan *data table*, *data row*, dan *data columns* pada bab ini, namun kita tidak membahas mengenai *relation constraint*.

4.4.3 Mengenal Data Adapter

Saat dataset dibuat sebagai object (instance) untuk kali pertama, dataset tidak memiliki data. Anda memperoleh datanya dengan cara melewatkannya ke data adapter, data adapter yang mengurusinya adalah connection. Dataset bukanlah bagian dari data adapter. Dataset seperti ember yang siap untuk diisi dengan air, tetapi dataset membutuhkan pipa dari luar untuk mengalirkan air ke dalamnya. Dengan kata lain, dataset membutuhkan data adapter untuk memberinya data dan mendukung pengaksesan ke data source (database MySQL).

Setiap *data provider* mempunyai data adapter-nya masing-masing, termasuk di dalamnya adalah connection, command, dan data reader

masing-masing pula. Bagan di bawah ini menggambarkan hubungan antara dataset data adapter dan data source.



Anda dapat menggunakan dan membuat data adapter baru dengan berbagai cara seperti daftar di bawah ini. Pada contoh di bawah, data source yang digunakan adalah MySQL dan SQL Server. Contoh pembuatan data adapter pada MySQL:

```
Dim da As MySqlDataAdapter = New MySqlDataAdapter()
Dim da As MySqlDataAdapter = New MySqlDataAdapter(cmd)
Dim da As MySqlDataAdapter = New MySqlDataAdapter(Sql, conn)
Dim da As MySqlDataAdapter = New MySqlDataAdapter(Sql, connString)
```

Membuat data adapter pada SQL Server:

```
Dim da As SqlDataAdapter = New SqlDataAdapter()
Dim da As SqlDataAdapter = New SqlDataAdapter(cmd)
Dim da As SqlDataAdapter = New SqlDataAdapter(Sql, conn)
Dim da As SqlDataAdapter = New SqlDataAdapter(Sql, connString)
```

Jadi, Anda dapat membuat data adapter dalam empat cara:

- Menggunakan tanpa parameter constructor (pencantuman SQL dan Connection dilakukan diwaktu yang lain).
- Dengan melewaskan parameter constructor berupa command (disini, cmd adalah object MySqlCommand atau SqlCommand).
- Dengan melewaskan parameter constructor berupa string SQL dan connection.

- Dengan melewatkkan parameter constructor berupa string SQL dan string Connection.

4.4.4 Data Table, Data Kolom, dan Data Baris

Data table secara konseptual dapat disamakan sebagai *relational table*. Data table mempunyai koleksi data baris (*data rows*), data kolom (*data columns*), dan constraint (tidak dibahas di sini). Anda dapat mengakses data table dan data kolom secara berulang-ulang melalui property baris dan kolom dari data table.

Data table – Data table dapat mewakili sebagai table bebas yang berdiri sendiri, sama seperti table dalam dataset.

Data kolom – Data kolom mewakili schema kolom yang berada dalam data table, selanjutnya dapat digunakan untuk mengisi dan memperoleh properti kolom. Anda bisa mendapatkan koleksi data kolom dengan menggunakan properti kolom dari data table yang diakses melalui nomor index atau nama kolom, contoh (**dt** adalah data table):

```
Dim col As DataColumn = dt.Columns("nama_penulis")
Dim col As DataColumn = dt.Columns(2)
```

Data baris – Data baris mewakili data dalam baris. Secara program anda dapat menambah, meng-update, atau menghapus baris dalam data table. Untuk mengakses baris-baris dalam data table gunakan properti baris berdasarkan nomor index, contoh (**dt** adalah data table):

```
Dim row as DataRow = dt.Rows(2)
```

4.4.5 Bekerja Dengan Dataset dan Data Adapter

Pembentukan dataset:

```
Dim ds As DataSet = New DataSet()
```

```
Dim ds As DataSet = New DataSet("MyDataSet")
```

Ada beberapa cara untuk mengisi dataset, diantaranya dengan menggunakan data adapter dan dari pembacaan dokumen XML. Buku ini hanya membahas pengisian dataset melalui data adapter saja, tidak menyinggung pengisian dataset melalui pembacaan dokumen XML. Sekarang mari kita buat program yang mengisi dataset dengan data adapter:

- Buat Project baru bernama **PopDataSet** dengan mengeklik menu **File->New->Project**.
- Template pilih dengan **Console Application** hingga terbentuk file **Module1.vb**.
- Tambahkan reference **MySQL.Data** pada menu **Project->Add Reference->.NET->MySQL.Data**.

PopDataSet: Module1.vb

```
Imports MySql.Data.MySqlClient

Module Module1

    Sub Main()
        Dim connString, db, user, pwd As String

        db = "dbpenjualanbuku" 'database
        user = "root" 'user default
        pwd = "" 'password kosong
        connString = "Database=" & db & ";Data Source=localhost;User
Id=" & user & ";Password=" & pwd

        'set up query
        Dim sql As String
        sql = "select judul, harga_jual as 'Harga' "
        sql = sql & "from buku_tb "
        sql = sql & "where harga_jual < 35000"

        'Membuat connection
        Dim conn As MySqlConnection = New MySqlConnection(connString)

        Try
            'membuka connection
            conn.Open()
```

```

        'membuat data adapter
        Dim da As MySqlDataAdapter = New MySqlDataAdapter(sql,
conn)

        'membuat dataset
        Dim ds As DataSet = New DataSet()

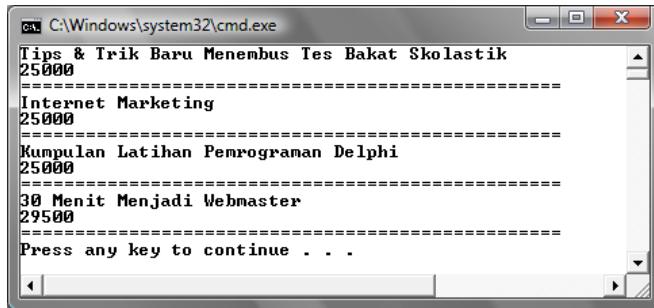
        'mengisi dataset
        da.Fill(ds, "buku_tb")

        'Memperoleh data table
        Dim dt As DataTable = ds.Tables("buku_tb")

        'mengulang isi data table
        For Each baris As DataRow In dt.Rows
            For Each kolom As DataColumn In dt.Columns
                Console.WriteLine(baris(kolom))
            Next
            Console.WriteLine("".PadLeft(50, "="))
        Next
    Catch ex As Exception
        'menampilkan pesan kesalahan
        Console.WriteLine("Error: " & ex.ToString)
    Finally
        'menutup connection
        conn.Close()
    End Try
End Sub
End Module

```

Jalankan program dengan menekan tombol keyboard **Ctrl+F5**, hasilnya kurang lebih seperti berikut ini:



Setelah membuat query dan membuka connection, selanjutnya adalah membuat dan memberi nilai awal pada data adapter:

```
'membuat data adapter
Dim da As MySqlDataAdapter = New MySqlDataAdapter(sql, conn)
```

Kemudian membuat dataset:

```
'membuat dataset
Dim ds As DataSet = New DataSet()
```

Pada tahap ini dataset masih kosong. Method **Fill** pada data adapter adalah untuk mengeksekusi query, menelusuri data, dan menempatkannya pada dataset:

```
'mengisi dataset
da.Fill(ds, "buku_tb")
```

Method **Fill** menggunakan data reader secara internal untuk mengakses *schema table* dan data, kemudian menempatkannya pada dataset. Setelah informasi menempati dataset, sekarang Anda dapat mengakses data masing-masing dari dalam data table.

```
'Memperoleh data table
Dim dt As DataTable = ds.Tables("buku_tb")
```

Terakhir, Anda menggunakan perulangan **For Each** untuk mengakses kolom pada setiap baris dan mengeluarkan nilai datanya pada layar:

```
'mengulang isi data table
For Each baris As DataRow In dt.Rows
    For Each kolom As DataColumn In dt.Columns
        Console.WriteLine(baris(kolom))
    Next
    Console.WriteLine("".PadLeft(50, "="))
Next
```

4.4.6 Penyaringan Dan Pengurutan Data Dalam Dataset

Kadang kita memerlukan penyaringan dan pengurutan terhadap data yang diperoleh. Berikut ini contoh untuk menampilkan baris dan kolom dari table **penulis_tb**, kemudian menyaring hasilnya dengan syarat harga jual lebih dari 25000 dan diurutkan berdasarkan kolom **judul_buku**.

- Buat Project baru bernama **FilterSort** dengan mengeklik menu **File->New->Project**.
- Template pilih dengan **Console Application** hingga terbentuk file **Module1.vb**.
- Tambahkan reference **MySQL.Data** di menu **Project->Add Reference->.NET->MySQL.Data**.

FilterSort: Module1.vb

```
Imports MySql.Data.MySqlClient

Module Module1

    Sub Main()
        Dim connString, db, user, pwd As String

        db = "dbpenjualanbuku" 'database
        user = "root" 'user default
        pwd = "" 'password kosong
        connString = "Database=" & db & ";Data Source=localhost;User
Id=" & user & ";Password=" & pwd

        'set up query pertama
        Dim sql1 As String = "select * from buku_tb; "

        'set up query kedua
        Dim sql2 As String = "select * from penulis_tb where
nama_penulis like 'B%'; "

        'kombinasi query
        Dim sql As String = sql1 & sql2

        'Membuat connection
        Dim conn As MySqlConnection = New MySqlConnection(connString)
```

```

Try
    'membuka connection
    conn.Open()

    'membuat data adapter
    Dim da As MySqlDataAdapter = New MySqlDataAdapter(sql,
conn)

    'membuat dataset
    Dim ds As DataSet = New DataSet()
    'mengisi dataset
    da.Fill(ds, "buku_tb")

    'mendapatkan data tables collection
    Dim dtc As DataTableCollection = ds.Tables

    'menampilkan data dari table pertama
    'menampilkan header
    Console.WriteLine("Hasil dari table buku_tb:")
    Console.WriteLine("Judul".PadRight(50) &
"Harga".PadLeft(6))

    'atur penyaringan data
    Dim fl As String = "harga_jual <= 30000"

    'atur pengurutan
    Dim srt As String = "judul desc"
    '
    'tampilkan data yang sudah tersaring dan terurut
    For Each row As DataRow In dtc("buku_tb").Select(fl, srt)
        Console.WriteLine("{0} {1}",
row("judul").ToString().PadRight(50), _
                    row("harga_jual"))
    Next

    'tampilkan data dari table kedua
    'tampilkan header
    Console.WriteLine("-----")
    Console.WriteLine("Hasil dari table penulis_tb:")
    Console.WriteLine("Kode Penulis".PadRight(15) & "Nama
Penulis")
    '

    ' Display data
    For Each row As DataRow In dtc(1).Rows
        Console.WriteLine("{0} {1}",
row("kode_penulis").ToString().PadRight(15), _
                    row("nama_penulis"))
    Next
Catch ex As Exception
    'menampilkan pesan kesalahan
    Console.WriteLine("Error: " & ex.ToString)
Finally

```

```
        'menutup connection
        conn.Close()
    End Try
End Sub
End Module
```

Buat penggabungan dua query untuk dieksekusi pada connection yang sama:

```
'set up query pertama
Dim sql1 As String = "select * from buku_tb; "

'set up query kedua
Dim sql2 As String = "select * from penulis_tb where nama_penulis
like 'B%'; "

'kombinasi query
Dim sql As String = sql1 & sql2
```

Membuat data adapter, pemberian nilai pada property command **SelectCommand** yang membungkus query dan connection (digunakan secara internal oleh method **Fill** data adapter):

```
'membuat data adapter
Dim da As MySqlDataAdapter = New MySqlDataAdapter(sql, conn)
```

Kemudian mengisi dataset:

```
'membuat dataset
Dim ds As DataSet = New DataSet()
'mengisi dataset
da.Fill(ds, "buku_tb")
```

Setiap query akan mengembalikan serangkaian hasil (*result set*) dan setiap *result set* tersimpan dalam data table terpisah. Table pertama secara eksplisit bernama **buku_tb**; table kedua diberikan secara default bernama **buku_tb1**. Mendapatkan *data table collection* dari property dataset **Tables**:

```
'mendapatkan data tables collection
Dim dtc As DataTableCollection = ds.Tables
```

Pada bagian ini akan menampilkan data table pertama. Anda mendeklarasikan dua string:

```
'atur penyaringan data
Dim fl As String = "harga_jual <= 30000"

'atur pengurutan
Dim srt As String = "judul desc"
```

String pertama adalah ekspresi penyaringan baris yang dipilih sesuai dengan kriteria, penulisan kode yang digunakan sama dengan klausa **WHERE** pada SQL, menampilkan baris-baris data dimana harga jual kurang dari 30000. String kedua merupakan penentuan urutan baris data, penulisan kode yang digunakan sama dengan klausa **ORDER BY** pada SQL, diurutkan **DESC** (dari besar ke kecil) berdasarkan judul buku. Gunakan perulangan **For Each** untuk menampilkan baris-baris terpilih dari data table, melewatkannya penyaringan dan pengurutan ke method **Select** dari data table. Untuk saat ini data table-nya bernama **buku_tb** dalam *data table collection*.

```
'tampilkan data yang sudah tersaring dan terurut
For Each row As DataRow In dtc("buku_tb").Select(fl, srt)
    Console.WriteLine("{0} {1}", row("judul").ToString().PadRight(50),
    row("harga_jual"))
Next
```

Anda merujuk ke satu data table dari *data table collection* (object **dtc**) dengan menggunakan nama table yang telah ditentukan saat pembuatan dataset. Method **Select** bekerja secara internal untuk mencari baris pada data table, membuang baris yang tidak sesuai dengan kriteria, mengurutkan hasil, dan terakhir mengembalikan baris dalam bentuk data array. Akses setiap kolom dalam baris data berdasarkan nama kolom sesuai nomor index-nya. Perulangan yang terjadi pada data table kedua:

```
For Each row As DataRow In dtc(1).Rows
```

dtc(1) merujuk ke table kedua, jika index bernilai 0 maka nama table yang dirujuk adalah **buku_tb** dan jika bernilai 1 maka secara default nama table-nya adalah **buku_tb1** sehingga pernyataan di atas bisa diubah menjadi:

```
For Each row As DataRow In dtc("buku_tb1").Rows
```

4.4.7 Data View

Pada contoh sebelumnya Anda telah melihat penyaringan dan pengurutan data table secara dinamis dengan method **SELECT**. ADO.NET memiliki pendekatan lain untuk melakukan hal yang sama namun lebih baik yaitu *data view*. Dengan *data view* Anda dapat membuat *view* dinamis dari data yang tersimpan dalam data table, mencerminkan seluruh perubahan yang pernah dibuat baik isi maupun urutannya. Berbeda dengan method **SELECT** yang mengembalikan array baris data dimana isinya mencerminkan perubahan nilai data tetapi bukan urutan datanya.

Catatan: data view mewakili isi data table secara dinamis. Seperti view SQL, data view tidak memegang data dalam arti sesungguhnya.

Sekarang buat program yang akan menyaring data dengan data view.

- Buat Project baru bernama **DataViews** dengan mengeklik menu **File->New->Project**.
- Template pilih dengan **Console Application** hingga terbentuk file **Module1.vb**.
- Tambahkan reference **MySQL.Data** pada menu **Project->Add Reference->.NET->MySQL.Data**.

DataViews: Module1.vb

```
Imports MySql.Data.MySqlClient

Module Module1

    Sub Main()
        Dim connString, db, user, pwd As String
```

```

db = "dbpenjualanbuku" 'database
user = "root" 'user default
pwd = "" 'password kosong
connString = "Database=" & db & ";Data Source=localhost;User
Id=" & user & ";Password=" & pwd

'set up query pertama
Dim sql As String = "select kode_buku, judul, harga_jual from
buku_tb; "

'Membuat connection
Dim conn As MySqlConnection = New MySqlConnection(connString)

Try
    'membuka connection
    conn.Open()

    'membuat data adapter
    Dim da As MySqlDataAdapter = New MySqlDataAdapter(sql,
conn)

    'membuat dataset
    Dim ds As DataSet = New DataSet()
    'mengisi dataset
    da.Fill(ds, "buku_tb")

    'mendapatkan rujukan ke data table
    Dim dt As DataTable = ds.Tables("buku_tb")

    'membuat data view
    Dim dv As DataView = New DataView(dt, "harga_jual <
30000", _
"judul desc", DataViewRowState.CurrentRows)

    'menampilkan data dari data view
    For Each drv As DataRowView In dv
        'For i As Integer = 0 To dv.Table.Columns.Count - 1
        'Console.WriteLine(drv(i).ToString().PadRight(30))
        'Next

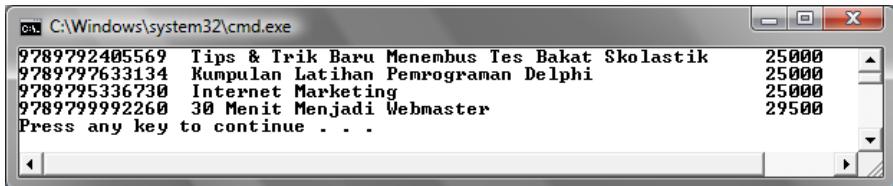
        Console.WriteLine(drv(0).ToString().PadRight(15))
        Console.WriteLine(drv(1).ToString().PadRight(50))
        Console.WriteLine(drv(2))
        Console.WriteLine()

    Next
    Catch ex As Exception
        'menampilkan pesan kesalahan
        Console.WriteLine("Error: " & ex.ToString())
    Finally
        'menutup connection
        conn.Close()
    End Try

```

```
End Sub  
End Module
```

Jalankan programnya dengan menekan tombol **Ctrl+F5**, hasilnya kurang lebih seperti berikut ini:



The screenshot shows a Windows command prompt window titled 'cmd C:\Windows\system32\cmd.exe'. Inside the window, there is a data grid view displaying four rows of data from a table. The columns are labeled: 'Kode', 'Judul', 'Pembelaan', and 'Harga Jual'. The data is as follows:

Kode	Judul	Pembelaan	Harga Jual
9789792405569	Tips & Trik Baru Menembus Tes Bakat Skolastik	25000	
9789797633134	Kumpulan Latihan Pemrograman Delphi	25000	
9789795336730	Internet Marketing	25000	

Below the grid, the message 'Press any key to continue . . .' is displayed.

Program dasarnya sama dengan contoh lainnya, kita menitikberatkan pada penggunaan data view. Anda membuat data view baru dan memberi nilai awal dengan melewatkannya empat parameter ke *constructor* data view:

```
'membuat data view  
Dim dv As DataView = New DataView(dt, "harga_jual < 30000", _  
"judul desc", DataGridViewRowState.CurrentRows)
```

Parameter pertama adalah data table, parameter kedua menyaring isi data table, paramater ketiga adalah pengurutan kolom (asc atau desc), parameter keempat penentuan jenis baris data yang dimasukkan ke data view.

System.Data.DataViewRowState adalah daftar pernyataan baris data view yang mendasari data table. Berikut ini daftar property DaftarRowViewState:

Anggota DataViewRowState	Keterangan
Added	Baris baru
CurrentRows	Baris saat ini, termasuk tanpa perubahan, baru, dan yang termodifikasi.
Deleted	Baris yang dihapus
ModifiedCurrent	Versi saat ini dari baris yang telah dimodifikasi.

ModifiedOriginal	Versi asli dari baris yang telah dimodifikasi.
None	Tidak terdapat baris.
OriginalRows	Baris asli, termasuk yang tak berubah dan yang terhapus.
Unchanged	Baris yang tidak mengalami perubahan.

Setiap kali baris ditambahkan, dimodifikasi, atau dihapus, maka pernyataan (*state*) pada baris bersangkutan akan berubah sesuai dengan salah satu daftar table di atas. Hal ini berguna ketika Anda terlibat dengan proses penelusuran, pengurutan dan penyaringan baris-baris data tertentu didasarkan pada *state*-nya masing-masing. Untuk menampilkan baris-baris data dalam data view bisa dengan cara perulangan:

```
'menampilkan data dari data view
For Each drv As DataRowView In dv
    'For i As Integer = 0 To dv.Table.Columns.Count - 1
    'Console.WriteLine(drv(i).ToString().PadRight(30))
    'Next

    Console.WriteLine(drv(0).ToString().PadRight(15))
    Console.WriteLine(drv(1).ToString().PadRight(50))
    Console.WriteLine(drv(2))
    Console.WriteLine()
Next
```

4.4.8 Memodifikasi Data Dalam Dataset

Berikut ini contoh mengubah data dalam dataset tetapi belum mengubah data dalam database.

- Buat Project baru bernama **ModifyDataTable** dengan mengeklik menu **File->New->Project**.
- Template pilih dengan **Console Application** hingga terbentuk file **Module1.vb**.
- Tambahkan reference **MySQL.Data** pada menu **Project->Add Reference->.NET->MySQL.Data**.

ModifyDataTable: Module1.vb

```
Imports MySql.Data.MySqlClient

Module Module1

    Sub Main()
        Dim connString, db, user, pwd As String

        db = "dbpenjualanbuku" 'database
        user = "root" 'user default
        pwd = "" 'password kosong
        connString = "Database=" & db & ";Data Source=localhost;User
Id=" & user & ";Password=" & pwd

        'set up query
        Dim sql As String
        sql = "select kode_penerbit, nama_penerbit, alamat,
        sql &= "kota, kode_pos, telp, email, website from penerbit_tb
        "

        'Membuat connection
        Dim conn As MySqlConnection = New MySqlConnection(connString)

        Try
            'membuka connection
            conn.Open()

            'membuat data adapter
            Dim da As MySqlDataAdapter = New MySqlDataAdapter(sql,
conn)

            'membuat dan mengisi dataset
            Dim ds As DataSet = New DataSet()
            da.Fill(ds, "penerbit_tb")

            'mendapatkan referensi table
            Dim dt As DataTable = ds.Tables("penerbit_tb")

            'perubahan schema, kolom nama_penulis boleh null
            dt.Columns("nama_penerbit").AllowDBNull = True

            'memodifikasi email pada baris pertama
            dt.Rows(0)("nama_penerbit") = "New Penerbit"

            'menambahkan baris
            Dim newRow As DataRow = dt.NewRow()
            newRow("nama_penerbit") = "Joe Satriani"
            newRow("alamat") = "Sartono SH 4"
            newRow("kota") = "Malang"
            newRow("kode_pos") = "65118"
            newRow("telp") = "0341-356781"
            newRow("email") = "joe@yahoo.com"
```

```

newRow("website") = "www.joes.com"
dt.Rows.Add(newRow)

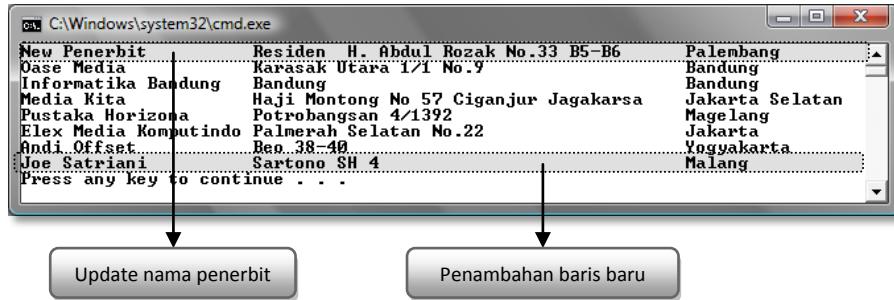
'menampilkan baris di dalam data table
For Each row As DataRow In dt.Rows
    Console.WriteLine(
        "{0} {1} {2}",
        row("nama_penerbit").ToString().PadRight(21), _
        row("alamat").ToString().PadRight(40), _
        row("kota"))
Next

'
'kode untuk mengubahnya ke database berada disini
'

Catch ex As Exception
    'menampilkan pesan kesalahan
    Console.WriteLine("Error: " & ex.ToString)
Finally
    'menutup connection
    conn.Close()
End Try
End Sub
End Module

```

Jalankan programnya dengan menekan tombol keyboard **Ctrl+F5**, hasilnya kurang lebih seperti berikut ini:



Perubahan data di atas belum benar-benar mengubah data pada database, jika dibandingkan antara data asli dengan data dalam data source pasti hasilnya berbeda.

nama_penerbit	alamat	kota
Maxikom CV.	Residen H. Abdul Rozak No.33 B5-B6	Palembang
Oase Media	Karasak Utara 1/1 No.9	Bandung
Informatika Bandung	Bandung	Bandung
Media Kita	Haji Montong No 57 Ciganjur Jagakarsa	Jakarta Selatan
Pustaka Horizone	Potrobagongan 4/1392	Magelang
Elex Media Komputindo	Palmerah Selatan No. 22	Jakarta
Andi Offset	Beo 38-40	Yogyakarta

Gunakan satu data table dalam dataset:

```
'mendapatkan referensi table
Dim dt As DataTable = ds.Tables("penerbit_tb")
```

Terdapat perubahan informasi *schema*, yaitu kolom **nama_penerbit** boleh bernilai **Null (AllowDBNull = True)** padahal pada kenyataannya dalam database kolom **nama_penerbit** telah diatur tidak boleh **Null**. Kondisi **AllowDBNull** kurang bermanfaat, hanya sekedar mendemonstrasikan bahwa dengan data table Anda dapat melakukan perubahan informasi *schema*.

```
'perubahan schema, kolom nama_penulis boleh null
dt.Columns("nama_penerbit").AllowDBNull = True
```

Anda dapat memodifikasi baris dengan teknik yang sama. Pilih baris yang dikehendaki dan isi kolom dengan nilai yang diinginkan, namun tetap mempertimbangkan kesesuaian tipe data kolomnya. Berikut ini bagaimana merubah kolom **nama_penerbit** yang semula adalah **Maxikom.CV** menjadi **New Penerbit**.

```
'memodifikasi email pada baris pertama
dt.Rows(0)("nama_penerbit") = "New Penerbit"
```

Kemudian Anda menambahkan baris baru ke data table:

```
'menambahkan baris
Dim newRow As DataRow = dt.NewRow()
newRow("nama_penerbit") = "Joe Satriani"
newRow("alamat") = "Sartono SH 4"
newRow("kota") = "Malang"
newRow("kode_pos") = "65118"
newRow("telp") = "0341-356781"
newRow("email") = "joe@yahoo.com"
newRow("website") = "www.joes.com"
dt.Rows.Add(newRow)
```

Method **NewRow** adalah untuk membuat data row (object **System.Data.DataRow**). Gunakan index data row untuk memberi nilai pada setiap kolom. Tambahkan baris baru ke data table melalui pemanggilan method **Add** pada property data table **Rows** yang merujuk ke **rows collection**.

4.4.9 Melimpahkan Perubahan Ke Data Source

Ada tiga jenis *property* yang mendukung proses perubahan dan penyamaan data antara *data source* dengan data yang berasal dari *dataset*, antara lain: **UpdateCommand**, **InsertCommand** dan **DeleteCommand**.

4.4.9.1 UpdateCommand

Property data adapter **UpdateCommand** mengendalikan command yang akan mengubah data source saat method data adapter **Update** dipanggil. Sebagai contoh buatlah program yang mengubah kolom **website** dari table **penerbit_tb** dan menetapkan perubahannya ke database:

- Buat Project baru bernama **PersistChanges** dengan mengeklik menu **File->New->Project**
- Template pilih dengan **Console Application** hingga terbentuk file **Module1.vb**.

- Tambahkan reference MySQL.Data pada menu Project->Add Reference->.NET->MySQL.Data.

PersistChanges: Module1.vb

```

Imports MySql.Data.MySqlClient

Module Module1

    Sub Main()
        Dim connString, db, user, pwd As String

        db = "dbpenjualanbuku" 'database
        user = "root" 'user default
        pwd = "" 'password kosong
        connString = "Database=" & db & ";Data Source=localhost;User
Id=" & user & ";Password=" & pwd

        'set up query
        Dim sql As String = "select * from penerbit_tb;"

        'set up DML
        Dim upd As String
        upd = "update penerbit_tb set website = @website "
        upd &= "where kode_penerbit = @kode_penerbit;"

        'Membuat connection
        Dim conn As MySqlConnection = New MySqlConnection(connString)

        Try
            'membuka connection
            conn.Open()

            'membuat data adapter
            Dim da As MySqlDataAdapter = New MySqlDataAdapter(sql,
conn)

            'membuat dan mengisi dataset
            Dim ds As DataSet = New DataSet()
            da.Fill(ds, "penerbit_tb")

            'mendapatkan referensi table
            Dim dt As DataTable = ds.Tables("penerbit_tb")

            'memodifikasi kolom website di baris kedua
            dt.Rows(1)("website") = "www.oasemedia.com"

            'menampilkan baris-baris di dalam data table
            For Each row As DataRow In dt.Rows
                Console.WriteLine("{0} {1} {2}",
row("kode_penerbit").ToString().PadRight(10),
row("nama_penerbit").ToString().PadLeft(30),

```

```

        row("website"))
    Next

    'Update penerbit_tb di data source
    '
    'membuat command
    Dim cmd As MySqlCommand = New MySqlCommand(upd, conn)
    '
    'memetakan parameter
    '
    'Website
    cmd.Parameters.Add("@website", MySqlDbType.TinyText, 100,
"website")
    '

    'kode_penerbit
    Dim parm As MySqlParameter =
cmd.Parameters.Add("@kode_penerbit",
MySqlDbType.VarChar, 10, "kode_penerbit")
    parm.SourceVersion = DataRowVersion.Original
    '

    'Update database
    da.UpdateCommand = cmd
    da.Update(ds, "penerbit_tb")
    Catch ex As Exception
        'menampilkan pesan kesalahan
        Console.WriteLine("Error: " & ex.ToString)
    Finally
        'menutup connection
        conn.Close()
    End Try
End Sub
End Module

```

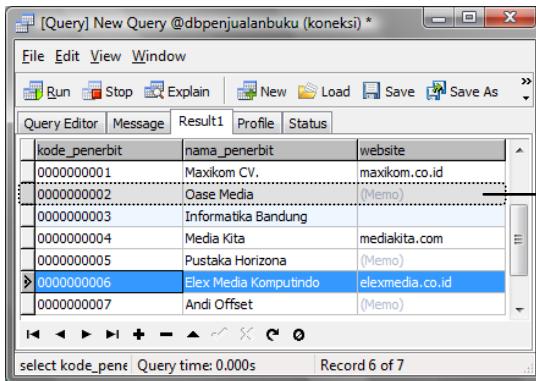
Jalankan programnya dengan menekan tombol keyboard **Ctrl+F5**, hasilnya kurang lebih seperti berikut ini:

ID	Penerbit	Alamat
0000000001	Maxikom CU.	maxikom.co.id
0000000002	Oase Media	www.oasemedia.com
0000000003	Informatika Bandung	
0000000004	Media Kita	mediakita.com
0000000005	Pustaka Horizonta	
0000000006	Elex Media Komputindo	elexmedia.co.id
0000000007	Andi Offset	

Press any key to continue . . .

Hasil perubahan di atas merupakan tampilan yang diperoleh dari data table, hasil tersebut juga berlaku sama dengan hasil pada data source

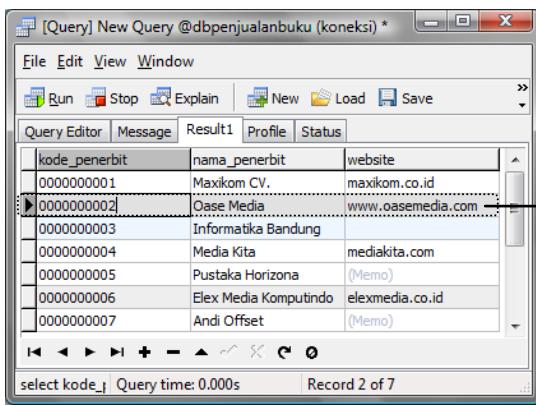
(database MySQL). Gambar di bawah ini menunjukkan baris data sebelum mengalami perubahan:



kode_penerbit	nama_penerbit	website
0000000001	Maxikom CV.	mavikom.co.id
0000000002	Oase Media	(Memo)
0000000003	Informatika Bandung	
0000000004	Media Kita	mediakita.com
0000000005	Pustaka Horizone	(Memo)
0000000006	Elex Media Komputindo	elexmedia.co.id
0000000007	Andi Offset	(Memo)

Kolom website masih kosong.

Baris data pada data source setelah mengalami perubahan:



kode_penerbit	nama_penerbit	website
0000000001	Maxikom CV.	mavikom.co.id
0000000002	Oase Media	www.oasemedia.com
0000000003	Informatika Bandung	
0000000004	Media Kita	mediakita.com
0000000005	Pustaka Horizone	(Memo)
0000000006	Elex Media Komputindo	www.elexmedia.co.id
0000000007	Andi Offset	(Memo)

Kolom website pada data source telah berubah menjadi www.oasemedia.com

Anda menambahkan pernyataan **UPDATE** dan mengubah nama variable query string asli dari **sql** ke **upd**.

```
'set up DML
Dim upd As String
upd = "update penerbit_tb set website = @website "
upd &= "where kode_penerbit = @kode_penerbit;"
```

Membuat command berisi perintah yang berasal dari variable SQL update (**upd**), bukan query sebelumnya (**sql**).

```
'Update penerbit tb di data source
'
'membuat command
Dim cmd As MySqlCommand = New MySqlCommand(upd, conn)
```

Mengkonfigurasi parameter *command (CMD)*. Parameter **@website** dipetakan ke kolom **website**. Karena yang digunakan adalah data source, bukan data table, maka pastikan bahwa tipe dan lebar kolom sesuai dengan kolom website pada database.

```
'Website
cmd.Parameters.Add("@website", MySqlDbType.TinyText, 100, "website")
```

Anda telah mengkonfigurasi parameter **@kode_penerbit**, memetakkannya ke kolom data **kode_penerbit**. Tidak seperti **@website** yang secara *default* mengambil nilai data table versi saat ini, **@kode_penerbit** memperoleh nilai dari versi sebelum perubahan terjadi. Meskipun hal ini tidak berpengaruh selama tidak mengubah kolom **kode_penerbit**, namun biasakan untuk menetapkan versi asli bagi *primary key*. Jadi, jika **kode_penerbit** tidak berubah maka baris saat ini diakses dari table database.

```
'kode_penerbit
Dim parm As MySqlParameter = cmd.Parameters.Add("@kode_penerbit", _
MySqlDbType.VarChar, 10, "kode_penerbit")
parm.SourceVersion = DataRowVersion.Original
```

Pemberian nilai property data adapter **UpdateCommand** dengan *command (CMD)* untuk mengubah table **penerbit_tb**, jadi SQL akan dieksekusi oleh adapter ketika method **Update** dipanggil. Kemudian memanggil **Update** data adapter untuk melimpahkan perubahannya ke database.

```
'Update database
da.UpdateCommand = cmd
da.Update(ds, "penerbit_tb")
```

4.4.9.2 Property InsertCommand

Data adapter menggunakan property **InsertCommand** untuk menambahkan baris data ke table. Melalui pemanggilan method **Update**, semua baris yang ditambahkan ke data table akan dicari dan dilimpahkan ke database.

- Buat Project baru bernama **PersistAdds** dengan mengeklik menu **File->New->Project**.
- Template pilih dengan **Console Application** hingga terbentuk file **Module1.vb**.
- Tambahkan reference **MySQL.Data** pada menu **Project->Add Reference->.NET->MySQL.Data**.

PersistAdds: Module1.vb

```
Imports MySql.Data.MySqlClient

Module Module1

    Sub Main()
        Dim connString, db, user, pwd As String

        db = "dbpenjualanbuku" 'database
        user = "root" 'user default
        pwd = "" 'password kosong
        connString = "Database=" & db & ";Data Source=localhost;User
Id=" & user & ";Password=" & pwd

        'set up query
        Dim sql As String = "select * from penerbit_tb;"

        'set up DML
        Dim ins As String
        ins = "insert into penerbit_tb "
        ins &= "(kode_penerbit, nama_penerbit, alamat, kota, "
        ins &= "kode_pos, telp, email, website) "
        ins &= "values (@kode_penerbit, @nama_penerbit, @alamat,
@kota, "
        ins &= "@kode_pos, @telp, @email, @website);"

        'Membuat connection
        Dim conn As MySqlConnection = New MySqlConnection(connString)
```

```

Try
    'membuka connection
    conn.Open()

    'membuat data adapter
    Dim da As MySqlDataAdapter = New MySqlDataAdapter(sql,
conn)

    'membuat dan mengisi dataset
    Dim ds As DataSet = New DataSet()
    da.Fill(ds, "penerbit_tb")

    'mendapatkan referensi table
    Dim dt As DataTable = ds.Tables("penerbit_tb")

    'menambah baris
    Dim newRow As DataRow = dt.NewRow()
    newRow("kode_penerbit") = "0000000008"
    newRow("nama_penerbit") = "Penerbit Baru"
    newRow("alamat") = "Jl. Sartono Sh 4"
    newRow("kota") = "Malang"
    newRow("kode_pos") = "65118"
    newRow("telp") = "(0341) 356781"
    newRow("email") = "d0dit@yahoo.com"
    newRow("website") = "www.newpenerbit.com"
    dt.Rows.Add(newRow)

    'Menampilkan baris dari data table
    For Each row As DataRow In dt.Rows
        Console.WriteLine("{0} {1} {2}",
        row("kode_penerbit").ToString().PadRight(10), _
        row("nama_penerbit").ToString().PadLeft(30), _
        row("alamat"), row("kota"))
    Next

    'menambahkan penerbit
    'membuat command
    Dim cmd As MySqlCommand = New MySqlCommand(ins, conn)
    '
    'memetakan parameter
    '
    cmd.Parameters.Add("@kode_penerbit", MySqlDbType.VarChar,
10, "kode_penerbit")
    cmd.Parameters.Add("@nama_penerbit", MySqlDbType.VarChar,
30, "nama_penerbit")
    cmd.Parameters.Add("@alamat", MySqlDbType.VarChar, 50,
"alamat")
    cmd.Parameters.Add("@kota", MySqlDbType.VarChar, 25,
"kota")
    cmd.Parameters.Add("@kode_pos", MySqlDbType.VarChar, 6,
"kode_pos")
    cmd.Parameters.Add("@telp", MySqlDbType.VarChar, 15,
"telp")

```

```

        cmd.Parameters.Add("@email", MySqlDbType.TinyText, 100,
"email")
        cmd.Parameters.Add("@website", MySqlDbType.TinyText, 100,
"website")

        'Update database
        da.InsertCommand = cmd
        da.Update(ds, "penerbit_tb")
    Catch ex As Exception
        'menampilkan pesan kesalahan
        Console.WriteLine("Error: " & ex.ToString)
    Finally
        'menutup connection
        conn.Close()
    End Try
End Sub
End Module

```

Jalankan programnya dengan menekan tombol **Ctrl+F5**, hasilnya kurang lebih seperti berikut ini:

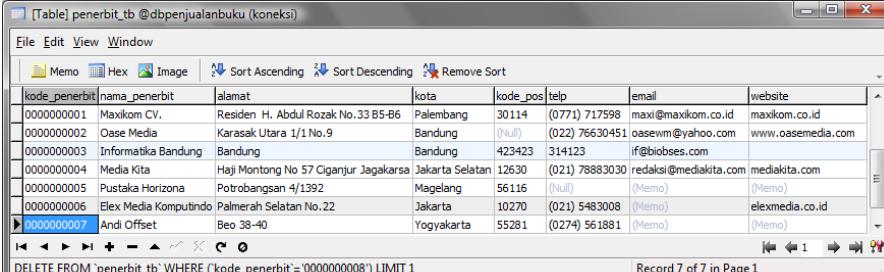


```

C:\Windows\system32\cmd.exe
0000000001      Maxikom CV. Residen H. Abdul Rozak No.33 B5-B6
0000000002          Oase Media Karasak Utara 1/1 No.9
0000000003      Informatika Bandung Bandung
0000000004          Media Kita Haji Montong No 57 Ciganjur Jagakarsa
0000000005      Pustaka Horizone Petrobangsar 4/1392
0000000006      Elex Media Komputindo Palmerah Selatan No.22
0000000007          Andi Offset Beo 38-40
0000000008      Penerbit Baru Jl. Sartono Sh 4
Press any key to continue . . .

```

Isi table dalam database sebelum mengalami penambahan baris:



	kode_penerbit	nama_penerbit	alamat	kota	kode_pos	telp	email	website
0000000001	Maxikom CV.	Residen H. Abdul Rozak No.33 B5-B6	Palembang	30114	(0771) 717598	maxi@maxikom.co.id	maxikom.co.id	
0000000002	Oase Media	Karasak Utara 1/1 No.9	Bandung	(Null)	(022) 76630451	oasewm@yahoo.com	www.oasemedia.com	
0000000003	Informatika Bandung	Bandung		423423	314123	(#)biobses.com		
0000000004	Media Kita	Haji Montong No 57 Ciganjur Jagakarsa	Jakarta Selatan	12630	(021) 78883030	redaksi@mediakita.com	mediakita.com	
0000000005	Pustaka Horizone	Petrobangsan 4/1392	Magelang	56116	(Null)	(Memo)	(Memo)	
0000000006	Elex Media Komputindo	Palmerah Selatan No.22	Jakarta	10270	(021) 5483008	(Memo)	elexmedia.co.id	
0000000007	Andi Offset	Beo 38-40	Yogyakarta	55281	(0274) 561881	(Memo)	(Memo)	

DELETE FROM 'penerbit_tb' WHERE (kode_penerbit='0000000008') LIMIT 1

Record 7 of 7 in Page 1

Isi table dalam database setelah mengalami penambahan baris:

[Table] penerbit_tb @dbpenjualanbuku (koneksi)

kode_penerbit	nama_penerbit	alamat	kota	kode_pos	telp	email	website	
0000000001	Maxikom CV.	Residen H. Abdul Rozak No.33 B5-B6	Palembang	30114	(071) 717598	maxi@maxikom.co.id	maxikom.co.id	
0000000002	Oase Media	Karasik Utara 1/1 No.9	Bandung	(Null)	(022) 76630451	oasewm@yahoo.com	www.oasemedia.com	
0000000003	Informatika Bandung	Bandung	Bandung	423423	314123	f@biobses.com		
0000000004	Media Kita	Haji Montong No 57 Ciganjur Jagakarsa	Jakarta Selatan	12630	(021) 78883030	redaksi@mediakita.com	mediakita.com	
0000000005	Pustaka Horison	Potrobangsari 4/1392	Magelang	56116	(Null)	(Memo)	(Memo)	
0000000006	Elex Media Komputindo	Palmerah Selatan No.22	Jakarta	10270	(021) 5483008	(Memo)	elexmedia.co.id	
0000000007	Andi Offset	Bdg 38-40	Yogyakarta	55281	(0274) 56.1881	(...)	(...)	
0000000008	Penerbit Baru	Jl. Sartono Sh 4	Malang	65118	(0341) 356781	ddit@yahoo.com	www.newpenerbit.com	

SELECT * FROM `penerbit_tb` LIMIT 0,1000

Record 1 of 8 in Page 1

Anda menambahkan pernyataan **INSERT** dan mengubah nama variable query string asli dari **sql** menjadi **ins**.

```
'set up DML
Dim ins As String
ins = "insert into penerbit_tb "
ins &= "(kode_penerbit, nama_penerbit, alamat, kota, "
ins &= "kode pos, telp, email, website) "
ins &= "values (@kode penerbit, @nama penerbit, @alamat, @kota, "
ins &= "@kode_pos, @telp, @email, @website);"
```

Membuat **command (CMD)**, tetapi yang digunakan adalah variable SQL **insert (ins)** bukan query sebelumnya (**sql**).

```
'menambahkan penerbit
'membuat command
Dim cmd As MySqlCommand = New MySqlCommand(ins, conn)
```

Mengkonfigurasi parameter **command (CMD)**. Terdapat delapan kolom, setiap kolom memperoleh nilai dari pemetaan paramater command. Saat ini Anda tidak menetapkan suatu *primary key* karena telah mengikuti struktur kolom pada MySQL dan tidak perlu pula menentukan kolom mana yang boleh **Null**.

```
'memetakan parameter
cmd.Parameters.Add("@kode_penerbit", MySqlDbType.VarChar, 10,
"kode_penerbit")
cmd.Parameters.Add("@nama_penerbit", MySqlDbType.VarChar, 30,
"nama_penerbit")
cmd.Parameters.Add("@alamat", MySqlDbType.VarChar, 50, "alamat")
cmd.Parameters.Add("@kota", MySqlDbType.VarChar, 25, "kota")
cmd.Parameters.Add("@kode_pos", MySqlDbType.VarChar, 6, "kode_pos")
```

```
cmd.Parameters.Add("@telp", MySqlDbType.VarChar, 15, "telp")
cmd.Parameters.Add("@email", MySqlDbType.TinyText, 100, "email")
cmd.Parameters.Add("@website", MySqlDbType.TinyText, 100, "website")
```

Pengaturan property data adapter **InsertCommand** dengan perintah **INSERT INTO** ke table **penerbit_tb**, sehingga SQL dijalankan oleh data adapter saat pemanggilan method **Update**. Selanjutnya **Update** terpanggil pada data adapter untuk melimpahkan perubahannya ke database.

```
'Update database
da.InsertCommand = cmd
da.Update(ds, "penerbit_tb")
```

4.4.9.3 Property DeleteCommand

Gunakan property data adapter **DeleteCommand** untuk menjalankan pernyataan SQL **DELETE**.

- Buat Project baru bernama **PersistDeletes** dengan mengeklik menu **File->New->Project**.
- Template pilih dengan **Console Application** hingga terbentuk file **Module1.vb**.
- Tambahkan reference **MySQL.Data** pada menu **Project->Add Reference->.NET->MySQL.Data**.

PersistDeletes: Module1.vb

```
Imports MySql.Data.MySqlClient

Module Module1

    Sub Main()
        Dim connString, db, user, pwd As String

        db = "dbpenjualanbuku" 'database
        user = "root" 'user default
        pwd = "" 'password kosong
        connString = "Database=" & db & ";Data Source=localhost;User
Id=" & user & ";Password=" & pwd

        'set up query
        Dim sql As String = "select * from penerbit_tb;"
```

```

'set up DML
Dim del As String
del = "delete from penerbit_tb "
del &= "where kode_penerbit = @kode_penerbit;"

'Membuat connection
Dim conn As MySqlConnection = New MySqlConnection(connString)

Try
    'membuka connection
    conn.Open()

    'membuat data adapter
    Dim da As MySqlDataAdapter = New MySqlDataAdapter(sql,
conn)

    'membuat dan mengisi dataset
    Dim ds As DataSet = New DataSet()
    da.Fill(ds, "penerbit_tb")

    'mendapatkan referensi table
    Dim dt As DataTable = ds.Tables("penerbit_tb")

    'menghapus penerbit_tb
    '
    'membuat command
    Dim cmd As MySqlCommand = New MySqlCommand(del, conn)

    'memetakan parameter
    cmd.Parameters.Add("@kode_penerbit", MySqlDbType.VarChar,
10, "kode_penerbit")

    'memilih penerbit_tb untuk dihapus
    Dim filt As String = "kode_penerbit = '0000000008' " -
& "and nama_penerbit = 'Penerbit Baru' "
    '
    'menghapus penerbit_tb dari data table
    For Each row As DataRow In dt.Select(filt)
        row.Delete()
    Next

    'Update database
    da.DeleteCommand = cmd
    da.Update(ds, "penerbit_tb")

    'menampilkan data table
    For Each row As DataRow In dt.Rows
        Console.WriteLine("{0} {1} {2}", _
        row("kode_penerbit").ToString().PadRight(10), _
        row("nama_penerbit").ToString().PadLeft(30), _
        row("alamat"))
    Next

```

```

    Catch ex As Exception
        'menampilkan pesan kesalahan
        Console.WriteLine("Error: " & ex.ToString)
    Finally
        'menutup connection
        conn.Close()
    End Try

End Sub

End Module

```

Jalankan programnya dengan menekan tombol **Ctrl+F5**, dan hasilnya kurang lebih seperti tampak gambar berikut ini:



Anda menambahkan pernyataan **DELETE** dan mengubah nama query string asli dari **sql** ke **del**.

```

'set up DML
Dim del As String
del = "delete from penerbit_tb "
del &= "where kode_penerbit = @kode_penerbit;"

```

Menambahkan kode penghapusan berupa query string **del**, kemudian membuat command dan memetakan parameternya

```

'menghapus penerbit_tb
'
'membuat command
Dim cmd As MySqlCommand = New MySqlCommand(del, conn)

'memetakan parameter
cmd.Parameters.Add("@kode_penerbit", MySqlDbType.VarChar, 10,
"kode_penerbit")

```

Memilih baris data untuk dihapus berdasarkan kriteria yang ditetapkan

```
'memilih penerbit tb untuk dihapus
Dim filt As String = "kode_penerbit = '0000000008' "
& "and nama_penerbit = 'Penerbit Baru' "
'
'menghapus penerbit_tb dari data table
For Each row As DataRow In dt.Select(filt)
    row.Delete()
Next
```

Pemberian nilai property data adapter **DeleteCommand** dengan perintah penghapusan baris dari table **penerbit_tb** sehingga data adapter menjalankan SQL saat method **Update** dipanggil. Memanggil method **Update()** pada data adapter untuk melimpahkan perubahan yang terjadi ke database.

```
'Update database
da.DeleteCommand = cmd
da.Update(ds, "penerbit_tb")
```

4.4.10 Dataset Dan XML

Dalam .NET, XML menempati tingkat menengah dari pengiriman data, XML adalah pembentuk utama ADO.NET. Dataset mengorganisasi data secara internal dengan format XML dan memiliki berbagai jenis method untuk membaca dan menulis XML, sebagai contoh:

- Anda dapat mengimpor atau mengekspor struktur dataset sebagai skema XML dengan menggunakan *System.Data.DataSet ReadXmlSchema* dan method *WriteXmlSchema*.
- Anda dapat membaca data dari dataset dan menulisnya ke file XML atau sebaliknya dengan **ReadXml()** dan **WriteXml()**. Ini sangat berguna ketika terjadi pertukaran data antar aplikasi atau membuat duplikasi dataset secara lokal.

Anda dapat menyimpan isi dan skema dataset ke dalam file XML dengan menggunakan method *WriteXml* atau dalam file terpisah menggunakan

WriteXml() dan **WriteXmlSchema()**. Contoh berikut ini menunjukkan bagaimana penguraian data dan skema.

- Buat Project baru bernama **WriteXml** dengan mengeklik menu **File->New->Project**
- Template pilih dengan **Console Application** hingga terbentuk file **Module1.vb**.
- Tambahkan reference **MySQL.Data** pada menu **Project->Add Reference->.NET->MySQL.Data**.

WriteXml: Module1.vb

```
Imports MySql.Data.MySqlClient

Module Module1

    Sub Main()
        Dim connString, db, user, pwd As String

        db = "dbpenjualanbuku" 'database
        user = "root" 'user default
        pwd = "" 'password kosong
        connString = "Database=" & db & ";Data Source=localhost;User
Id=" & user & ";Password=" & pwd

        'set up query
        Dim qry As String = "select * from penerbit_tb limit 0,3;"

        'Membuat connection
        Dim conn As MySqlConnection = New MySqlConnection(connString)

        Try
            'membuka connection
            conn.Open()

            'membuat data adapter
            Dim da As MySqlDataAdapter = New MySqlDataAdapter(qry,
conn)

            'membuat dan mengisi dataset
            Dim ds As DataSet = New DataSet()
            da.Fill(ds, "penerbit")

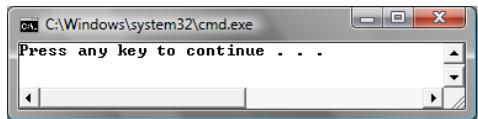
            'menguraikan dataset ke file XML
            ds.WriteXml("d:\penerbit\VB_MySQL\penerbit.xml")
        Catch ex As Exception
        End Try
    End Sub
End Module
```

```

'menampilkan pesan kesalahan
Console.WriteLine("Error: " & ex.ToString)
Finally
'menutup connection
conn.Close()
End Try
End Sub
End Module

```

Jalankan programnya dengan menekan tombol **Ctrl+F5**, hasilnya tampak seperti berikut ini:



Buka file XML dengan memilih menu **File->Open->File**, nama file dan folder sesuai dengan yang dituliskan pada program. Hasilnya kurang lebih seperti berikut ini:

```

<?xml version="1.0" standalone="yes"?>
<NewDataSet>
  <penerbit>
    <kode_penerbit>0000000001</kode_penerbit>
    <nama_penerbit>Maxikom CV.</nama_penerbit>
    <alamat>Residen H. Abdul Rozak No.33 B5-B6</alamat>
    <kota>Palembang</kota>
    <kode_pos>30114</kode_pos>
    <telp>(0771) 717598</telp>
    <email>maxi@maxikom.co.id</email>
    <website>maxikom.co.id</website>
  </penerbit>
  <penerbit>
    <kode_penerbit>0000000002</kode_penerbit>
    <nama_penerbit>Oase Media</nama_penerbit>
    <alamat>Karasak Utara 1/1 No.9</alamat>
    <kota>Bandung</kota>
    <telp>(022) 76630451</telp>
    <email>oasewm@yahoo.com</email>
    <website>www.oasemedia.com</website>
  </penerbit>
  <penerbit>
    <kode_penerbit>0000000003</kode_penerbit>
    <nama_penerbit>Informatika Bandung</nama_penerbit>
    <alamat>Bandung</alamat>
    <kota>Bandung</kota>
  
```

```
<kode pos>423423</kode pos>
<telp>314123</telp>
<email>if@biobses.com</email>
<website />
</penerbit>
</NewDataSet>
```

XML memetakan dataset sebagai hirarki. Elemen pertama XML adalah <**NewDataSet**> merupakan nama dataset, secara default dataset diberi nama **NewDataSet** jika Anda tidak menentukannya. Elemen <**penerbit**> sebagai pengelompok kolom data. Elemen selanjutnya dibuat secara berulang-ulang, merupakan nama kolom data, misalnya: <**kode_penerbit**> dan <**nama_penerbit**>.

4.5 Data Binding

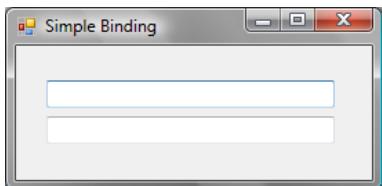
Kontrol Windows Form dapat diikatkan ke data database. Sekali data binding dibuat maka Anda dapat menampilkan dan mengubah data dengan kode program minimal. Sebagai contoh, Anda dapat mengikat kolom data ke property **Text** dari kontrol **TextBox**, bahkan mengikat keseluruhan table ke data grid melalui kontrol **DataGridView**.

4.5.1 Data Binding Sederhana

Data binding sederhana adalah hubungan satu-ke-satu antara property kontrol dan elemen tunggal dari data source. Anda dapat menggunakan *binding* kontrol yang menampilkan hanya satu nilai dalam satu waktu. Jika terjadi modifikasi data source maka method **Refresh** kontrol akan mengubah nilai yang sudah ter-*binding*.

- Buat Project baru bernama **SimpleBinding** dengan mengeklik menu **File->New->Project**.
- Pilih template dengan **Windows Application** hingga terbentuk file **Form1.vb**.

- Ubah property **Text** pada form, semula **Form1** menjadi **Simple Binding**.
- Tambahkan dua kontrol **TextBox** seperti tampak berikut ini:



- Tambahkan reference **MySQL.Data** pada menu **Project->Add Reference->.NET->MySQL.Data**.
- Klik dua kali pada Form (**event Form1_Load**), masukkan kode program berikut ini:

SimpleBinding: Form1.vb

```
Imports MySql.Data.MySqlClient

Public Class Form1

    Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load
        Dim connString, db, user, pwd As String

        db = "dbpenjualanbuku" 'database
        user = "root" 'user default
        pwd = "" 'password kosong
        connString = "Database=" & db & ";Data Source=localhost;User Id=" & user & ";Password=" & pwd

        'set up query
        Dim sql As String = "select * from penulis_tb"

        'Membuat connection
        Dim conn As MySqlConnection = New MySqlConnection(connString)

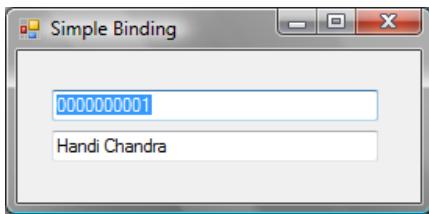
        'membuat data adapter
        Dim da As MySqlDataAdapter = New MySqlDataAdapter(sql, conn)

        'membuat dataset
        Dim ds As DataSet = New DataSet
        da.Fill(ds, "penulis_tb")

        'Bind kolom kode_penulis dari table penulis_tb ke TextBox1
        TextBox1.DataBindings.Add("text", ds,
        "penulis_tb.kode_penulis")
        'Bind kolom nama penulis dari table penulis_tb ke TextBox2
```

```
    TextBox2.DataBindings.Add("text", ds,
"penulis_tb.nama_penulis")
End Sub
End Class
```

Jalankan programnya dengan menekan tombol keyboard **Ctrl+F5**, hasilnya kurang lebih seperti berikut ini:



Disini Anda mengisi dataset dengan semua kolom dan baris dari table **penulis_tb**.

```
'set up query
Dim sql As String = "select * from penulis_tb"

'Membuat connection
Dim conn As MySqlConnection = New MySqlConnection(connString)

'membuat data adapter
Dim da As MySqlDataAdapter = New MySqlDataAdapter(sql, conn)

'membuat dataset
Dim ds As DataSet = New DataSet
da.Fill(ds, "penulis_tb")
```

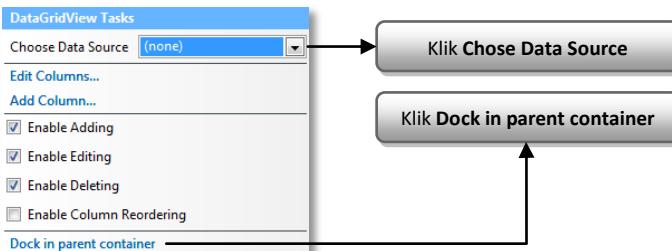
Cara mengikatkan property **Text** kontrol **TextBox1** dengan kolom **kode_penulis** dan property **Text** kontrol **TextBox2** dengan kolom **nama_penulis** adalah sebagai berikut:

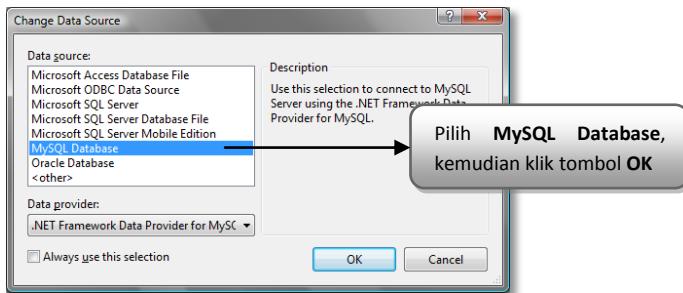
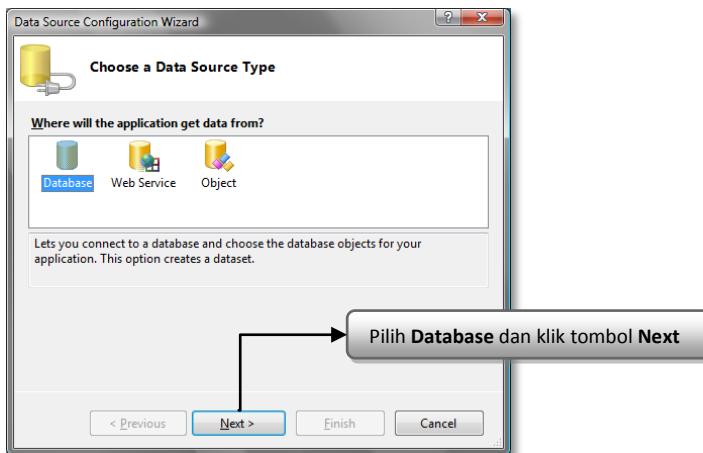
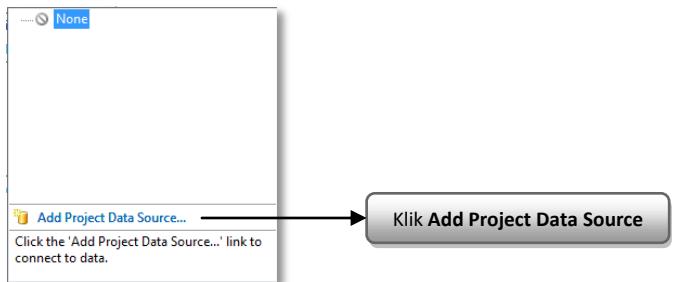
```
'Bind kolom kode_penulis dari table penulis_tb ke TextBox1
TextBox1.DataBindings.Add("text", ds, "penulis_tb.kode_penulis")
'Bind kolom nama_penulis dari table penulis_tb ke TextBox2
TextBox2.DataBindings.Add("text", ds, "penulis_tb.nama_penulis")
```

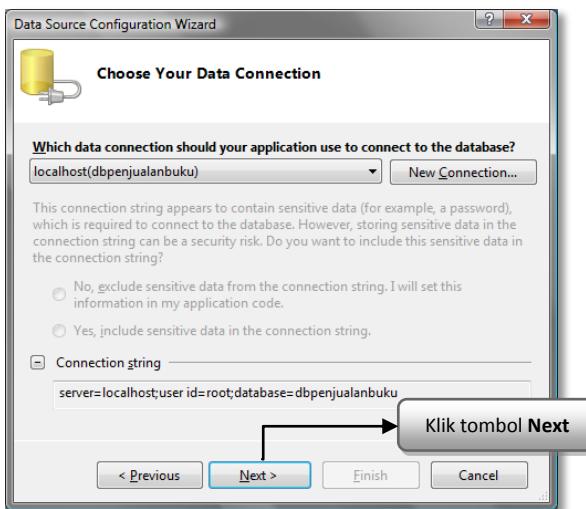
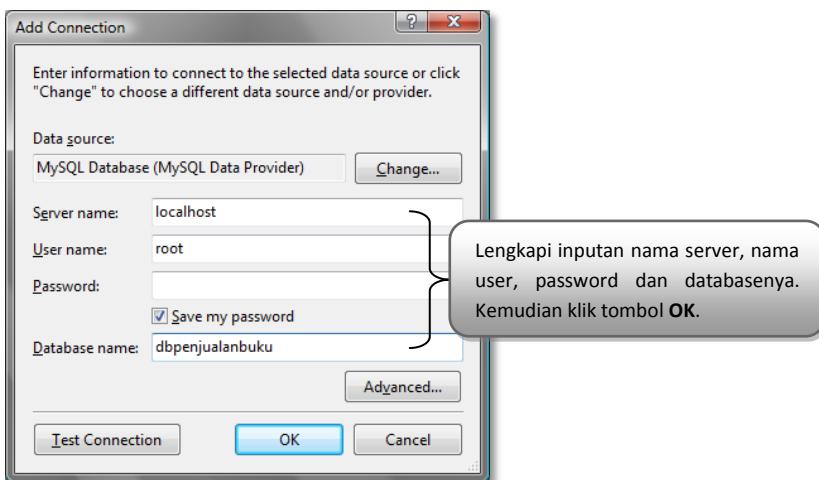
4.5.2 Data Binding Kompleks

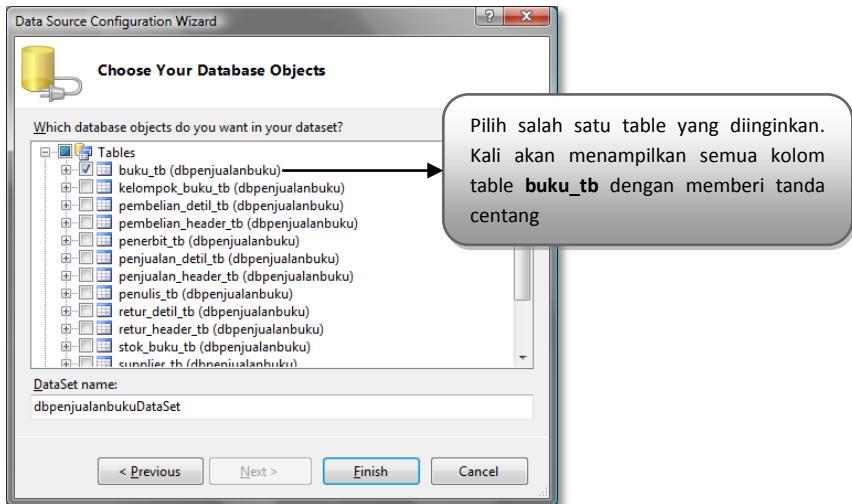
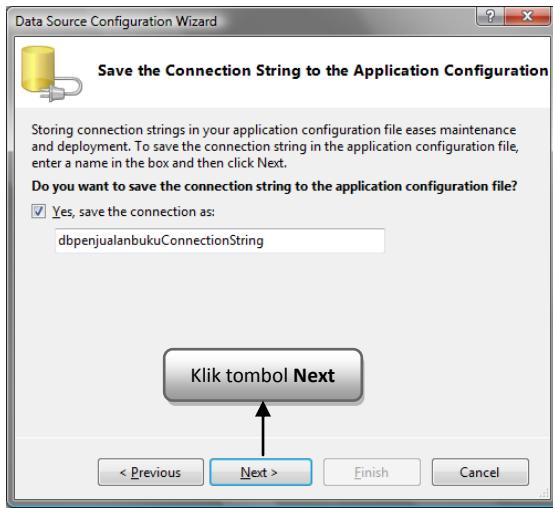
Dikatakan data binding kompleks jika terdapat hubungan satu atau lebih elemen data antara kontrol dengan data source. Jenis ini memungkinkan kita untuk mengikat kontrol dengan lebih dari satu nilai untuk ditampilkan dalam satu waktu, seperti kontrol **Data Grid** atau **Data List**. Buatlah program yang melibatkan kontrol **DataGridView**:

- Buat Project baru bernama **ComplexBinding** dengan mengeklik menu **File->New->Project**.
- Pilih template dengan **Windows Application** hingga terbentuk file **Form1.vb**.
- Tambahkan reference **MySQL.Data** pada menu **Project->Add Reference->.NET->MySQL.Data**.
- Ubah property **Text** pada form, semula **Form1** menjadi **Complex Binding**.
- Tambahkan kontrol **DataGridView** dari **ToolBox**, ikuti langkah-langkah selanjutnya, seperti tampak gambar di bawah ini:









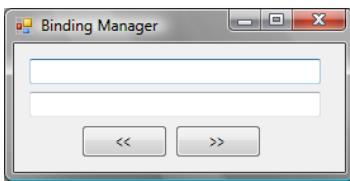
Jalankan programnya dengan menekan tombol keyboard **Ctrl+F5**, hasilnya kurang lebih seperti berikut ini:

kode_buku	isbn	judul	kode_penulis	kode_penerbit	kode_kelompok	harga_jual
9789792075830	9792075836	Aplikasi Manajemen Perekutran Berbasis Access 97/20...	0000000005	0000000005	0000000001	35000
9789792405569	9792405569	Tips & Trik Baru Menembus Tes Bakat Skolastik	0000000015	0000000005	0000000005	25000
9789793338125	9793338121	Esenси-есенси Bahasa Pemrograman JAVA Disertai Lebih...	0000000004	0000000003	0000000001	90000
9789793338729	9793338725	Membuat Aplikasi Web Interaktif dengan ASP	0000000014	0000000003	0000000001	45000
9789793767079	9793767073	Efek Partikel 3ds max 6	0000000001	0000000001	0000000001	35000
9789793767543	9793767545	7 Jam Belajar Visual Basic .NET Untuk Orang Awam	0000000010	0000000001	0000000001	40000
9789793767669	9793767669	7 Jam Belajar Interaktif Autocad Untuk Orang Awam	0000000001	0000000001	0000000001	45000
9789795336730	9795336738	Internet Marketing	0000000013	0000000007	0000000006	25000
9789797633134	9797633136	Kumpulan Latihan Pemrograman Delphi	0000000009	0000000007	0000000001	25000
9789797941307	9797941302	Joomla Cara Cepat & Mudah Membuat Website	0000000011	0000000004	0000000001	45000
9789797941468	9797941469	Tanya Jawab Sepertai Joomla	0000000003	0000000004	0000000001	45000
978979992260	979992265	30 Menit Menjadi Webmaster	0000000012	0000000002	0000000001	29500

4.5.3 Binding Manajer

Kita akan menambahkan sedikit fitur pada data binding, yaitu fitur untuk memajukan atau memundurkan *pointer* (penunjuk baris), sehingga dapat diketahui baris datanya baik sesudah maupun sebelum baris yang ditunjuk.

- Buat Project baru bernama **BindingManager** dengan mengeklik menu **File->New->Project**.
- Pilih template dengan **Windows Application** hingga terbentuk file **Form1.vb**.
- Ubah Property **Text** pada Form, semula **Form1** menjadi **Binding Manager**.
- Tambahkan dua kontrol **TextBox** dan dua **Button** seperti tampak gambar berikut ini:



- Tambahkan reference MySQL.Data pada menu Project->Add Reference->.NET->MySQL.Data.
- Klik dua kali pada Form (**event Form1_Load**), masukkan kode program berikut ini:

BindingManager: Form1.vb

```

Imports MySql.Data.MySqlClient

Public Class Form1

    'mendeklarasikan field binding manager
    Private bMgr As BindingManagerBase

    Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
        Dim connString, db, user, pwd As String

        db = "dbpenjualanbuku" 'database
        user = "root" 'user default
        pwd = "" 'password kosong
        connString = "Database=" & db & ";Data Source=localhost;User
Id=" & user & ";Password=" & pwd

        'set up query
        Dim sql As String = "select * from penulis_tb"

        'Membuat connection
        Dim conn As MySqlConnection = New MySqlConnection(connString)

        'membuat data adapter
        Dim da As MySqlDataAdapter = New MySqlDataAdapter(sql, conn)

        'membuat dataset
        Dim ds As DataSet = New DataSet
        da.Fill(ds, "penulis_tb")

        'Bind kolom kode_penulis dari table penulis_tb ke TextBox1
        TextBox1.DataBindings.Add("text", ds,
"penulis_tb.kode_penulis")
        'Bind kolom nama_penulis dari table penulis_tb ke TextBox2
        TextBox2.DataBindings.Add("text", ds,
"penulis_tb.nama_penulis")

        'Membuat binding manager
        bMgr = MyBase.BindingContext(ds, "penulis_tb")

    End Sub

    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e

```

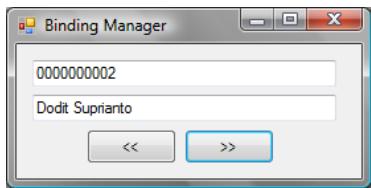
```

As System.EventArgs) Handles Button1.Click
    'menunjuk ke baris sebelumnya dan menyegarkan kembali isi
dari textbox
    bMgr.Position -= 1
End Sub

Private Sub Button2_Click(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles Button2.Click
    'menunjuk ke baris selanjutnya dan menyegarkan kembali isi
dari textbox
    bMgr.Position += 1
End Sub
End Class

```

Jalankan programnya dengan menekan tombol keyboard **Ctrl+F5**, hasilnya kurang lebih seperti berikut ini:



Aplikasi ini serupa dengan **SimpleBinding**, tetapi ditambah dengan manager binding untuk menavigasi data pada table. Anda harus mendeklarasikan **BindingManagerBase** sebagai variable global dalam class **Form1**, sehingga dapat dikenali pada seluruh bagian Form.

```
'mendeklarasikan field binding manager
Private bMgr As BindingManagerBase
```

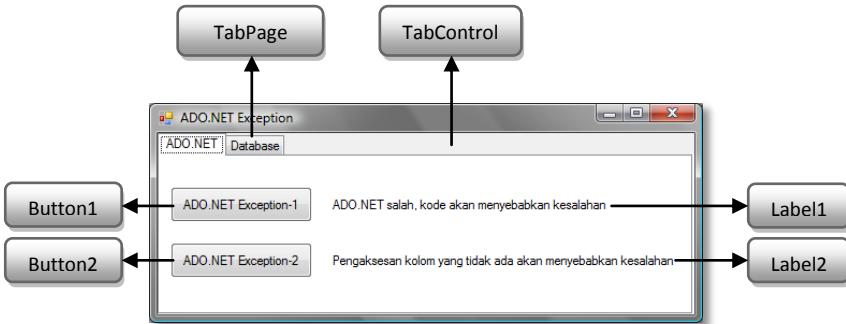
4.6 Penanganan Exception

Suatu aplikasi database yang handal bergantung pada tingkat kehati-hatian terhadap aplikasi yang dibuat. Kita harus serius memperhatikan segala kemungkinan kesalahan yang mungkin bakal terjadi. Dalam pemrograman database, kesalahan timbul dari tiga jenis sumber, antara lain: **Program aplikasi**, **ADO.NET**, dan **Server database**. Kali ini kita menitikberatkan pada penanganan kesalahan **ADO.NET**.

4.6.1 Penanganan Exception ADO.NET (Bagian 1)

Perhatikan bagaimana penanganan exception yang dilempar oleh ADO.NET. Exception ini timbul saat ADO.NET berusaha menjalin komunikasi dengan MySQL sebelum database server memberikan tanggapan. Program menggunakan **Windows Application** karena kita akan menampilkan pesan kesalahannya melalui object **MessageBox**. Sebagai percobaan, kita jalankan store procedure tanpa disertai property **CommandText**. Pertama, tanpa menyertakan penanganan exception, dan kedua kita memodifikasi program namun disertai dengan penanganan exception.

- Buat Project baru bernama **AdoNetException** dengan mengeklik menu **File->New->Project**.
- Pilih template **Windows Application** hingga terbentuk file **Form1.vb**.
- Ubah Property **Text** pada Form, semula **Form1** menjadi **ADO.NET Exception**.
- Tambahkan reference **MySQL.Data** pada menu **Project->Add Reference->.NET->MySQL.Data**.
- Tambahkan kontrol **TabControl1**, terdiri dari **TabPage1** dan **TabPage2**. Ubah property **Text** dari **TabPage1** menjadi “**ADO.NET**” dan **TabPage2** menjadi “**Database**”. Tambahkan dua kontrol **Button** pada **TabPage1** yaitu **Button1** dan **Button2**, ubah property **Text** dari **Button1** menjadi “**ADO.NET Exception-1**” dan **Button2** menjadi “**ADO.NET Exception-2**”. Tambahkan dua kontrol **Label** yaitu **Label1** dan **Label2**, ubah property **Text** label menjadi “**ADO.NET salah, kode akan menyebabkan kesalahan**” untuk **Label1** dan “**Pengaksesan kolom yang tidak ada akan menyebabkan kesalahan**” untuk **Label2**.



AdoNetException: Form1.vb

```

Imports MySql.Data.MySqlClient

Public Class Form1

    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click
        Dim connString, db, user, pwd As String

        db = "dbpenjualanbuku" 'database
        user = "root" 'user default
        pwd = "" 'password kosong
        connString = "Database=" & db & ";Data Source=localhost;User Id=" & user & ";Password=" & pwd

        'Membuat connection
        Dim conn As MySqlConnection = New MySqlConnection(connString)

        'membuat command
        Dim cmd As MySqlCommand = conn.CreateCommand
        cmd.CommandType = CommandType.Text
        'Sengaja pernyataan di bawah ini ditutup, untuk melihat kesalahannya
        'cmd.CommandText = "CALL BROWSE_SP ('PENULIS', '', 'Dodit')"

        ' Open connection
        conn.Open()

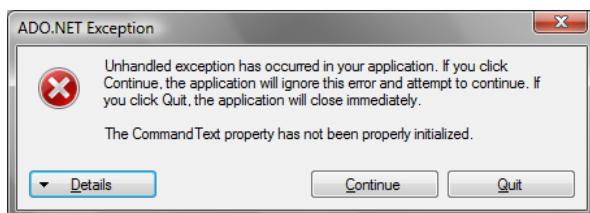
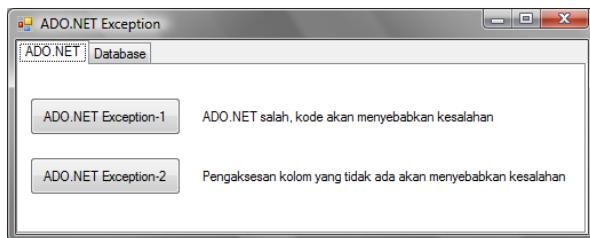
        ' membuat data reader
        Dim rdr As MySqlDataReader = cmd.ExecuteReader()

        ' menutup data reader
        rdr.Close()
        If conn.State = ConnectionState.Open Then
            MessageBox.Show("Blok akhir penutupan connection ",
                           "Akhir")
        End If
    End Sub

```

```
    conn.Close()
End If
End Sub
End Class
```

Jalankan programnya dengan menekan tombol keyboard **Ctrl+F5**, kemudian klik Button1 (ADO.NET Exception-1) hingga keluar pesan kesalahan.



Pesan di atas menunjukkan bahwa kesalahan tidak tertangani dengan baik. Seandainya program telah terpaket menjadi file **Setup** (paket deployment), maka saat terjadi kesalahan, program akan keluar dengan sendirinya tanpa memberi pesan apapun, hal ini akan membingungkan pengguna program aplikasi.

Selanjutnya memodifikasi program sebelumnya dengan menambahkan kode penanganan exception (**Try Catch Finally**), seperti berikut ini:

AdoNetException: Form1.vb

```
Imports MySql.Data.MySqlClient
Public Class Form1
```

```

    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles Button1.Click
        Dim connString, db, user, pwd As String

        db = "dbpenjualanbuku" 'database
        user = "root" 'user default
        pwd = "" 'password kosong
        connString = "Database=" & db & ";Data Source=localhost;User
Id=" & user & ";Password=" & pwd

        'membuat command
        Dim cmd As MySqlCommand = conn.CreateCommand
        cmd.CommandType = CommandType.Text
        'Sengaja pernyataan di bawah ini ditutup, untuk melihat
kesalahannya
        'cmd.CommandText = "CALL BROWSE_SP ('PENULIS', '', 'Dodit')"

        Try
            ' Open connection
            conn.Open()

            ' membuat data reader
            Dim rdr As MySqlDataReader = cmd.ExecuteReader()

            ' menutup data reader
            rdr.Close()

        Catch ex As MySqlException
            Dim str As String = "Sumber: " & ex.Source
            str &= ControlChars.NewLine
            str &= "Pesan Exception: " & ex.Message
            MessageBox.Show(str, "Database Exception")

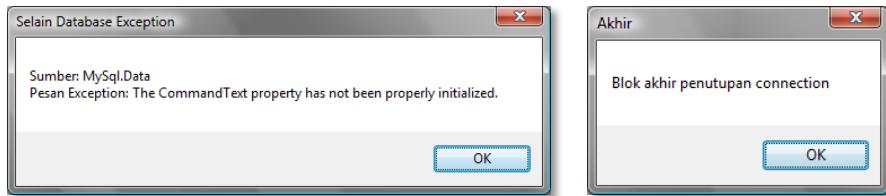
        Catch ex As Exception
            Dim str As String = "Sumber: " & ex.Source
            str &= ControlChars.NewLine
            str &= "Pesan Exception: " & ex.Message
            MessageBox.Show(str, "Selain Database Exception")

        Finally
            If conn.State = ConnectionState.Open Then
                MessageBox.Show("Blok akhir penutupan connection",
"Akhir")
                conn.Close()
            End If
        End Try

    End Sub
End Class

```

Jalankan programnya dengan menekan tombol keyboard **Ctrl+F5**, kemudian klik Button1 (ADO.NET Exception-1).



Di dalam program terdapat dua klausa Catch:

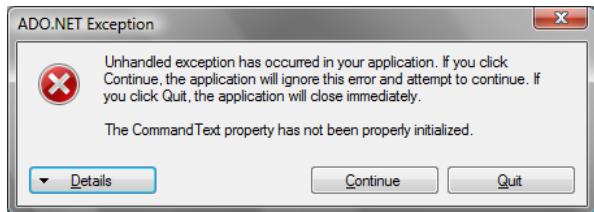
```
Catch ex As MySqlException
    Dim str As String = "Sumber: " & ex.Source
    str &= ControlChars.NewLine
    str &= "Peser Exception: " & ex.Message
    MessageBox.Show(str, "Database Exception")

Catch ex As Exception
    Dim str As String = "Sumber: " & ex.Source
    str &= ControlChars.NewLine
    str &= "Peser Exception: " & ex.Message
    MessageBox.Show(str, "Selain Database Exception")
```

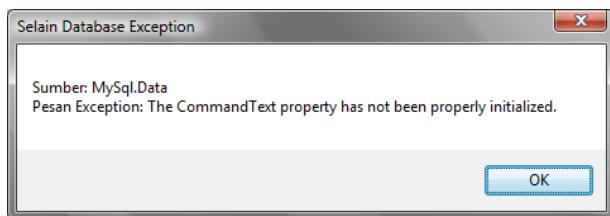
Contoh di atas memang tidak biasa karena pengaturan property CommandText dihilangkan, hal ini bertujuan untuk menampilkan pesan kesalahan ADO.NET (ADO.NET exception). Anda telah menyiapkan store procedure, tetapi menentukan store procedure mana yang akan dipanggil:

```
'membuat command
Dim cmd As MySqlCommand = conn.CreateCommand
cmd.CommandType = CommandType.Text
'Sengaja pernyataan di bawah ini ditutup, untuk melihat kesalahannya
'cmd.CommandText = "CALL BROWSE_SP ('PENULIS', '', 'Dodit')"
```

Saat pemanggilan method **ExecuteReader** maka keluar pesan exception seperti berikut ini:



Pesan tersebut terjadi sebelum dilakukan penanganan exception. Setelah dilakukan penanganan exception, maka diagnosa kesalahan menjadi semakin akurat, yaitu “**CommandText property has not been initialized.**”:



Setelah membaca apa yang sedang terjadi maka aliran program akan menuju ke pernyataan **Try**. Klausa **Catch** pertama adalah jenis database exception. **Catch** kedua adalah jenis exception umum yang menangkap kesalahan seperti pesan: “**CommandText property has not been initialized.**” Meskipun Anda kira kegagalan tersebut tergolong exception database, namun sebenarnya adalah exception ADO.NET dengan kata lain jebakan kesalahan terjadi sebelum mencapai ke server database.

Karena property **CommandText** belum ditetapkan maka ketika **Button1** diklik, maka exception akan dilempar dan ditangkap oleh klausa **Catch** yang kedua sekalipun klausa **Catch** **MySQLException** telah tersedia.

Segala pernyataan atau kode program yang berada diantara blok **Finally** pasti akan dieksekusi, baik program sukses maupun gagal dilaksanakan.

```
Finally  
If conn.State = ConnectionState.Open Then
```

```
    MessageBox.Show("Blok akhir penutupan connection ", "Akhir")
    conn.Close()
End If
```

4.6.2 Penanganan Exception ADO.NET (Bagian 2)

Anda akan menjalankan store procedure dan selanjutnya merujuk ke suatu kolom yang belum ada, kemudian mengembalikannya ke dataset. Kesalahan ini akan dilempar ke exception ADO.NET.

Klik dua kali pada **Button2**, kemudian masukkan kode program berikut:

AdoNetException: Button2_Click

```
Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button2.Click
    Dim connString, db, user, pwd As String

    db = "dbpenjualanbuku" 'database
    user = "root" 'user default
    pwd = "" 'password kosong
    connString = "Database=" & db & ";Data Source=localhost;User
Id=" & user & ";Password=" & pwd

    'Membuat connection
    Dim conn As MySqlConnection = New MySqlConnection(connString)

    'membuat command
    Dim cmd As MySqlCommand = conn.CreateCommand
    cmd.CommandType = CommandType.Text
    'Sengaja pernyataan di bawah ini ditutup, untuk melihat
kesalahannya
    cmd.CommandText = "CALL BROWSE_SP ('PENULIS', '', 'Dodit')"

    Try
        ' Open connection
        conn.Open()

        ' membuat data reader
        Dim rdr As MySqlDataReader = cmd.ExecuteReader()
        'mengakses kolom yang tidak ada
        Dim str As String = rdr.GetValue(20).ToString()

        ' menutup data reader
        rdr.Close()

    Catch ex As System.InvalidOperationException
        Dim str As String = "Source: " & ex.Source
    End Try
End Sub
```

```

        str &= ControlChars.NewLine
        str &= "Message: " & ex.Message
        str &= ControlChars.NewLine & ControlChars.NewLine
        str &= "Stack Trace: " & ex.StackTrace
        MessageBox.Show(str, "Specific Exception")

    Catch ex As MySqlException
        Dim str As String = "Sumber: " & ex.Source
        str &= ControlChars.NewLine
        str &= "Pesanan Exception: " & ex.Message
        MessageBox.Show(str, "Database Exception")

    Catch ex As Exception
        Dim str As String = "Sumber: " & ex.Source
        str &= ControlChars.NewLine
        str &= "Pesanan Exception: " & ex.Message
        MessageBox.Show(str, "Selain Database Exception")

    Finally
        If conn.State = ConnectionState.Open Then
            MessageBox.Show("Blok akhir penutupan connection",
    "Akhir")
        conn.Close()
    End If
End Try
End Sub

```

Jalankan programnya dengan menekan tombol keyboard Ctrl+F5, klik **Button2** (ADO.NET Exception-2)

4.7 Transaction

Transaction merupakan serangkaian operasi yang dilakukan untuk menjamin bahwa suatu operasi sukses atau gagal sekaligus dalam satu kesatuan proses. Sebagai contoh adalah proses transaksi pengiriman uang dari suatu akun. Transaksi ini meliputi dua operasi yaitu pencatatan pengurangan uang dari akun pengurangan dan penambahan uang ke akun penyimpanan. Kedua taransaksi yang terjadi harus sukses bersamaan dan tercatat pada akun (**commit**), atau keduanya gagal bersamaan pula, sehingga dilakukan transaksi balik (**rollback**). Dengan demikian akun selalu pada kondisi yang konsisten.

Pada dasarnya **Transaction** bertujuan untuk menjaga keutuhan data, apalagi jika aplikasi melibatkan operasi banyak table yang saling berhubungan dan banyak pengguna berusaha untuk membarui database secara bersamaan.

4.7.1 Penggunaan Transaction

Transaction digunakan ketika beberapa operasi harus sukses atau gagal dalam satu kesatuan proses. Berikut ini skenario yang memungkinkan transaction digunakan:

- Dalam tumpukan proses yang menunggu untuk dilaksanakan, dimana terdapat banyak baris yang harus ditambahkan atau dihapus dalam satu kesatuan.
- Perubahan yang terjadi sewaktu-waktu ke suatu table yang membutuhkan table lain untuk menjaga kesesuaian data.
- Ketika dilakukan modifikasi data pada dua atau lebih database secara bersamaan.
- Dalam transaksi terdistribusi, dimana data dimanipulasi dalam database pada server yang berbeda.

Saat menggunakan transaction, Anda menempatkan pengunci pada antrian data, tidak boleh ada operasi yang dapat mengambil tempat pada data yang terkunci sampai Anda melepaskan kuncian. Anda dapat mengunci apapun, mulai dari baris tunggal sampai keseluruhan database. Hal ini disebut dengan *concurrency*.

Seperti transaksi yang terjadi pada bank, penguncian akan memastikan bahwa ada dua pemisahan transaksi secara jelas, tidak boleh mengakses satu akun yang sama dalam satu waktu karena bisa menyebabkan catatan penyimpanan atau penarikan menjadi hilang.

4.7.2 Kode Program Transaction

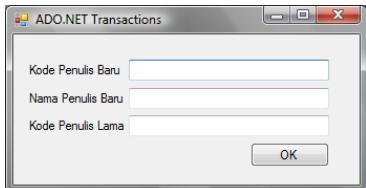
Penulisan kode program transaction dapat dibuat dengan dua cara, pertama ditulis langsung pada sisi *backend* (Database MySQL), kedua melalui *front-end* (Transaction ADO.NET). Kali ini kita akan membahas bagaimana penggunaan transaction dengan cara kedua. Ada tiga jenis kode transaction, antara lain:

- **BeginTransaction**, menandai sebagai awal dari transaction.
- **Commit**, menandakan bahwa transaction sukses dilaksanakan sekaligus mengakhiri transaction dan memberi perintah ke database untuk menyimpan perubahannya secara permanen.
- **Rollback**, menandakan bahwa transaction gagal dilaksanakan sekaligus mengakhiri transaction dan memberi perintah ke database untuk mengembalikan ke kondisi asli seperti sebelum terjadi perubahan.

Catatan: Tidak ada pernyataan **End Transaction**, baik secara nyata maupun tersembunyi karena transaction akan berakhir dengan sendirinya jika menemui pernyataan **commit** atau **rollback**.

Berikut ini contoh penggunaan transaction sederhana:

- Buat Project baru bernama **AdoNetTransactions** dengan mengeklik menu **File->New->Project**
- Pilih template dengan **Windows Application** hingga terbentuk file **Form1.vb**.
- Ubah Property **Text** Form, semula **Form1** menjadi **ADO.NET Transactions**.
- Tambahkan tiga kontrol **Label**, tiga kontrol **TextBox** dan satu **Button** seperti tampak berikut ini:



- Tambahkan reference **MySQL.Data** pada menu **Project->Add Reference->.NET->MySQL.Data**.

AdoNetTransactions: Form1.vb

```
Imports MySql.Data.MySqlClient

Public Class Form1

    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click
        Dim connString, db, user, pwd As String

        db = "dbpenjualanbuku" 'database
        user = "root" 'user default
        pwd = "" 'password kosong
        connString = "Database=" & db & ";Data Source=localhost;User Id=" & user & ";Password=" & pwd

        'Membuat connection
        Dim conn As MySqlConnection = New MySqlConnection(connString)

        'pernyataan INSERT
        Dim sqlins As String
        sqlins = "insert into penulis_tb "
        sqlins &= "(kode_penulis, nama_penulis) "
        sqlins &= "values (@kd_baru, @nm_baru)"

        'pernyataan DELETE
        Dim sqldel As String
        sqldel = "delete from penulis_tb "
        sqldel &= "where kode_penulis = @kd_lama"

        'membuka connection
        conn.Open()

        'awal transaction
        Dim sqltrans As MySqlTransaction = conn.BeginTransaction()

        Try
            'membuat command insert
            Dim cmdins As MySqlCommand = conn.CreateCommand()
            cmdins.CommandText = sqlins
```

```

cmdins.Transaction = sqltrans
cmdins.Parameters.Add("@kd_baru", MySqlDbType.VarChar,
10)
cmdins.Parameters.Add("@nm_baru", MySqlDbType.VarChar,
30)

'membuat command delete
Dim cmddel As MySqlCommand = conn.CreateCommand()
cmddel.CommandText = sqldel
cmddel.Transaction = sqltrans
cmddel.Parameters.Add("@kd_lama", MySqlDbType.VarChar,
10)

'Menambahkan penulis
cmdins.Parameters("@kd_baru").Value = TextBox1.Text
cmdins.Parameters("@nm_baru").Value = TextBox2.Text
cmdins.ExecuteNonQuery()

'Menghapus penulis
cmddel.Parameters("@kd_lama").Value = TextBox3.Text
cmddel.ExecuteNonQuery()

'Commit transaction
sqltrans.Commit()

'Tidak ada exception,
'transaction committed (terupdate permanen)
MessageBox.Show("Transaction committed")

Catch ex As MySqlException
    ' Roll back transaction
    sqltrans.Rollback()
    MessageBox.Show("Transaction rolled back: " + ex.Message,
"Rollback Transaction")

Catch ex As Exception
    MessageBox.Show("System Error: " + ex.Message, "Error")

Finally
    'Menutup connection
    conn.Close()
End Try
End Sub
End Class

```

BAB 5

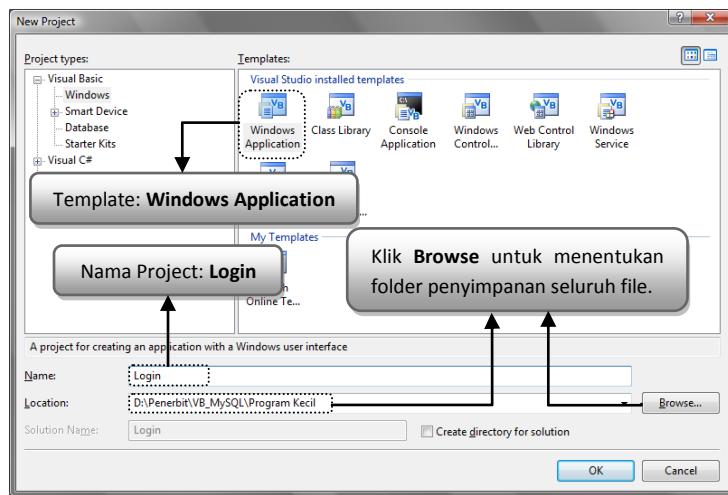
Contoh-contoh Program

Pada bab ini akan diberikan beberapa contoh program kecil yang akan memudahkan Anda memahami keseluruhan project demo **Penjualan Buku** yang ada dalam CD penyerta buku.

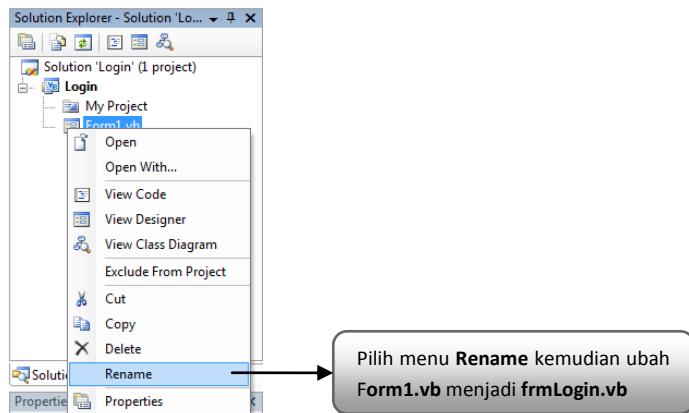
5.1 Form Login Tingkat Database (Project Login)

Form Login adalah form yang bertujuan untuk memastikan bahwa user tersebut berhak untuk mengakses aplikasi. Karena aplikasi berhubungan erat dengan database maka harus dipastikan bahwa user tersebut berhak untuk menjalin koneksi dengan database serta memanipulasi data yang ada di dalamnya.

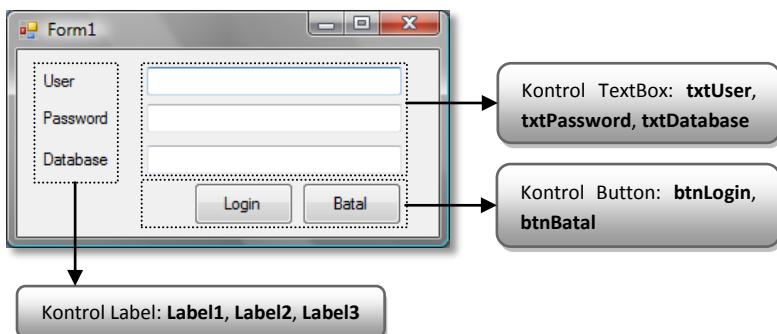
1. Buka menu **File->New->Project**, pilih Template **Windows Application**, beri nama Project dengan **Login**.



2. Klik kanan pada **Form1.vb** di bagian **Solution Explorer** kemudian ganti namanya menjadi **frmLogin.vb**



3. Rancang desain **frmLogin** beserta kelengkapan kontrolnya seperti ini

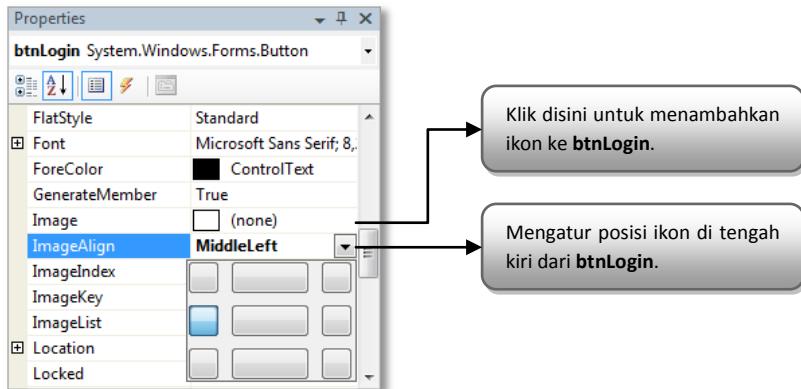


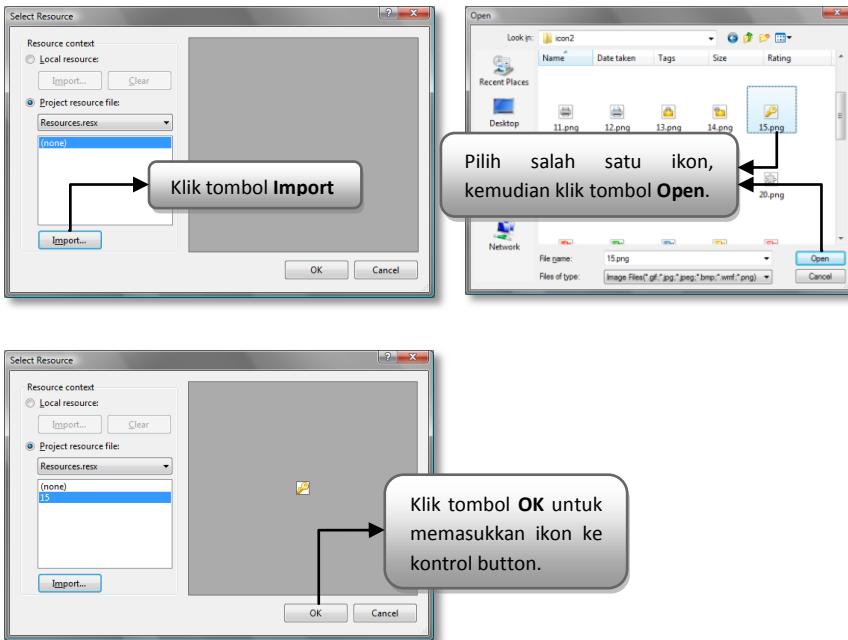
Dafar objek dan nilai property yang dibutuhkan dalam form login:

Kontrol	Properties	Nilai
Form	Name	frmLogin
	Text	Login
	StartPosition	CenterScreen
	Event	frmLogin_Load
Label	Name	Label1
	Text	User

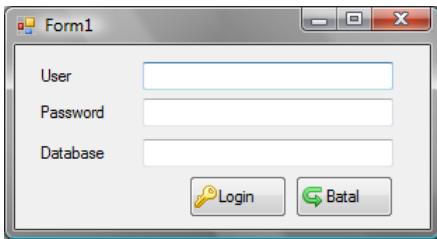
Label	Name	Label2
	Text	Password
Label	Name	Label3
	Text	Database
TextBox	Name	txtUser
TextBox	Name	txtPassword
TextBox	Name	txtDatabase
Button	Name	btnLogin
	Text	Login
	Image	15.png
	Event	btnLogin_Click
Button	Name	btnBatal
	Text	Batal
	Image	17.png
	Event	btnBatal_Click

4. Menambahkan ikon pada **btnLogin** dan **btnBatal**. Klik sekali pada kontrol **btnLogin** kemudian tambahkan ikon di **Properties->Images**, pilih salah satu gambar yang dikehendaki untuk ditempatkan pada kontrol button:

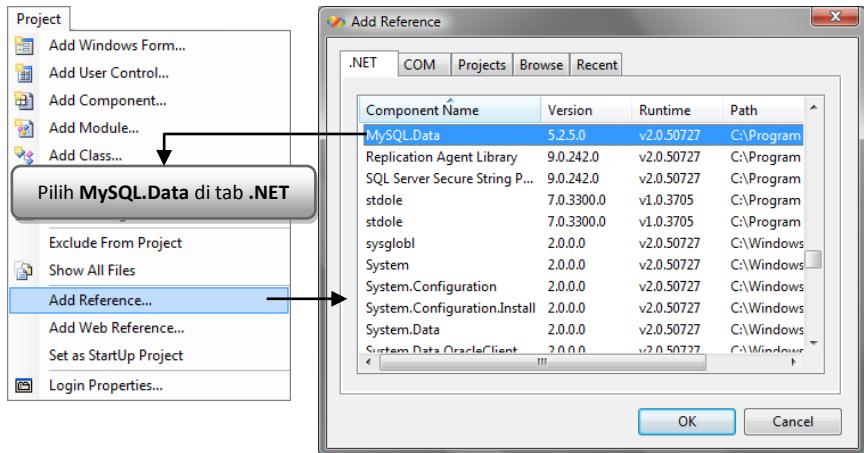




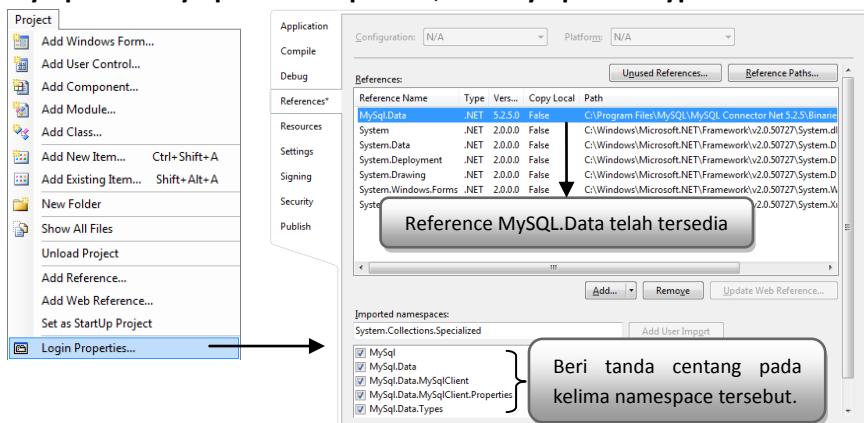
Untuk penambahan ikon pada **btnBatal** lakukan dengan cara yang sama. Desain akhir form login seperti gambar di bawah ini:



5. Buka menu **Project->Add Reference** kemudian pilih tab **.NET->MySQL.Data**. Referensi **MySQL.Data** pada dasarnya merujuk ke abstraksi class hasil instalasi **MySQL Connector .NET**.

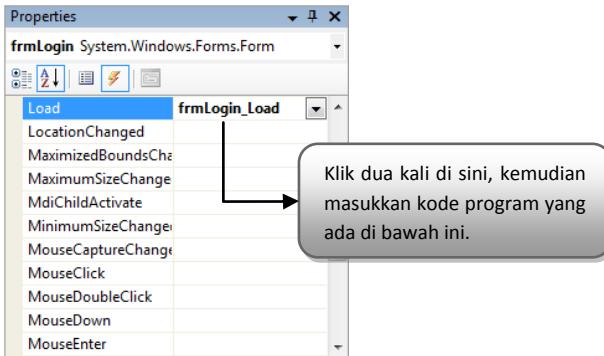


Buka menu **Project->Login Properties->References** kemudian beri tanda centang pada beberapa item *namespace* **MySQL.Data** antara lain: **MySql**, **MySQL.Data**, **MySQL.Data.MySqlClient**, **MySQL.Data.MySqlClient.Properties**, dan **MySQL.Data.Types**.



6. Pemberian kode program pada form login.

Klik satu kali pada object **frmLogin**, kemudian klik dua kali pada **Properties->Events->Load**.



Masukkan kode program di bawah ini pada event **frmLogin Load**:

Event frmLogin_Load

```

Private Sub frmLogin_Load(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles MyBase.Load
    'Judul Form
    Me.Text = "Login User"

    'memosisikan teks di tengah kiri
    btnLogin.TextAlign = ContentAlignment.MiddleLeft
    btnBatal.TextAlign = ContentAlignment.MiddleLeft

    'memosisikan ikon di tengah kanan
    btnLogin.ImageAlign = ContentAlignment.MiddleRight
    btnBatal.ImageAlign = ContentAlignment.MiddleRight

    'me-masking password dengan karakter *
    txtPassword.PasswordChar = "*"
End Sub

```

Klik satu kali pada **btnLogin**, kemudian klik dua kali pada **Properties->Events->Click**. Masukkan kode program pada event **btnLogin Click**.

Event btnLogin_Click

```

Private Sub btnLogin_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnLogin.Click
    Dim CN As New MySqlConnection 'Deklarasi variable object
    Connection ke MySQL
    Dim db, user, pwd, myConnectionString As String
    Try
        db = txtDatabase.Text
        user = txtUser.Text

```

```

pwd = txtPassword.Text
If db.ToString <> "" And user.ToString <> "" Then
    myConnectionString = "Database=' " & db & "';Data
Source=localhost;User Id=' " & user & "';Password=' " & pwd & " '"
    CN = New MySqlConnection(myConnectionString)
    CN.Open()
    'lakukan sesuatu disini, seperti membuka form MDI.
    'frmMDI.Show()
    MessageBox.Show("User memiliki hak akses", "Login",
MessageBoxButtons.OK, MessageBoxIcon.Information)
Else
    MessageBox.Show("Data harus diisi lengkap!", "Inputan",
MessageBoxButtons.OK, MessageBoxIcon.Warning)
End If

Catch ex As Exception
    CN.Dispose()
    CN = Nothing
    MessageBox.Show(ex.Message, "Terjadi Kegagalan!",
MessageBoxButtons.OK, MessageBoxIcon.Error)
End Try
End Sub

```

Klik satu kali pada **btnBatal**, kemudian klik dua kali pada **Properties->Events->Click**. Masukkan kode program pada event **btnBatal Click**.

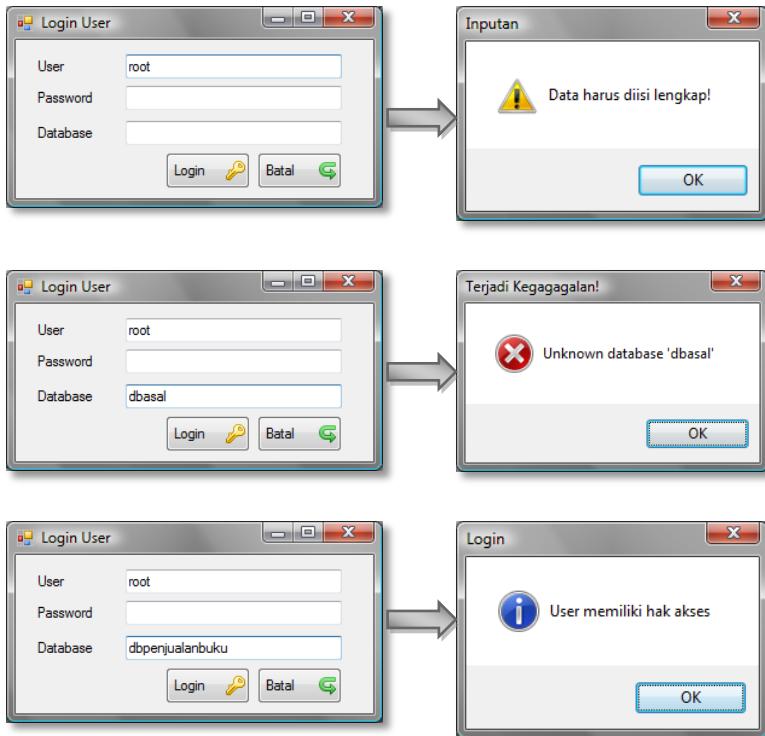
Event btnBatal_Click

```

Private Sub btnBatal_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnBatal.Click
    'keluar dari form login
    Me.Close()
End Sub

```

7. Jalankan programnya dengan menekan tombol keyboard **F5** atau **Ctrl+F5**. Secara default User: **root**, Password: <kosong>, Database: **dbpenjualanbuku**. Adapun kemungkinan hasilnya adalah:



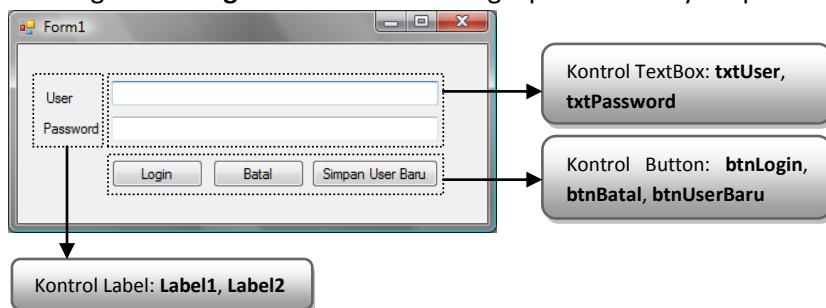
5.2 Form Login Tingkat Table (Project LoginMD5)

Kalau sebelumnya kita telah membuat form login pada tingkat database, sekarang kita akan membuat login pada tingkat table, dengan syarat program telah terkoneksi dengan database terlebih dahulu sebelum melakukan login pada tingkat table. Program login table membutuhkan satu table bernama **login_tb** yang terdiri dari kolom **user** dan **password**. Kolom password berisi data password yang disandikan dengan metode **MD5**.

- Buat table **login_tb** yang diletakkan di salah satu database yang Anda miliki, dalam kasus ini kita menggunakan database **dbpenjualanbuku**. Jalankan script SQL di bawah ini melalui **Tools->Console** navicat:

```
CREATE TABLE IF NOT EXISTS `login_tb` (
  `user` varchar(30) NOT NULL,
  `password` varchar(32) NOT NULL,
  PRIMARY KEY (`user`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

- Buka menu **File->New->Project**, pilih Template **Windows Application**, beri nama Project dengan **LoginMD5**.
- Klik kanan pada **Form1.vb** di bagian Solution Explorer kemudian ganti namanya menjadi **frmLoginMD5.vb**
- Rancang desain **LoginMD5** beserta kelengkapan kontrolnya seperti ini



- Buka menu **Project->Add Reference** pilih tab **.NET->MySQL.Data**. Referensi MySQL.Data pada dasarnya merujuk ke abstraksi class hasil instalasi MySQL Connector .NET.
- Buka menu **Project-> LoginMD5 Properties->References** kemudian beri tanda centang pada beberapa item *namespace MySQL.Data* antara lain: **MySql**, **MySQL.Data**, **MySQL.Data.MySqlClient**, **MySQL.Data.MySqlClient.Properties**, dan **MySQL.Data.Types**.
- Kode program lengkapnya adalah sebagai berikut:

frmLoginMD5.vb

```
Imports System
Imports System.Security.Cryptography
Imports System.Text

Public Class frmLoginMD5
    'Deklarasi variable object Connection ke MySQL
```

```

Private CN As New MySqlConnection

Function getMd5Hash(ByVal input As String) As String
    'Menciptakan instan object MD5
    Dim md5Hasher As MD5 = MD5.Create()

    'Mengkonversi input string ke array byte dan menghitung hash
    Dim data As Byte() =
    md5Hasher.ComputeHash(Encoding.Default.GetBytes(input))

    ' Create a new StringBuilder to collect the bytes
    ' and create a string.
    Dim sBuilder As New StringBuilder()

    ' Perulangan setiap byte data yang ter-hash
    ' dan memformatnya menjadi string heksadesimal
    Dim i As Integer
    For i = 0 To data.Length - 1
        sBuilder.Append(data(i).ToString("x2"))
    Next i

    'mengembalikan nilai heksadesimal
    Return sBuilder.ToString()
End Function

Function verifyMd5Hash(ByVal input As String, ByVal hash As String) As Boolean
    Dim hashOfInput As String = getMd5Hash(input)

    'Perbandingan string
    Dim comparer As StringComparer =
    StringComparer.OrdinalIgnoreCase
    If 0 = comparer.Compare(hashOfInput, hash) Then
        Return True
    Else
        Return False
    End If
End Function

Private Sub btnLogin_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnLogin.Click
    Dim myConnectionString As String
    myConnectionString = "Database=dbpenjualanbuku;Data
Source=localhost;User Id=root;Password="
    CN = New MySqlConnection(myConnectionString)
    CN.Open()

    Dim DataReader As MySqlDataReader = Nothing
    Dim CMDTampilData As MySqlCommand = CN.CreateCommand()
    Try
        Dim strSQL As String = "SELECT * FROM login_tb WHERE
user=@user AND password=@password"
        CMDTampilData.Parameters.Add("@user",

```

```

MySqlDbType.VarChar, 30).Value = txtUser.Text
        CMDTampilData.Parameters.Add("@password",
MySqlDbType.VarChar, 32).Value = getMd5Hash(txtPassword.Text)
        CMDTampilData.CommandText = strSQL
        DataReader = CMDTampilData.ExecuteReader()

        If DataReader.Read() Then
            'Lakukan sesuatu di sini, seperti masuk ke proses
selanjutnya
            'Biasanya adalah ke Form MDI atau ke kaa palikasi
yang lebih dalam
            MessageBox.Show("User Anda: " &
DataReader("user").ToString & vbCrLf & "Password Asli Anda: " &
txtPassword.Text & vbCrLf & "User Tersandikan : " &
DataReader("password").ToString & vbCrLf & "telah sesuai dan
terdaftar di database", "Konfirmasi Password", MessageBoxButtons.OK,
MessageBoxIcon.Information)
        Else
            MessageBox.Show("User tidak terdaftar", "Konfirmasi
Password", MessageBoxButtons.OK, MessageBoxIcon.Warning)
        End If
        DataReader.Close()
    Catch ex As Exception
        MessageBox.Show(ex.Message, "Terjadi Kegagalan!", MessageBoxButtons.OK,
MessageBoxIcon.Error)
    Finally
        CMDTampilData.Dispose()
        CN.Close()
        CN = Nothing
    End Try
End Sub

Private Sub frmLoginMD5_Load(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles MyBase.Load
    Me.Text = "Login Table MD5"
    txtPassword.PasswordChar = Chr(12)
End Sub

Private Sub btnUserBaru_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles btnUserBaru.Click
    Dim myConnectionString As String
    myConnectionString = "Database=dbpenjualanbuku;Data
Source=localhost;User Id=root;Password="
    CN = New MySqlConnection(myConnectionString)
    CN.Open()

    Dim CMDLogin As MySqlCommand = CN.CreateCommand
    Try
        Dim strSQL As String = "INSERT INTO login_tb (user,
password) VALUES (@user, @password)"
        With CMDLogin
            .CommandText = strSQL
            .Connection = CN

```

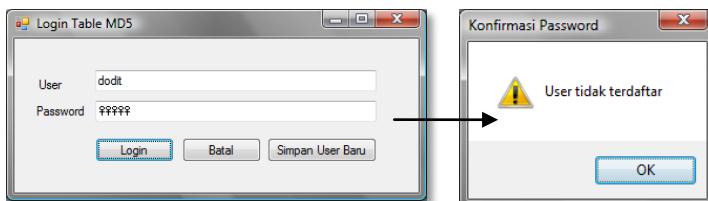
```

        .Parameters.Add("@user", MySqlDbType.VarChar,
30).Value = txtUser.Text
        .Parameters.Add("@password", MySqlDbType.VarChar,
32).Value = getMd5Hash(txtPassword.Text)
        .ExecuteNonQuery()
    End With
    txtUser.Text = ""
    txtPassword.Text = ""
Catch ex As Exception
    MessageBox.Show(ex.Message, "Terjadi Kegagalan!",
MessageBoxButtons.OK, MessageBoxIcon.Error)
Finally
    CMDLogin.Dispose()
    CN.Close()
    CN = Nothing
End Try
End Sub

Private Sub btnBatal_Click(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles btnBatal.Click
    Me.Close()
End Sub
End Class

```

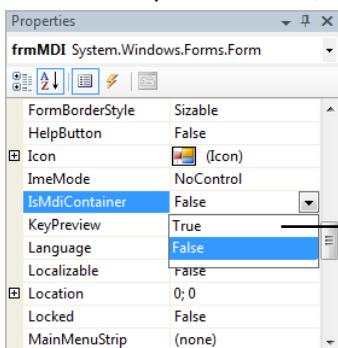
8. Sekarang jalankan programnya dengan menekan tombol keyboard **F5** atau **Ctrl+F5**. Jika user belum terbentuk, silakan isi kotak teks **user** dan **password**, kemudian klik tombol **Simpan User Baru**. Lakukan pengecekan terhadap user dan paswword dengan cara memasukkan ulang user dan password pada kotak teks dengan nilai yang sama. Berikut ini kemungkinan hasil yang akan muncul:



5.3 Form MDI Dan Menu Pulldown (Project MDI)

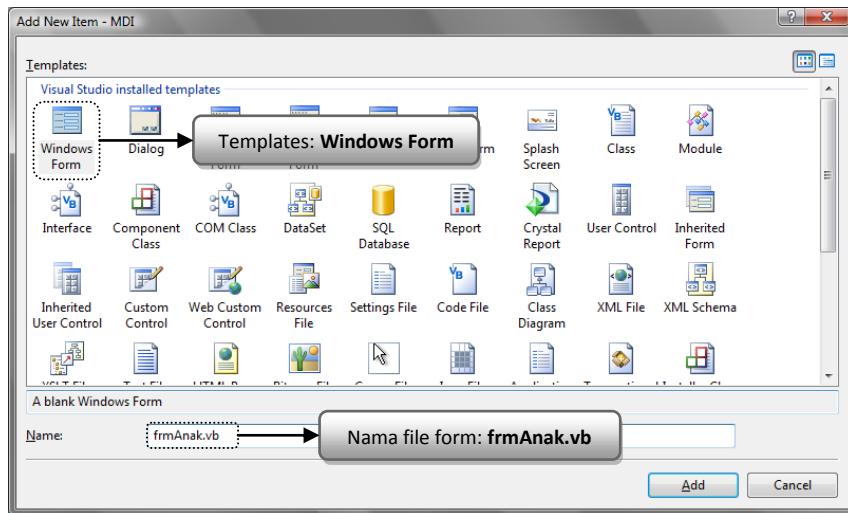
Form MDI adalah form induk yang mewadahi form-form anak, form anak tidak dapat keluar dari form MDI saat ditampilkan sehingga organisasi form aplikasi menjadi lebih tertata.

1. Buka menu **File->New->Project**, pilih Template **Windows Application**, beri nama Project dengan **MDI**. Kemudian klik tombol **Add**.
2. Klik kanan pada **Form1.vb** di bagian Solution Explorer, ganti namanya menjadi **frmMDI.vb**
3. Klik satu kali pada **frmMDI**, ubah **IsMDIContainer** menjadi **True**.

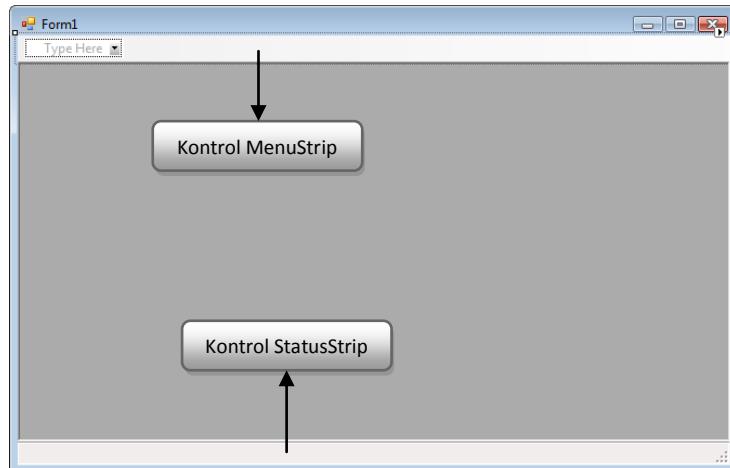


Pada property **frmMDI** ubah nilai **IsMdiContainer=True**

Tambahkan satu form baru yang akan digunakan sebagai form anak. Form anak akan dipanggil dari menu *pull down*. Buka menu **Project->Add Windows Form**, pilih Template **Windows Form**, beri nama file dengan **frmAnak.vb**

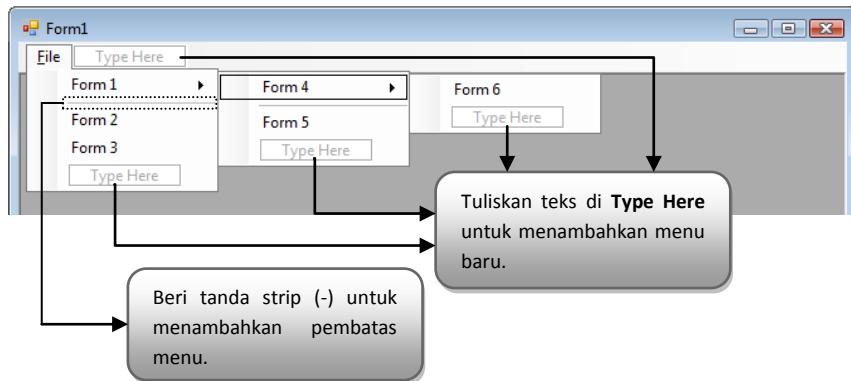


4. Tambahkan kontrol **MenuStrip**, **StatusStrip**, dan **Timer** dari **Toolbox** ke **frmMDI** dengan cara *drag drop* (klik kiri satu kali pada salah satu kontrol, geser ke **frmMDI** kemudian lepaskan).

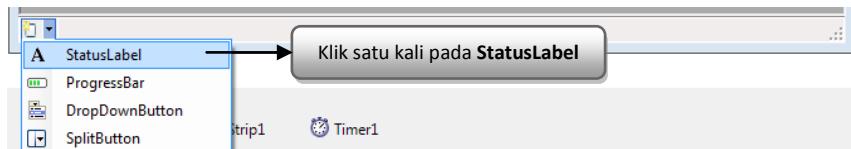


Untuk membuat menu dengan kontrol **MenuStrip**, cantumkan teks pada penanda **Type Here**. Untuk memberi pembatas berupa garis pada

setiap item menu maka tambahkan karakter **strip (-)**. Cara tersebut berlaku pula bagi sub menu lainnya.



Pada **StatusStrip** tambahkan satu label yang akan menampilkan tanggal dan waktu saat ini. Klik sekali pada **StatusStrip** kemudian pilih **StatusLabel** seperti gambar di bawah ini:



5. Memberi kode program pada **frmMDI**.

Klik satu kali pada form **frmMDI**. Pilih **Properties->Events->Load** kemudian tuliskan kode program berikut

Event **frmMDI_Load**

```
Private Sub frmMDI_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load
    'Menjadikan frmMDI full screen
    Me.WindowState = FormWindowState.Maximized
    'Form berlaku sebagai MDI (form parent)
    Me.IsMdiContainer = True
    'Judul form
    Me.Text = "Demo Form MDI"
    'Mengaktifkan tanggal dan waktu
    Me.Timer1.Enabled = True
End Sub
```

Klik dua kali pada kontrol **Timer**  dan masukkan kode program berikut.

Event Timer1_Tick

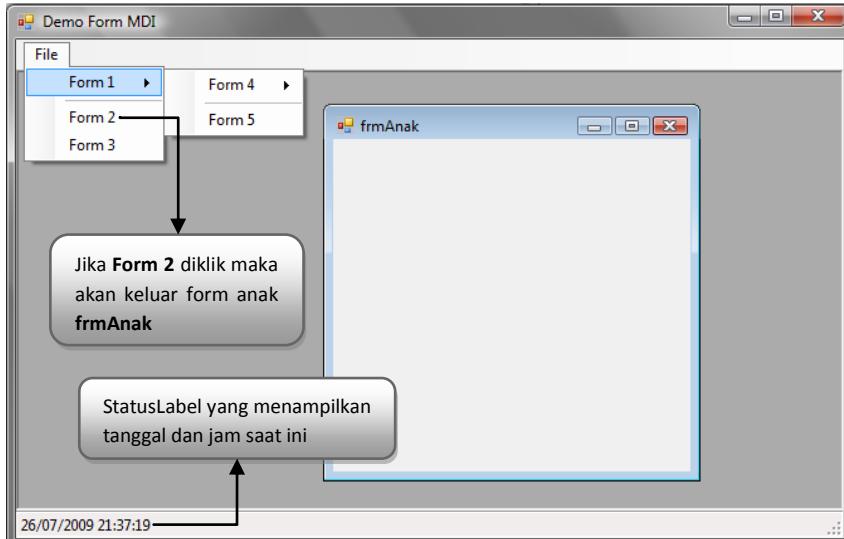
```
Private Sub Timer1_Tick(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Timer1.Tick
    'Status label index ke-0 tanggal dan waktu
    Me.StatusStrip1.Items(0).Text = Now()
End Sub
```

Klik dua kali pada menu **Form 2** pada **MenuStrip** kemudian masukkan kode program berikut.

Event Form2ToolStripMenuItem_Click

```
Private Sub Form2ToolStripMenuItem_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Form2ToolStripMenuItem.Click
    'Pemanggilan form anak frmAnak.vb dari form MDI
    Dim fAnak As frmAnak = New frmAnak
    fAnak.MdiParent = Me
    fAnak.Show()
End Sub
```

6. Sekarang jalankan programnya dengan menekan tombol keyboard **F5** atau **Ctrl+F5**. Kemudian klik menu **Form 2** maka form anak frmAnak akan tampil.

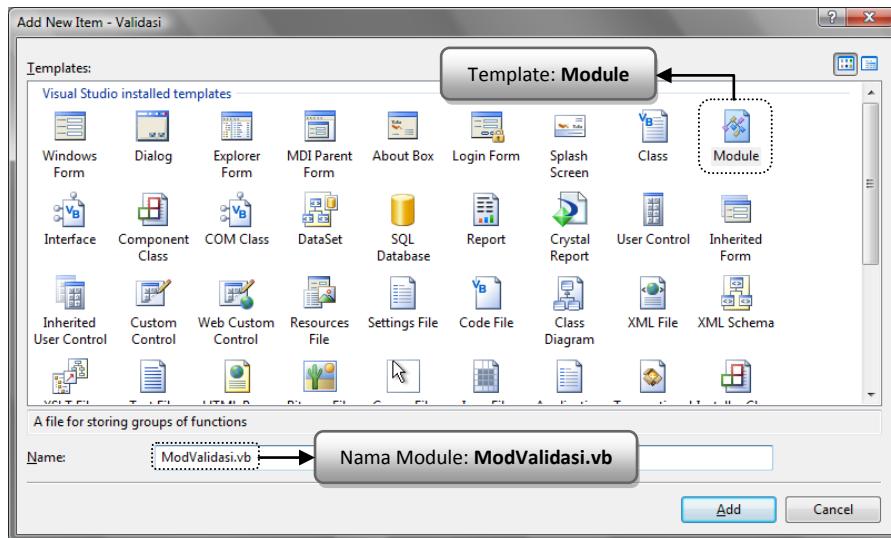


5.4 Form Validasi Email, URL, Numerik, Huruf, Dan Tanggal (Project Validasi)

Untuk membuat aplikasi database yang handal kita harus memperhatikan keabsahan data sebelum dimasukkan ke dalam table sehingga keutuhan data tetap terjaga. Sebagai contoh, kolom harga buku tentu tidak boleh diisi dengan huruf, kolom nama supplier juga tidak mungkin diisi dengan angka, format alamat email dan alamat website harus benar karena ada standarnya. Kita juga perlu memperhatikan sistem penanggalan karena database hanya menerima format **Tahun-Bulan-Tanggal** (YYYY-MM-DD), sedangkan pada sisi program format penanggalan bisa berbeda-beda tergantung dari pengaturan waktu (*time zone*) sistem operasi.

Program aplikasi dapat mendeteksi keabsahan string melalui pola pencocokan terhadap string masukkan atau lebih dikenal dengan **Regular Expressions**, pesan peringatan akan dimunculkan jika pola tidak cocok.

1. Buka menu **File->New->Project**, pilih Template **Windows Application**, beri nama Project dengan **Validasi**.
2. Kita akan mengelompokkan beberapa class validasi dalam file module. Buka menu **Project->Add Module**, pilih Template dengan **Module**, beri nama dengan **ModValidasi.vb**. Kemudian klik tombol **Add**.



Masukkan kode program berikut ini pada module **ModValidasi.vb**:

ModValidasi.vb

```

Imports System.Text.RegularExpressions

Module ModValidasi
    Public Sub Web_Valid(ByVal Karakter As String, ByVal tb As
Object)
        '
        'pengecekan keabsahan input email
        'melalui pola pencocokan Regular Expression
        '
        Dim webPattern As String = "^(https?:\/\/([\w-]+\.)+[\w-]+(/
[\w-./?%]=)*$)"

        If Regex.IsMatch(Karakter.ToString, webPattern) = True Then
            tb.BackColor = Color.White
        Else
            tb.BackColor = Color.Gold
            Beep()
            MessageBox.Show("Inputan tidak valid, mohon dicek
ulang!", "Terjadi Error", MessageBoxButtons.OK, MessageBoxIcon.Error)
            tb.Focus()
        End If
    End Sub

    Public Sub Email_Valid(ByVal Karakter As String, ByVal tb As
Object)
        '
        'pengecekan keabsahan input email
    End Sub

```

```

'melalui pola pencocokan Regular Expression
'-----
Dim emailPattern As String = "^( [0-9a-zA-Z]+[-._+&])*[0-9a-
zA-Z]+@[0-9a-zA-Z]+[.]+[a-zA-Z]{2,6}$"

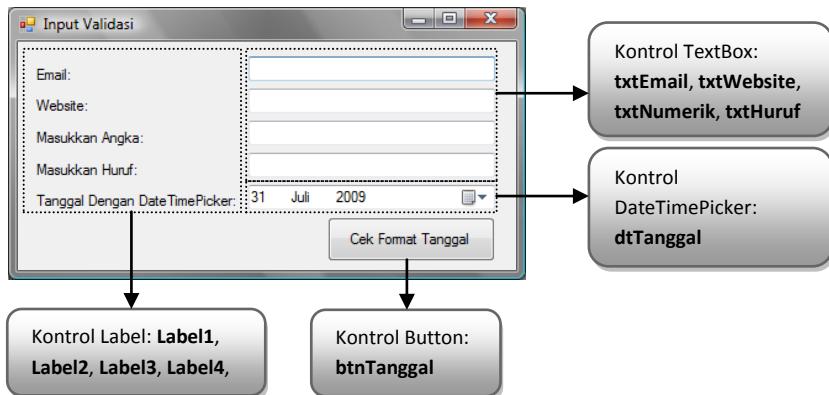
If Regex.IsMatch(Karakter.ToString, emailPattern) = True Then
    tb.BackColor = Color.White
Else
    tb.BackColor = Color.Gold
    Beep()
    MessageBox.Show("Inputan tidak valid, mohon dicek
    ulang!", "Terjadi Error", MessageBoxButtons.OK, MessageBoxIcon.Error)
    tb.Focus()
End If
End Sub

Public Sub Numerik_Valid(ByVal Karakter As String, ByVal tb As
Object)
'-----
'pengecekan keabsahan input numerik
'melalui pola pencocokan Regular Expression
'-----
Dim numDouble As String = "[+-]?([0-9]*\.)?[0-9]+([eE][-
+]?[0-9]+)?"

If Regex.IsMatch(Karakter.ToString, numDouble) = True Then
    tb.BackColor = Color.White
Else
    tb.BackColor = Color.Gold
    Beep()
    MessageBox.Show("Inputan tidak valid, mohon dicek
    ulang!", "Terjadi Error", MessageBoxButtons.OK, MessageBoxIcon.Error)
    tb.Focus()
End If
End Sub
End Module

```

3. Klik kanan pada **Form1.vb** di bagian **Solution Explorer**, ganti nama form menjadi **frmValidasi.vb**. Rancang **frmValidasi.vb** beserta kelengkapan kontrolnya seperti berikut ini:



4. Tuliskan kode program form **frmValidasi.vb** seperti berikut ini:

frmValidasi.vb

```
'namespace penanggalian
Imports System.Globalization

Public Class frmValidasi

    Private Sub frmValidasi_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load
        Me.Text = "Input Validasi"
    End Sub

    Private Sub txtEmail_Leave(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles txtEmail.Leave
        '-----
        'memastikan bahwa inputan mengikuti format email
        '-----
        Email_Valid(txtEmail.Text, txtEmail)
    End Sub

    Private Sub txtWebsite_Leave(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles txtWebsite.Leave
        '-----
        'memastikan bahwa inputan mengikuti format website
        '-----
        Web_Valid(txtWebsite.Text, txtWebsite)
    End Sub

    Private Sub txtNumerik_Leave(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles txtNumerik.Leave
        '-----
        'memastikan bahwa inputan berupa angka
        '-----
        Numerik_Valid(txtNumerik.Text, txtNumerik)
    End Sub

```

```

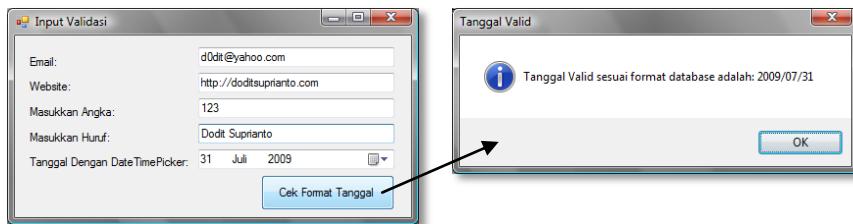
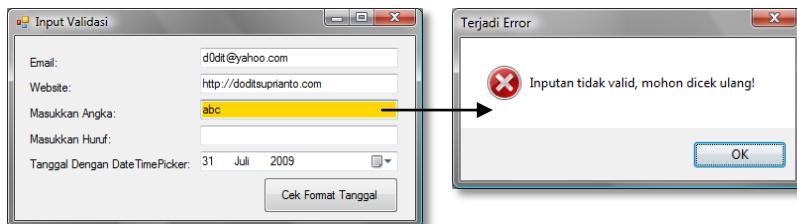
Private Sub txtHuruf_Leave(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles txtHuruf.Leave
    '
    'memastikan bahwa inputan berupa angka
    '
    Alphabet_Valid(txtHuruf.Text, txtHuruf)
End Sub

Private Sub btnTanggal_Click(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles btnTanggal.Click
    Dim Pesan As String
    Pesan = "Tanggal Valid sesuai format database adalah: " &
dtTanggal.Value.ToString("yyyy/MM/dd",
DateTimeFormatInfo.InvariantInfo)
    MessageBox.Show(Pesan, "Tanggal Valid", MessageBoxButtons.OK,
MessageBoxIcon.Information)
End Sub
End Class

```

5. Jalankan program dengan menekan tombol keyboard **F5** atau **Ctrl+F5**.

Masukkan beberapa nilai yang salah pada salah satu kotak teks kemudian tekan tombol **TAB** maka hasilnya adalah seperti berikut:



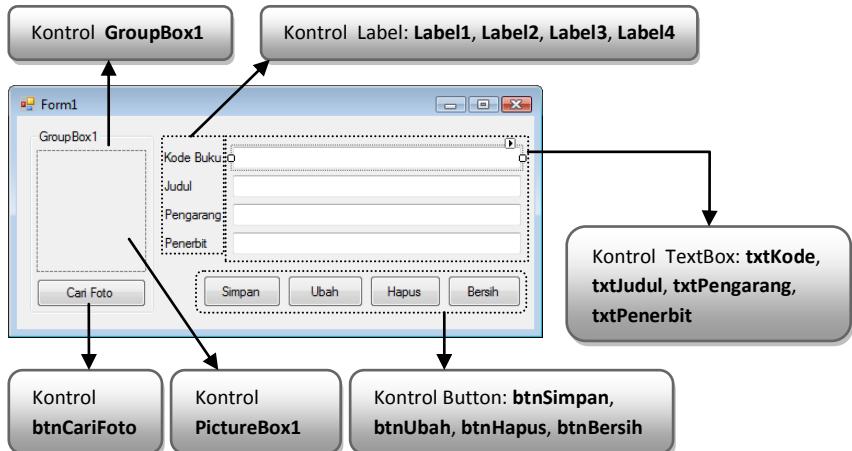
5.5 Form Data Table Disertai Foto (Project FotoBLOB)

Terkadang aplikasi database membutuhkan penyimpanan data berupa foto produk. Ada dua cara untuk menyimpan data foto, pertama disimpan dalam bentuk file sehingga database hanya menyimpan alamat *path* foto saja, kedua data foto secara fisik disimpan dalam database berbentuk data biner atau lebih dikenal dengan BLOB (*Binary Large Object*). Contoh program berikut ini menggunakan data BLOB untuk menyimpan Foto.

1. Buat table **foto_tb** yang diletakkan pada salah satu database yang Anda miliki, dalam kasus ini kita menggunakan database **dbpenjualanbuku**. Jalankan script SQL di bawah ini melalui **Tools->Console** navicat:

```
CREATE TABLE IF NOT EXISTS `foto_tb` (
  `kode_buku` char(10) NOT NULL,
  `judul` tinytext NOT NULL,
  `pengarang` varchar(30) NOT NULL,
  `penerbit` varchar(30) NOT NULL,
  `foto` blob NOT NULL,
  PRIMARY KEY (`kode_buku`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

2. Buka menu **File->New->Project**, pilih Template **Windows Application**, beri nama Project dengan **FotoBLOB**.
3. Klik kanan pada **Form1.vb** di bagian **Solution Explorer** kemudian ganti namanya menjadi **frmFotoBLOB.vb**.
4. Rancang desain **frmFotoBLOB** beserta kelengkapan kontrolnya seperti ini:



5. Buka menu **Project->Add Reference** pilih TAB **.NET->MySQL.Data**.
6. Buka menu **Project->FotoBLOB Properties->References** kemudian beri tanda centang pada beberapa item *namespace MySQL.Data* antara lain: **MySql**, **MySql.Data**, **MySql.Data.MySqlClient**, **MySql.Data.MySqlClient.Properties**, dan **MySql.Data.Types**.
7. Tuliskan kode program lengkap **frmFotoBLOB.vb** seperti di bawah ini:

frmFotoBLOB.vb

```
'Namespace yang berhubungan dengan gambar
Imports System.IO
Imports System.Drawing.Bitmap

Public Class frmFotoBLOB

    Private strImgName As String
    Private imageBytes As Byte = Nothing
    Private FileSize As UInt32
    Private rawData() As Byte
    Private fs As FileStream
    Private CN As New MySqlConnection 'Deklarasi variable object
Connection ke MySQL

    Private Sub frmFotoBLOB_Load(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles MyBase.Load
        GroupBox1.Text = "Foto"
        Me.Text = "Menyimpan Data Dan Foto Ke Table"
    End Sub

    Private Sub Insert_Foto()
        '-----
```

```

-----+
      'mengkonversi file image yang diperoleh dari OpenFileDialog
      'menjadi file bertipe Binary atau BLOB untuk disimpan di
database
      '
-----+
      fs = New FileStream(strImgName, FileMode.Open,
 FileAccess.Read)
      FileSize = fs.Length
      rawData = New Byte(FileSize) {}
      fs.Read(rawData, 0, FileSize)
      fs.Close()
End Sub

Private Sub ClearForm()
    txtKode.Text = ""
    txtJudul.Text = ""
    txtPengarang.Text = ""
    txtPenerbit.Text = ""
    PictureBox1.Image = Nothing

    '
    'Menempatkan cursor keyboard ke txtKodeBuku
    '
txtKode.Focus()

    btnSimpan.Enabled = True
    btnUbah.Enabled = True
    btnHapus.Enabled = True
End Sub

Private Sub btnSimpan_Click(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles btnSimpan.Click
    '
    '
    'menciptakan connection string dan object koneksi ke database
    '
    Dim myConnectionString = "Database=dbpenjualanbuku;Data
Source=localhost;User Id=root;Password="
    CN = New MySqlConnection(myConnectionString)
    CN.Open()
    '
    'memanggil method Insert_Foto()
    '
    Insert_Foto()
    '
    'Deklarasi object command
    '
    Dim CMDInsert As MySqlCommand = CN.CreateCommand

    If txtKode.Text <> "" Then
        Try

```

```

        Dim Sql As String = "INSERT INTO foto_tb (kode_buku,
judul, pengarang, penerbit, foto)"
        Sql &= "VALUES (@kode_buku, @judul, @pengarang,
@penerbit, @foto);"
        With CMDInsert
            .CommandText = Sql
            .Connection = CN
            '
            '-----'
            'Penentuan parameter sekaligus memberi nilai
terhadap parameter tersebut
            '
            '-----'

            .Parameters.Add("@kode_buku", MySqlDbType.String,
10).Value = txtKode.Text
            .Parameters.Add("@judul",
MySqlDbType.TinyText).Value = txtJudul.Text
            .Parameters.Add("@pengarang",
MySqlDbType.VarChar, 30).Value = txtPengarang.Text
            .Parameters.Add("@penerbit",
MySqlDbType.VarChar,
30).Value = txtPenerbit.Text
            .Parameters.Add("@foto", MySqlDbType.Blob).Value
= rawData
            '
            '-----'
            'Mengeksekusi Query dia atas
            '
            .ExecuteNonQuery()
        End With
        '
        '-----'
        'memanggil method ClearForm() untuk membersihkan Form
        '

        '
        ClearForm()
    Catch ex As MySqlException
        '
        'Pesan kesalahan ditampilkan melalui MessageBox.Show
        '
        MessageBox.Show(ex.Message, "Terjadi Kegagalan!",
MessageBoxButtons.OK, MessageBoxIcon.Error)
    Finally
        CMDInsert.Dispose()
        CN.Close()
        CN = Nothing
    End Try
    End If
End Sub

Private Sub btnUbah_Click(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles btnUbah.Click
    '
    'menciptakan connection string dan object koneksi ke database

```

```

-
    Dim myConnectionString = "Database=dbpenjualanbuku;Data
Source=localhost;User Id=root;Password="
    CN = New MySqlConnection(myConnectionString)
    CN.Open()
    '
    'memanggil method Insert_Foto()
    '
Insert_Foto()
'
'Deklarasi object command
'
Dim CMDUbah As MySqlCommand = CN.CreateCommand
If txtKode.Text <> "" Then
    Try
        Dim Sql As String = "UPDATE foto_tb SET judul=@judul,
pengarang=@pengarang, "
        Sql &= "penerbit=@penerbit, foto=@foto WHERE
kode_buku=@kode_buku;"

        With CMDUbah
            .CommandText = Sql
            .Connection = CN
            '
            '
            'Penentuan parameter sekaligus memberi nilai
terhadap parameter tersebut
            '
            '
.Parameters.AddWithValue("@kode_buku", MySqlDbType.String,
10).Value = txtKode.Text
.Parameters.AddWithValue("@judul",
MySqlDbType.TinyText).Value = txtJudul.Text
.Parameters.AddWithValue("@pengarang",
MySqlDbType.VarChar, 30).Value = txtPengarang.Text
.Parameters.AddWithValue("@penerbit", MySqlDbType.VarChar,
30).Value = txtPenerbit.Text
.Parameters.AddWithValue("@foto", MySqlDbType.Blob).Value
= rawData
            '
            '
            'Mengeksekusi Query dia atas
            '
            .ExecuteNonQuery()
        End With
        '
        '
        'memanggil method ClearForm() untuk membersihkan Form
        '
ClearForm()
    Catch ex As MySqlException

```

```

' -----
' Pesan kesalahan ditampilkan melalui MessageBox.Show
' -----
MessageBox.Show(ex.Message, "Terjadi Kegagalan!", MessageBoxButtons.OK, MessageBoxIcon.Error)
Finally
    CMDUbah.Dispose()
    CN.Close()
    CN = Nothing
End Try
End If
End Sub

Private Sub btnHapus_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnHapus.Click
' -----
' menciptakan connection string dan object koneksi ke database
' -----
Dim myConnectionString = "Database=dbpenjualanbuku;Data Source=localhost;User Id=root;Password="""
CN = New MySqlConnection(myConnectionString)
CN.Open()

' -----
' memanggil method Insert_Foto()
' -----
Insert_Foto()

' -----
' Deklarasi object command
' -----
Dim CMDDelete As MySqlCommand = CN.CreateCommand

If txtKode.Text <> "" Then
    If MessageBox.Show("Anda yakin data buku dengan kode " & txtKode.Text & " akan dihapus?", "Peringatan!", MessageBoxButtons.YesNo, MessageBoxIcon.Warning) =
    Windows.Forms.DialogResult.Yes Then
        Try
            Dim Sql As String = "DELETE FROM foto_tb "
            Sql &= "WHERE kode_buku=@kode_buku;"
            With CMDDelete
                .CommandText = Sql
                .Parameters.Add("@kode_buku",
MySqlDbType.String, 10).Value = txtKode.Text
                .ExecuteNonQuery()
            End With
        ' -----
        ' memanggil method ClearForm() untuk membersihkan
        Form
    End If
End If

```

```

        '-----'
        ClearForm()
    Catch ex As MySqlException
        MessageBox.Show(ex.Message, "Terjadi
Kegagalan!", MessageBoxButtons.OK, MessageBoxIcon.Error)
    Finally
        CMDDelete.Dispose()
        CN.Close()
        CN = Nothing
    End Try
End If
End If
End Sub

Private Sub TampilData()
    '-----'
    'menciptakan connection string dan object koneksi ke database
    '-----'

    Dim myConnectionString = "Database=dbpenjualanbuku;Data
Source=localhost;User Id=root;Password="
    CN = New MySqlConnection(myConnectionString)
    CN.Open()

    Dim DataReader As MySqlDataReader = Nothing
    Dim CMDTampilData As MySqlCommand = CN.CreateCommand()
    Try
        If txtKode.Text <> "" Then
            '-----'
            'menampilkan data saat cursor keyboard keluar dari
txtKode
            '-----'

            Dim Sql As String = "SELECT * FROM foto_tb WHERE
kode_buku=@kode_buku"
            CMDTampilData.Parameters.Add("@kode_buku",
MySqlDbType.String, 10).Value = txtKode.Text
            CMDTampilData.CommandText = Sql
            DataReader = CMDTampilData.ExecuteReader()

            If DataReader.Read() Then
                txtKode.Text = DataReader("kode_buku").ToString
                txtJudul.Text = DataReader("judul").ToString
                txtPengarang.Text =
DataReader("pengarang").ToString
                txtPenerbit.Text =
DataReader("penerbit").ToString
            End If
        End If
    End Try
End Sub

```

```

        'pengecekan apakah data foto tersedia.
        'jika ada, maka data dikonversi dari BLOB menjadi
image
        'untuk ditampilkan pada object PictureBox1
'-----



        If DataReader("foto") IsNot DBNull.Value Then
            Dim imageBytes() As Byte = Nothing
            imageBytes = CType(DataReader("foto"), 
Byte())
            Dim ms As New MemoryStream(imageBytes)
            Dim bmap As New Bitmap(ms)
            PictureBox1.Image = CType(bmap, Image)
            PictureBox1.SizeMode =
PictureBoxSizeMode.StretchImage
            End If
            btnSimpan.Enabled = False
            btnUbah.Enabled = True
            btnHapus.Enabled = True
        Else
            btnSimpan.Enabled = True
            btnUbah.Enabled = False
            btnHapus.Enabled = False
        End If
    End If

    Catch ex As MySqlException
        MessageBox.Show(ex.Message, "Terjadi Kegagalan!", 
MessageBoxButtons.OK, MessageBoxIcon.Error)
    Finally
        '-----
        'membebaskan semua resource yang telah terpakai
        '-----
        CMDTampilData.Dispose()
        CN.Close()
        CN = Nothing
    End Try
End Sub

Private Sub txtKode_Leave(ByVal sender As System.Object, ByVal e 
As System.EventArgs) Handles txtKode.Leave
    TampilData()
End Sub

Private Sub btnCariFoto_Click(ByVal sender As System.Object, 
ByVal e As System.EventArgs) Handles btnCariFoto.Click
    '
    'method yang terjadi saat button btn_Loadfoto di klik
    '
Dim dlgFile As OpenFileDialog
PictureBox1.Image = Nothing
'
```

```

-----  

'OpenDialog yang mencari file image dengan extension yang  

telah ditentukan  

'-----  

Try  

    dlgFile = New OpenFileDialog  

    dlgFile.Filter = "JPEG Images  

(*.jpg,*.jpeg)|*.jpg;*.jpeg|GIF Images (*.gif)|*.gif|Bitmaps  

(*.bmp)|*.bmp"  

    dlgFile.FilterIndex = 1 'file extension yang ditampilkan  

pertama adalah .jpg  

    PictureBox1.Visible = True  

    If dlgFile.ShowDialog = Windows.Forms.DialogResult.OK  

Then  

    strImgName = dlgFile.FileName  

    PictureBox1.SizeMode =  

PictureBoxSizeMode.StretchImage  

    PictureBox1.Image = Image.FromFile(strImgName)  

    End If  

    Catch ex As Exception  

        MessageBox.Show(ex.Message, "Terjadi kesalahan",  

MessageBoxButtons.OK, MessageBoxIcon.Error)
    End Try
End Sub

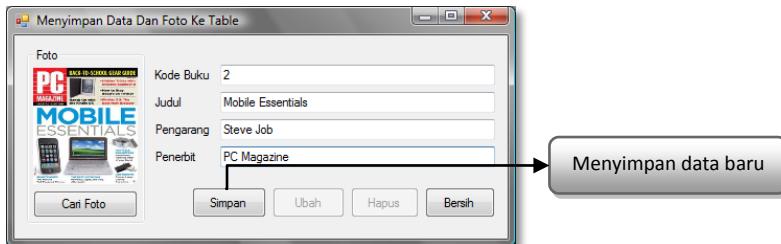
Private Sub btnBersih_Click(ByVal sender As System.Object, ByVal  

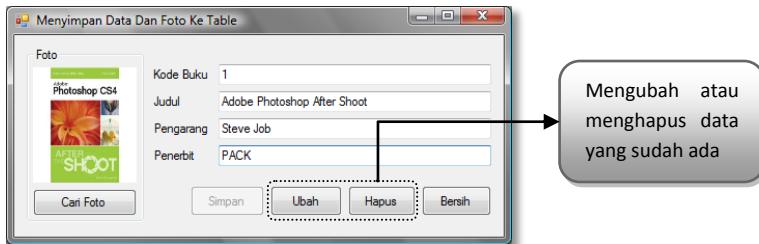
e As System.EventArgs) Handles btnBersih.Click  

    ClearForm()
End Sub
End Class

```

8. Jalankan programnya dengan menekan tombol keyboard **F5** atau **Ctrl+F5**. Hasilnya kurang lebih seperti berikut:





5.6 Form *Browse* Dengan *Drag Drop* (Project *Browse*)

Form *browse* bertujuan untuk memudahkan pencarian data tertentu. Sebagai contoh, bisa dibayangkan betapa sulitnya mencari data buku tertentu yang diperoleh dari ribuan daftar buku yang tersedia karena tidak mungkin bagi seseorang untuk menghafalkan seluruh kode buku yang ada.

Kita akan menambahkan fitur *browse* pada project **FotoBLOB** sebelumnya. Fitur pencarian diklasifikasi berdasarkan kode buku, judul buku, pengarang atau penerbit.

1. Buat store procedure berisi perintah select yang disertai *filter* (penyaringan dengan klausa WHERE). Store procedure berikut ini akan dipanggil pada form **frmBrowse.vb**. Tuliskan melalui program navicat **Tools->Console** atau **Query->New Query** kemudian jalankan.

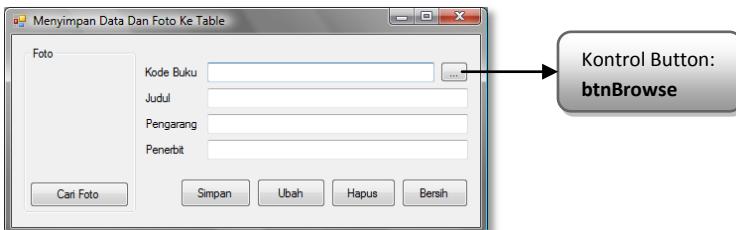
```
DROP PROCEDURE IF EXISTS BROWSE_FOTO_SP;
CREATE PROCEDURE BROWSE_FOTO_SP (IN KATEGORI VARCHAR(10),
                                 IN CARI VARCHAR(100))
BEGIN
    CASE KATEGORI
        WHEN 'JUDUL' THEN
            SELECT kode_buku AS 'KODE', judul AS 'JUDUL', pengarang AS
'PENGARANG', penerbit AS 'PENERBIT', foto AS 'FOTO'
            FROM foto_tb
            WHERE judul LIKE CONCAT('%', CARI, '%')
            ORDER BY judul;
        WHEN 'PENGARANG' THEN
```

```

SELECT kode_buku AS 'KODE', judul AS 'JUDUL', pengarang AS
'PENGARANG', penerbit AS 'PENERBIT', foto AS 'FOTO'
FROM foto_tb
WHERE pengarang LIKE CONCAT('%', CARI, '%')
ORDER BY pengarang;
WHEN 'PENERBIT' THEN
    SELECT kode_buku AS 'KODE', judul AS 'JUDUL', pengarang AS
'PENGARANG', penerbit AS 'PENERBIT', foto AS 'FOTO'
    FROM foto_tb
    WHERE penerbit LIKE CONCAT('%', CARI, '%')
    ORDER BY penerbit;
END CASE;
END;

```

2. Tambahkan kontrol **Button** bernama **btnBrowse** pada form **FotoBLOB.vb** seperti tampak berikut ini



Tambahkan kode program **frmFotoBLOB.vb** pada event **btnBrowse_Click**, **txtKode_DragEnter**, **txtKode_DragDrop** seperti berikut ini

Tambahan Kode **frmFotoBLOB.vb**

```

Private Sub frmFotoBLOB_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load
    txtKode.AllowDrop = True
    GroupBox1.Text = "Foto"
    Me.Text = "Menyimpan Data Dan Foto Ke Table"
End Sub

Private Sub btnBrowse_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnBrowse.Click
    Dim fb As frmBrowse = New frmBrowse
    fb.Show()
End Sub

Private Sub txtKode_DragDrop(ByVal sender As System.Object, ByVal e As System.Windows.Forms.DragEventArgs) Handles txtKode.DragDrop
    txtKode.Text = e.Data.GetData(DataFormats.Text)
    TampilData()
End Sub

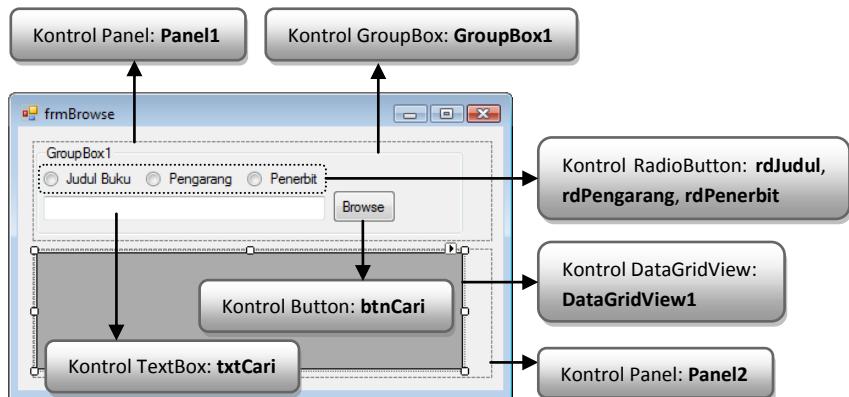
```

```

Private Sub txtKode_DragEnter(ByVal sender As System.Object,
ByVal e As System.Windows.Forms.DragEventArgs) Handles txtKode.DragEnter
    If (e.Data.GetDataPresent(DataFormats.Text)) Then
        e.Effect = DragDropEffects.Copy
    Else
        e.Effect = DragDropEffects.None
    End If
End Sub

```

3. Buka menu **Project->Add Windows Form** beri nama form dengan **frmBrowse.vb**. Rancang desain form beserta kelengkapan kontrolnya seperti berikut ini:



4. Tuliskan kode program berikut pada form **frmBrowse.vb**

frmBrowse.vb

```

Public Class frmBrowse

    Private MouseIsDown As Boolean = False
    'Deklarasi variable object adapter reader
    Private MyAdapter As New MySqlDataAdapter()
    'Deklarasi variable object dataset
    Private MyDataset As New DataSet()
    'Deklarasi variable object Connection ke MySQL
    Private CN As New MySqlConnection

    Private Sub frmBrowse_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load
        Me.Text = "Pencarian Buku"
        Panel1.Dock = DockStyle.Top
        Panel2.Dock = DockStyle.Fill
        GroupBox1.Text = "Dasar Pencarian"
    End Sub

```

```

        GroupBox1.Dock = DockStyle.Fill
        DataGridView1.Dock = DockStyle.Fill
        DataGridView1.ReadOnly = True
End Sub

Private Sub btnCari_Click(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles btnCari.Click
    'menciptakan connection string dan object koneksi ke database
    Dim myConnectionString = "Database=dbpenjualanbuku;Data
Source=localhost;User Id=root;Password="
    CN = New MySqlConnection(myConnectionString)
    CN.Open()

    Dim CMDCariBuku As MySqlCommand = CN.CreateCommand

    Try
        MyDataset.Clear()
        Dim strSQL As String = ""

        If rdJudul.Checked = True Then strSQL = "CALL
BROWSE_PHOTO_SP('JUDUL', @CARI)"
        If rdPengarang.Checked = True Then strSQL = "CALL
BROWSE_PHOTO_SP('PENGARANG', @CARI)"
        If rdPenerbit.Checked = True Then strSQL = "CALL
BROWSE_PHOTO_SP('PENERBIT', @CARI)"

        CMDCariBuku.CommandText = strSQL
        CMDCariBuku.Parameters.Add("@CARI",
MySqlDbType.VarChar).Value = txtCari.Text
        MyAdapter.SelectCommand = CMDCariBuku
        MyAdapter.Fill(MyDataset, "foto_tb")
        DataGridView1.DataSource = MyDataset
        DataGridView1.DataMember = "foto_tb"
        'Lebar kolom Judul
        DataGridView1.Columns(1).Width = 250
        'Lebar kolom Pengarang
        DataGridView1.Columns(2).Width = 250
        'Lebar kolom Penerbit
        DataGridView1.Columns(3).Width = 250
    Catch ex As Exception
        MessageBox.Show(ex.Message, "Terjadi Kegagalan!",
MessageBoxButtons.OK, MessageBoxIcon.Error)
    Finally
        CMDCariBuku.Dispose()
        CN.Dispose()
        CN = Nothing
    End Try
End Sub

Private Sub DataGridView1_MouseDown(ByVal sender As
System.Object, ByVal e As System.Windows.Forms.MouseEventHandler)
Handles DataGridView1.MouseDown
    MouseIsDown = True

```

```

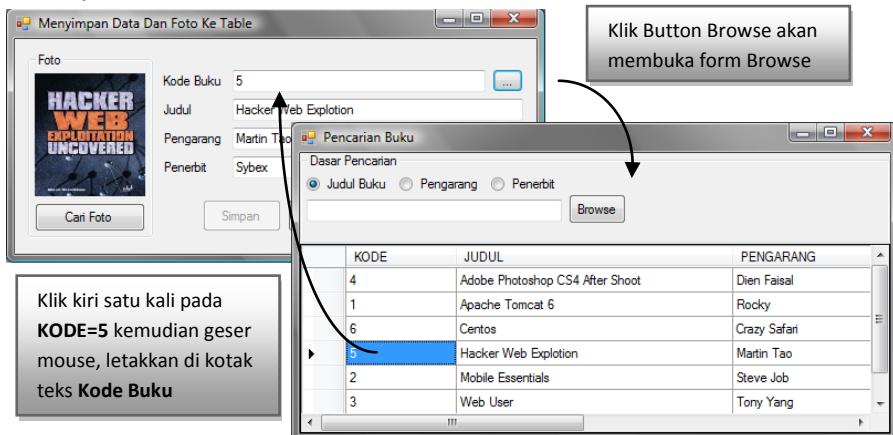
End Sub

Private Sub DataGridView1_MouseMove(ByVal sender As
System.Object, ByVal e As System.Windows.Forms.MouseEventArgs)
Handles DataGridView1.MouseMove
    If MouseIsDown Then
        DataGridView1.DoDragDrop(DataGridView1.Item(0,
DataGridView1.CurrentCell.RowIndex).Value, DragDropEffects.Copy)
    End If
    MouseIsDown = False
End Sub
End Class

```

5. Jalankan program dengan menekan tombol keyboard **F5** atau **Ctrl+F5**.

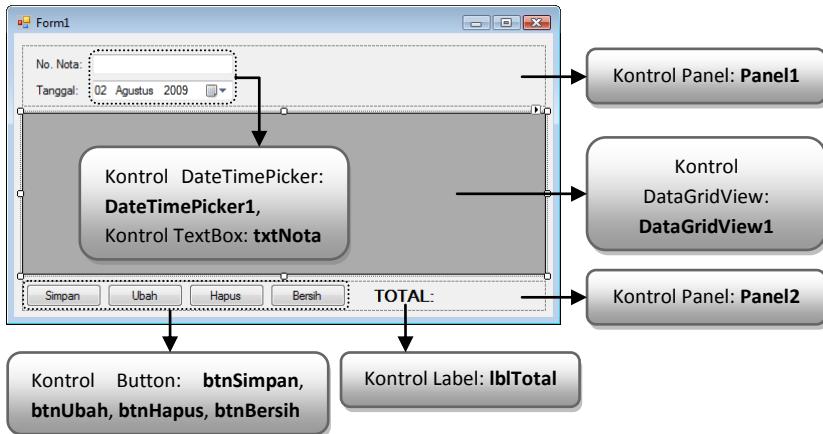
Klik tombol pencarian pada form **frmFotoBLOB** selanjutnya keluar form **frmBrowse**, lakukan pencarian. Pilih salah satu hasilnya pada **DataGridview**, geser dan letakkan di kotak teks **Kode Buku** (drag-drop).



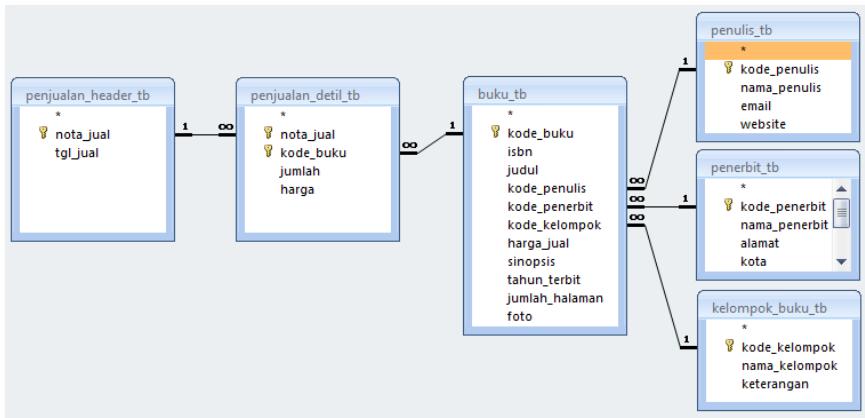
5.7 Form One To Many (Project OneToMany)

Form *one to many* adalah form yang melibatkan lebih dari satu table untuk dimanipulasi dalam satu kali operasi. Contoh nyata penggunaan form *one to many* adalah form transaksi yang terdiri dari *table header* dan *table detail*. Table detail bisa berupa *data grid* yang terdiri dari banyak baris. Konsep *one to many* adalah bagaimana mengoperasikan lebih dari satu table secara sekaligus dalam kondisi sukses atau gagal sekaligus.

1. Buka menu **File->New->Project**, pilih template **Windows Application**, beri nama project-nya dengan **OneToMany**.
2. Ganti nama **form1.vb** menjadi **frmOneToMany.vb**. Rancang desain form beserta kelengkapan kontrolnya seperti berikut ini.



3. Buka menu **Project->Add Reference** pilih TAB .NET->**MySQL.Data**.
4. Buka menu **Project->OneToMany Properties->References** kemudian beri tanda centang pada beberapa item *namespace MySQL.Data* antara lain: **MySql**, **MySql.Data**, **MySql.Data.MySqlClient**, **MySql.Data.MySqlClient.Properties**, dan **MySql.Data.Types**.
5. Table yang digunakan contoh dalam form taransaksi *one-to-many* adalah table **penjualan_header_tb**, **penjualan_detil_tb**, **buku_tb**, **penulis_tb**, **penerbit_tb**, dan **kelompok_buku_tb** dari database **dbpenjualanbuku**. Jika mengacu pada *integrity constraint* database **dbpenjualanbuku**, maka desain tablenya adalah sebagai berikut:



Script SQL struktur table dan relasinya adalah:

```

CREATE TABLE IF NOT EXISTS `buku_tb` (
  `kode_buku` char(13) NOT NULL DEFAULT '',
  `isbn` char(10) DEFAULT NULL,
  `judul` tinytext NOT NULL,
  `kode_penulis` char(10) NOT NULL,
  `kode_penerbit` char(10) NOT NULL,
  `kode_kelompok` char(10) NOT NULL,
  `harga_jual` double NOT NULL DEFAULT '0',
  `sinopsis` text,
  `tahun_terbit` smallint(6) DEFAULT NULL,
  `jumlah_halaman` int(11) DEFAULT NULL,
  `foto` blob,
  PRIMARY KEY (`kode_buku`),
  KEY `Fk_buku_penulis`(`kode_penulis`),
  KEY `Fk_buku_kelompok`(`kode_kelompok`),
  KEY `Fk_buku_penerbit`(`kode_penerbit`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

CREATE TABLE IF NOT EXISTS `kelompok_buku_tb` (
  `kode_kelompok` char(10) NOT NULL DEFAULT '',
  `nama_kelompok` varchar(30) NOT NULL,
  `keterangan` tinytext,
  PRIMARY KEY (`kode_kelompok`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

CREATE TABLE IF NOT EXISTS `penerbit_tb` (
  `kode_penerbit` char(10) NOT NULL DEFAULT '',
  `nama_penerbit` varchar(30) NOT NULL,
  ...
)

```

```

`alamat` varchar(50) DEFAULT NULL,
`kota` varchar(25) DEFAULT NULL,
`kode_pos` varchar(6) DEFAULT NULL,
`telp` varchar(15) DEFAULT NULL,
`email` tinytext,
`website` tinytext,
PRIMARY KEY (`kode_penerbit`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

CREATE TABLE IF NOT EXISTS `penjualan_detil_tb` (
`nota_jual` char(10) NOT NULL DEFAULT '',
`kode_buku` char(13) NOT NULL DEFAULT '',
`jumlah` smallint(11) unsigned NOT NULL DEFAULT '1',
`harga` double unsigned NOT NULL,
PRIMARY KEY (`nota_jual`,`kode_buku`),
KEY `Fk_penjualand_buku` (`kode_buku`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

CREATE TABLE IF NOT EXISTS `penjualan_header_tb` (
`nota_jual` char(10) NOT NULL DEFAULT '',
`tgl_jual` date NOT NULL,
PRIMARY KEY (`nota_jual`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

CREATE TABLE IF NOT EXISTS `penulis_tb` (
`kode_penulis` char(10) NOT NULL DEFAULT '',
`nama_penulis` varchar(30) NOT NULL,
`email` tinytext,
`website` tinytext,
`tentang_penulis` text,
PRIMARY KEY (`kode_penulis`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

ALTER TABLE `buku_tb`
ADD CONSTRAINT `Fk_buku_kelompok` FOREIGN KEY (`kode_kelompok`)
REFERENCES `kelompok_buku_tb` (`kode_kelompok`),
ADD CONSTRAINT `Fk_buku_penerbit` FOREIGN KEY (`kode_penerbit`)
REFERENCES `penerbit_tb` (`kode_penerbit`),
ADD CONSTRAINT `Fk_buku_penulis` FOREIGN KEY (`kode_penulis`)
REFERENCES `penulis_tb` (`kode_penulis`);

ALTER TABLE `penjualan_detil_tb`
ADD CONSTRAINT `Fk_penjualand_buku` FOREIGN KEY (`kode_buku`)
REFERENCES `buku_tb` (`kode_buku`),
ADD CONSTRAINT `Fk_penjualanh_penjualand` FOREIGN KEY
(`nota_jual`) REFERENCES `penjualan_header_tb` (`nota_jual`);

```

Buat view **TAMPIL_ITEM_BUKU_VW** yang akan menampilkan data buku saat kode buku ditulis pada cell kontrol **DataGridview**.

```
DROP VIEW IF EXISTS TAMPIL_ITEM_BUKU_VW;
CREATE VIEW TAMPIL_ITEM_BUKU_VW
AS
SELECT buku_tb.kode_buku, buku_tb.judul,
penerbit_tb.nama_penerbit, penulis_tb.nama_penulis,
kelompok_buku_tb.nama_kelompok, buku_tb.harga_jual
FROM penulis_tb INNER JOIN (penerbit_tb INNER JOIN
(kelompok_buku_tb INNER JOIN buku_tb ON
kelompok_buku_tb.kode_kelompok = buku_tb.kode_kelompok) ON
penerbit_tb.kode_penerbit = buku_tb.kode_penerbit) ON
penulis_tb.kode_penulis = buku_tb.kode_penulis;
```

Buat view **TAMPIL_JUAL_VW** yang akan menampilkan baris-baris data penjualan.

```
DROP VIEW IF EXISTS TAMPIL_JUAL_VW;
CREATE VIEW TAMPIL_JUAL_VW
AS
SELECT penjualan_detil_tb.kode_buku AS Kode, buku_tb.judul AS
'Judul Buku', penerbit_tb.nama_penerbit AS Penerbit,
penulis_tb.nama_penulis AS Penulis, kelompok_buku_tb.nama_kelompok
AS Kelompok, penjualan_detil_tb.harga AS Harga,
penjualan_detil_tb.jumlah AS Jumlah,
penjualan_detil_tb.harga*penjualan_detil_tb.jumlah AS 'Sub Total',
penjualan_header_tb.nota_jual, penjualan_header_tb.tgl_jual
FROM kelompok_buku_tb INNER JOIN (penulis_tb INNER JOIN
(penerbit_tb INNER JOIN (penjualan_header_tb INNER JOIN (buku_tb
INNER JOIN penjualan_detil_tb ON buku_tb.kode_buku =
penjualan_detil_tb.kode_buku) ON penjualan_header_tb.nota_jual =
penjualan_detil_tb.nota_jual) ON penerbit_tb.kode_penerbit =
buku_tb.kode_penerbit) ON penulis_tb.kode_penulis =
buku_tb.kode_penulis) ON kelompok_buku_tb.kode_kelompok =
buku_tb.kode_kelompok;
```

Catatan: Jika Anda merasa kesulitan membuat script SQL view yang melibatkan banyak table, ada baiknya Anda membuat struktur table beserta relasinya di **Microsoft Access** kemudian memanfaatkan fitur **Relationships** dan view **Query Design** untuk men-generate script SQL. Setelah terbentuk script SQL barulah Anda gandakan (*copy - paste*) ke

program Query Navicat. Khusus Navicat berbayar telah disediakan *Query Builder* yang fungsinya kurang lebih sama dengan fitur pada Microsoft Access.

Buat store procedure **DEMO_PENJUALAN_SP** yang bertujuan mengoperasikan perintah INSERT, UPDATE dan DELETE pada table **penjualan_header_tb** dan **penjualan_detail_tb** secara sekaligus.

```
DROP PROCEDURE IF EXISTS DEMO_PENJUALAN_SP;
CREATE PROCEDURE DEMO_PENJUALAN_SP
(
    IN pilihan VARCHAR(20),
    IN vnota_jual CHAR(10),
    IN vtgl_jual DATE,
    IN vdetil TEXT
)
BEGIN
    DECLARE i INT DEFAULT 1;
    DECLARE kolom INT DEFAULT 0;
    DECLARE awal INT DEFAULT 0;
    DECLARE lebar INT DEFAULT 0;
    DECLARE lebar_string INT;

    DECLARE vkode_buku VARCHAR(13) DEFAULT '';
    DECLARE vjumlah INT DEFAULT 0;
    DECLARE vharga DOUBLE DEFAULT 0;

    DECLARE error BOOLEAN DEFAULT FALSE;
    SET AUTOCOMMIT = 0;
    START TRANSACTION;

    SET lebar_string = LENGTH(vdetil);

    CASE pilihan
    WHEN 'INSERT' THEN
        INSERT INTO penjualan_header_tb (nota_jual, tgl_jual)
        VALUES (vnota_jual, vtgl_jual);
        IF error THEN
            ROLLBACK;
            SELECT ('Transaksi gagal!');
        ELSE
            COMMIT;
        END IF;
```

```

WHILE (i <= lebar_string) DO
IF SUBSTR(vdetil, i, 1) = CHAR(21) THEN

    SET kolom=kolom+1;
    IF awal=0 THEN
        SET lebar = i - 1;
    ELSE
        SET lebar = i - awal - 1;
    END IF;
    IF kolom=1 THEN
        SET vkode_buku = SUBSTR(vdetil, awal + 1, lebar);
    ELSEIF kolom=2 THEN
        SET vharga = SUBSTR(vdetil, awal + 1, lebar);
    ELSEIF kolom=3 THEN
        SET vjumlah = SUBSTR(vdetil, awal + 1, lebar);
        SET kolom=0;
        INSERT INTO penjualan_detil_tb (nota_jual, kode_buku,
jumlah, harga)
        VALUES (vnota_jual, vkode_buku, vjumlah, vharga);

        IF error THEN
            ROLLBACK;
            SELECT('Transaksi digagalkan!');
        ELSE
            COMMIT;
        END IF;
    END IF;
    SET awal = i;
END IF;
SET i = i + 1;
END WHILE;

WHEN 'UPDATE' THEN
    UPDATE penjualan_header_tb
    SET tgl_jual = vtgl_jual
    WHERE nota_jual = vnota_jual;
    IF error THEN
        ROLLBACK;
        SELECT('Transaksi digagalkan!');
    ELSE
        COMMIT;
    END IF;

    WHILE (i <= lebar_string) DO
    IF SUBSTR(vdetil, i, 1) = CHAR(21) THEN

```

```

SET kolom=kolom+1;
IF awal=0 THEN
    SET lebar = i - 1;
ELSE
    SET lebar = i - awal - 1;
END IF;
IF kolom=1 THEN
    SET vkode_buku = SUBSTR(vdetil, awal + 1, lebar);
ELSEIF kolom=2 THEN
    SET vharga = SUBSTR(vdetil, awal + 1, lebar);
ELSEIF kolom=3 THEN
    SET vjumlah = SUBSTR(vdetil, awal + 1, lebar);
    SET kolom=0;
    UPDATE penjualan_detil_tb
    SET jumlah = vjumlah, harga = vharga
    WHERE nota_jual = vnota_jual AND kode_buku =
vkode_buku;
    IF error THEN
        ROLLBACK;
        SELECT('Transaksi digagalkan!');
    ELSE
        COMMIT;
    END IF;
END IF;
    SET awal = i;
END IF;
SET i = i + 1;
END WHILE;

WHEN 'DELETE' THEN
    DELETE FROM penjualan_detil_tb
    WHERE nota_jual = vnota_jual;
    IF error THEN
        ROLLBACK;
        SELECT('Transaksi digagalkan!');
    ELSE
        COMMIT;
    END IF;

    DELETE FROM penjualan_header_tb
    WHERE nota_jual = vnota_jual;
    IF error THEN
        ROLLBACK;
        SELECT('Transaksi digagalkan!');
    ELSE
        COMMIT;

```

```
    END IF;
END CASE;
END;
```

6. Tuliskan kode program pada form **frmOneToMany.vb** berikut ini:

frmOneToMany.vb

```
Imports System.Globalization

Public Class frmOneToMany

    Private ds As DataSet = New DataSet
    Private CN As New MySqlConnection

    Private Sub frmOneToMany_Load(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles MyBase.Load
        Panel1.Dock = DockStyle.Top
        Panel2.Dock = DockStyle.Bottom
        DataGridView1.Dock = DockStyle.Fill
        DataGridView1.BorderStyle = BorderStyle.None
        Me.Text = "Form One To Many"
        ClearForm()
    End Sub

    Private Sub ClearForm()
        ' -----
        ' Membersihkan nilai yang masih ada
        ' pada object form dengan string kosong
        ' -----
        txtNota.Text = ""
        dtTanggal.Value = Now
        ds.Clear()
        ' -----
        ' Menempatkan cursor keyboard ke txtNota
        ' -----
        txtNota.Focus()
    End Sub

    Private Sub btnBersih_Click(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles btnBersih.Click
        ' -----
        ' memanggil method ClearForm()
        ' -----
        ClearForm()

        ' -----
        ' mengaktifkan Button btnSimpan, btnUbah, btnHapus
        ' -----
        btnSimpan.Enabled = True
        btnUbah.Enabled = True
        btnHapus.Enabled = True
    End Sub
```

```

Private Sub btnSimpan_Click(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles btnSimpan.Click
    '
    '
    'menciptakan connection string dan object koneksi ke database
    '

    Dim myConnectionString = "Database=dbpenjualanbuku;Data
Source=localhost;User Id=root;Password="
    CN = New MySqlConnection(myConnectionString)
    CN.Open()

    '
    'Deklarasi object transaction dan command
    '

    Dim SQLTrans As MySqlTransaction = CN.BeginTransaction
    Dim CMDInsert As MySqlCommand = CN.CreateCommand

    If txtNota.Text <> "" Then
        Try
            With CMDInsert
                Dim Detil As String = ""
                Dim Baris, TotBaris As Integer
                TotBaris = DataGridView1.RowCount - 2
                Detil = ""
                'menggabungkan nilai pada kolom ke-0 (kode
buku), ke-5(harga)
                'dan ke-6(jumlah) yang dibatasi oleh tanda |.
                'gabungan string ini akan diuraikan pada MySQL
                'untuk ditempatkan
                'penjualan_detil_tb
                For Baris = 0 To TotBaris
                    Detil &= DataGridView1.Item(0,
Baris).Value.ToString & Chr(21) & DataGridView1.Item(5,
Baris).Value.ToString & Chr(21) & DataGridView1.Item(6,
Baris).Value.ToString & Chr(21)
                Next

                'pemanggilan PENJUALAN_SP sekaligus memasukkan
                'nilai-nilai parameternya
                Dim SQLD As String = "CALL DEMO_PENJUALAN_SP
('INSERT', @no_nota, @tgl_jual, @detil); "
                .Parameters.Add("@no_nota", MySqlDbType.VarChar,
10).Value = txtNota.Text
                .Parameters.Add("@tgl_jual",
MySqlDbType.Date).Value = dtTanggal.Value.ToString("yyyy/MM/dd",
DateTimeFormatInfo.InvariantInfo)
                .Parameters.Add("@detil", MySqlDbType.Text).Value
= Detil
                .CommandText = SQLD
                .Transaction = SQLTrans
                .ExecuteNonQuery()
            End With
        Catch ex As Exception
            SQLTrans.Rollback()
            MessageBox.Show(ex.Message)
        End Try
    End If
End Sub

```

```

        End With

        ' -----
        ' Jika sukses dilaksanakan, maka perubahan dibuat
        ' permanen
        ' melalui perintah COMMIT transaction
        ' -----

        SQLTrans.Commit()

        ' -----
        ' memanggil method ClearForm() untuk membersihkan Form
        ' -----

        ClearForm()

    Catch ex As MySqlException
        ' -----
        ' Jika terjadi kegagalan, maka eksekusi dibatalkan
        ' melalui perintah ROLLBACK transaction
        ' -----

        SQLTrans.Rollback()

        ' -----
        ' Pesan kesalahan ditampilkan melalui MessageBox.Show
        ' -----

        MessageBox.Show(ex.Message, "Terjadi Kegagalan!",
MessageBoxButtons.OK, MessageBoxIcon.Error)
    Finally
        ' -----
        ' melalui try...finally, maka baik gagal atau sukses,
        ' maka SQLTrans dan CMDInsert dimusnahkan
        ' sehingga kedua resource tersebut dilepas
        ' -----

        SQLTrans.Dispose()
        CMDInsert.Dispose()
        CN.Close()
        CN = Nothing
    End Try
End If
End Sub

Private Sub btnUbah_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnUbah.Click
    ' -----
    ' menciptakan connection string dan object koneksi ke database
    ' -----

    Dim myConnectionString = "Database=dbpenjualanbuku;Data Source=localhost;User Id=root;Password="

```

```

CN = New MySqlConnection(myConnectionString)
CN.Open()

'-----
'Deklarasi object transaction dan command
'-----

Dim SQLTrans As MySqlTransaction = CN.BeginTransaction
Dim CMDUpdate As MySqlCommand = CN.CreateCommand

If txtnota.Text <> "" Then
    Try
        With CMDUpdate
            Dim Detil As String = ""
            Dim Baris, TotBaris As Integer
            TotBaris = DataGridView1.RowCount - 2
            Detil = ""
            'menggabungkan nilai pada kolom ke-0 (kode
buku), ke-5 (harga)
            'dan ke-6(jumlah) yang dibatasi oleh tanda |.
            'gabungan string ini akan diuraikan pada MySQL
            'pada kolomnya masing-masing pada table
penjualan_detil_tb
            For Baris = 0 To TotBaris
                Detil &= DataGridView1.Item(0,
Baris).Value.ToString & Chr(21) & DataGridView1.Item(5,
Baris).Value.ToString & Chr(21) & DataGridView1.Item(6,
Baris).Value.ToString & Chr(21)
            Next

            'pemanggilan PENJUALAN_SP sekaligus memasukkan
nilai-nilai parameternya
            Dim SQLD As String = "CALL DEMO_PENJUALAN_SP
('UPDATE', @no_nota, @tgl_jual, @detil); "
            .Parameters.Add("@no_nota", MySqlDbType.VarChar,
10).Value = txtNota.Text
            .Parameters.Add("@tgl_jual",
MySqlDbType.Date).Value = dtTanggal.Value.ToString("yyyy/MM/dd",
DateTimeFormatInfo.InvariantInfo)
            .Parameters.Add("@detil", MySqlDbType.Text).Value
= Detil
            .CommandText = SQLD
            .Transaction = SQLTrans
            .ExecuteNonQuery()
        End With

'-----
'Jika sukses dilaksanakan, maka perubahan dibuat
permanen
'melalui perintah COMMIT transaction
'-----

```

```

        SQLTrans.Commit()

        '
        '-----'
        'memanggil method ClearForm() untuk membersihkan Form
        '-----'

        ClearForm()
        Catch ex As MySqlException
        '
        'Jika terjadi kegagalan, maka eksekusi dibatalkan
        'melalui perintah ROLLBACK transaction
        '
        SQLTrans.Rollback()

        '
        'Pesan kesalahan ditampilkan melalui MessageBox.Show
        '
        MessageBox.Show(ex.Message, "Terjadi Kegagalan!",
        MessageBoxButtons.OK, MessageBoxIcon.Error)
        Finally
        '
        'melalui try...finally, maka baik gagal atau sukses.
        'maka SQLTrans dan CMDInsert dimusnahkan
        'sehingga kedua resource tersebut dilepas
        '
        SQLTrans.Dispose()
        CMDUpdate.Dispose()
        CN.Close()
        CN = Nothing
    End Try
    End If
End Sub

Private Sub btnHapus_Click(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles btnHapus.Click
    '
    'menciptakan connection string dan object koneksi ke database
    '
    Dim myConnectionString = "Database=dbpenjualanbuku;Data
Source=localhost;User Id=root;Password="
    CN = New MySqlConnection(myConnectionString)
    CN.Open()

    '
    'Deklarasi object transaction dan command
    '
    Dim SQLTrans As MySqlTransaction = CN.BeginTransaction
    Dim CMDDelete As MySqlCommand = CN.CreateCommand

    If txtNota.Text <> "" Then

```

```

If MessageBox.Show("Anda yakin data penjualan dengan no
nota: " & txtNota.Text & " akan dihapus?", "Peringatan!",
MessageBoxButtons.YesNo, MessageBoxIcon.Warning) =
Windows.Forms.DialogResult.Yes Then
    Try
        With CMDDelete
            Dim Detil As String = ""
            Dim Baris, TotBaris As Integer
            TotBaris = DataGridView1.RowCount - 2
            Detil = ""
            'menggabungkan nilai pada kolom ke-0 (kode
buku), ke-5(harga)
            'dan ke-6(jumlah) yang dibatasi oleh tanda | .
            'gabungan string ini akan diuraikan pada
            'pada kolomnya masing-masing di table
penjualan_detil_tb
            For Baris = 0 To TotBaris
                Detil &= DataGridView1.Item(0,
Baris).Value.ToString & Chr(21) & DataGridView1.Item(5,
Baris).Value.ToString & Chr(21) & DataGridView1.Item(6,
Baris).Value.ToString & Chr(21)
            Next

            'pemanggilan PENJUALAN_SP sekaligus
memasukkan nilai-nilai parameternya
            Dim SQLD As String = "CALL DEMO_PENJUALAN_SP
('DELETE', @no_nota, @tgl_jual, @detil); "
            .Parameters.Add("@no_nota",
MySqlDbType.VarChar, 10).Value = txtNota.Text
            .Parameters.Add("@tgl_jual",
MySqlDbType.Date).Value = dtTanggal.Value.ToString("yyyy/MM/dd",
DateTimeFormatInfo.InvariantInfo)
            .Parameters.Add("@detil",
MySqlDbType.Text).Value = Detil
            .CommandText = SQLD
            .Transaction = SQLTrans
            .ExecuteNonQuery()
        End With

        -----
        ' Jika sukses dilaksanakan, maka perubahan dibuat
permanen
        'melalui perintah COMMIT transaction
        -----
        SQLTrans.Commit()
        ClearForm()
    Catch ex As MySqlException
        -----
        ' Jika terjadi kegagalan, maka eksekusi dibatalkan

```

```

        'melalui perintah ROLLBACK transaction
        '-----
        -

            SQLTrans.Rollback()
            MessageBox.Show(ex.Message, "Terjadi
Kegagalan!", MessageBoxButtons.OK, MessageBoxIcon.Error)
        Finally
            SQLTrans.Dispose()
            CMDDelete.Dispose()
            CN.Close()
            CN = Nothing
        End Try
    End If
Else
    MessageBox.Show("No Nota belum terisi!")
End If
End Sub

Private Sub TampilDetil()
    '-----
    'menciptakan connection string dan object koneksi ke database
    '-----
    Dim myConnectionString = "Database=dbpenjualanbuku;Data
Source=localhost;User Id=root;Password="
    CN = New MySqlConnection(myConnectionString)
    CN.Open()

    '-----
    'Deklarasi object transaction dan command
    '-----
    Dim CMDTampil As MySqlCommand = CN.CreateCommand

    If txtNota.Text <> "" Then
        Try
            CMDTampil.CommandText = "select * from tampil_jual_vw
where nota_jual = @nota_jual"
            CMDTampil.Parameters.Add("@nota_jual",
MySqlDbType.VarChar, 10).Value = txtNota.Text

            Dim daTampil As MySqlDataAdapter = New
MySqlDataAdapter(CMDTampil)

            ds.Clear()
            daTampil.Fill(ds, "tampil_jual_vw")
            Dim dtTampil As DataTable =
ds.Tables("tampil_jual_vw")

            If dtTampil.Rows.Count > 0 Then
                dtTanggal.Value = dtTampil.Rows(0).ItemArray(9)
                btnSimpan.Enabled = False
                btnUbah.Enabled = True
        End If
    End Try
End If

```

```

        btnHapus.Enabled = True
    Else
        btnSimpan.Enabled = True
        btnUbah.Enabled = False
        btnHapus.Enabled = False
    End If

    With DataGridView1
        .DataSource = ds
        .DataMember = "tampil_jual_vw"
        'bagian ini mengubah atribut datagridview
        'Lebar kolom Judul Buku
        .Columns("Judul Buku").Width = 240
        'kolom judul buku hanya dapat dibaca
        .Columns("Judul Buku").ReadOnly = True
        'Kolom kode sampai judul buku dihentikan saat
scroll
        .Columns("Judul Buku").Frozen = True
        'kolom penerbit hanya dapat dibaca
        .Columns("Penerbit").ReadOnly = True
        .Columns("Penerbit").Width = 120
        'kolom penulis hanya dapat dibaca
        .Columns("Penulis").ReadOnly = True
        .Columns("Penulis").Width = 120
        'kolom kelompok hanya dapat dibaca
        .Columns("Kelompok").ReadOnly = True
        .Columns("Kelompok").Width = 120
        'Format Angka
        .Columns("Harga").DefaultCellStyle.Format = "N"
        .Columns("Harga").DefaultCellStyle.Alignment =
DataGridViewContentAlignment.MiddleRight
        .Columns("Harga").Width = 100
        .Columns("Jumlah").DefaultCellStyle.Format = "N"
        .Columns("Jumlah").DefaultCellStyle.Alignment =
DataGridViewContentAlignment.MiddleRight
        .Columns("Jumlah").Width = 70
        .Columns("Sub Total").DefaultCellStyle.Format =
"N"
        .Columns("Sub Total").DefaultCellStyle.Alignment =
DataGridViewContentAlignment.MiddleRight
        .Columns("Sub Total").DefaultCellStyle.ForeColor =
Color.Black
        .Columns("Sub Total").DefaultCellStyle.BackColor =
Color.Yellow
        'kolom sub total hanya dapat dibaca
        .Columns("Sub Total").ReadOnly = True
        'Menyembunyikan kolom nota jual
        .Columns("nota_jual").Visible = False
        'Menyembunyikan kolom tanggal jual
        .Columns("tgl_jual").Visible = False
        Dim font As New
Font(.DefaultCellStyle.Font.FontFamily, 9, FontStyle.Bold)
Try

```

```

        'mengubah atribut font kolom ke-7 (sub total)
        .Columns("Sub Total").DefaultCellStyle.Font =
font
            .ColumnHeadersDefaultCellStyle.Font = font
        Finally
            font.Dispose()
        End Try
    End With
    '-----
    'menampilkan penjualan total
    '-----
    HitungTotal()
Catch ex As Exception
    MessageBox.Show(ex.Message, "Terjadi Kegagalan!", MessageBoxButtons.OK, MessageBoxIcon.Error)
Finally
    CMDTampil.Dispose()
    CN.Close()
    CN = Nothing
End Try
End If
End Sub

Private Sub txtNota_Leave(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles txtNota.Leave
    TampilDetil()
End Sub

Private Sub HitungTotal()
    '-----
    'Menghitung nilai total, yang diperoleh dari sub total (kolom ke-7).
    'Kolom sub total diperoleh dari perkalian kolom harga dan jumlah.
    '-----
    Dim i As Integer
    Dim total As Double
    total = 0
    For i = 0 To DataGridView1.RowCount - 1
        total += DataGridView1.Item(7, i).Value
    Next
    '-----
    'melimpahkan nilai total ke label lblTotal,
    'sekaligus mem-format-nya ke tipe mata uang (Currency)
    '-----
    lblTotal.Text = "TOTAL : " & Format(total, "C")
End Sub

Private Sub DataGridView1_CellEndEdit(ByVal sender As System.Object, ByVal e As System.Windows.Forms.DataGridViewCellEventArgs) Handles

```

```

DataGridView1.CellEndEdit
'-----
-
' menciptakan connection string dan object koneksi ke database
'-----
-
Dim myConnectionString = "Database=dbpenjualanbuku;Data
Source=localhost;User Id=root;Password="
CN = New MySqlConnection(myConnectionString)
CN.Open()

Try
    Select Case DataGridView1.CurrentCell.ColumnIndex
'pilihan cell yang sedang di-edit
        Case 0
            Dim CMDItemBuku As MySqlCommand =
CN.CreateCommand
            Dim Sql = "SELECT * FROM TAMPIL_ITEM_BUKU_VW
WHERE kode_buku = @KodeBuku;"
            CMDItemBuku.Parameters.Add("@KodeBuku",
MySqlDbType.VarChar, 13).Value = DataGridView1.CurrentCell.Value
            CMDItemBuku.CommandText = Sql

            'menampilkan semua data dimana kode buku = cell
kolom ke-0
            'dari baris yang terpilih saat ini
            Dim dr As MySqlDataReader =
CMDItemBuku.ExecuteReader
            If dr.Read Then
                With DataGridView1.CurrentRow
                    .Cells(1).Value = dr.GetString("judul")
                    .Cells(2).Value =
dr.GetString("nama_penerbit")
                    .Cells(3).Value =
dr.GetString("nama_penulis")
                    .Cells(4).Value =
dr.GetString("nama_kelompok")
                    .Cells(5).Value =
dr.GetDouble("harga_jual")
                End With
            End If
            dr.Close()
        Case 5
            With DataGridView1.CurrentRow
                'jika harga jual atau jumlah masih kosong,
                'maka kolom harga jual dan jumlah diisi nilai
0
                .Cells(5).Value = 0
                If IsDBNull(.Cells(5).Value) Then
                    If IsDBNull(.Cells(6).Value) Then
                        .Cells(6).Value = 0
                        'mencari nilai sub total,

```

```

        'perkalian antara harga jual dan jumlah
        .Cells(7).Value = .Cells(5).Value *
.Cells(6).Value
        End With
        HitungTotal()
Case 6
    With DataGridView1.CurrentRow
        'jika harga jual atau jumlah masih kosong,
        'maka kolom harga jual dan jumlah diisi nilai
0
        If IsDBNull(.Cells(5).Value) Then
            If IsDBNull(.Cells(6).Value) Then
                .Cells(6).Value = 0
                'mencari nilai sub total,
                'perkalian antara harga jual dan jumlah
                .Cells(7).Value = .Cells(5).Value *
.Cells(6).Value
            End With
            HitungTotal()
        End Select
    Catch ex As Exception
        '-----
        'Pesan kesalahan ditampilkan melalui MessageBox.Show
        '-----
        MessageBox.Show(ex.Message, "Terjadi Kegagalan!",
MessageBoxButtons.OK, MessageBoxIcon.Error)
    Finally
        CN.Close()
        CN = Nothing
    End Try
End Sub
End Class

```

7. Jalankan programnya dengan menekan tombol **F5** atau **Ctrl+F5**. Masukkan beberapa data kemudian simpan, selanjutnya tampilkan datanya dengan memasukkan nomor nota yang sudah disimpan di kotak teks **txtNota**, perhatikan hasilnya kemudian ubah atau hapus datanya.

Table
penjualan_header_tb

Table
penjualan_detil_tb

Jika Anda hendak menghapus data maka akan keluar window pesan untuk memastikan bahwa nota yang akan dihapus benar.



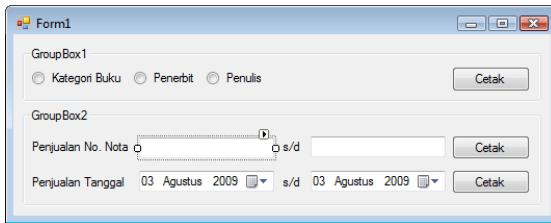
5.8 Form Laporan (Project Laporan)

Laporan dalam pemrograman database menduduki posisi terpenting karena merupakan tujuan akhir dari suatu aplikasi database. Laporan berisi informasi-informasi penting yang diperoleh dari rangkaian beberapa table atau view disusun sedemikian rupa sehingga menjadi suatu Query yang siap disajikan dalam bentuk laporan.

Laporan akan menjadi kompleks jika disertai dengan agregasi misalnya total, sub total, dan grand total. Memungkinkan pula laporan disajikan

dalam kelompok kolom tertentu dari suatu table, hal ini sering disebut dengan *grouping*.

1. Buka menu **File->New->Project**, Pilih Template **Windows Application**, beri nama project dengan **Laporan**.
2. Buka menu **Project->Add Reference** pilih TAB **.NET->MySQL.Data**.
3. Buka menu **Project->Laporan Properties->References** kemudian beri tanda centang pada beberapa item *namespace MySQL.Data* antara lain: **MySql**, **MySql.Data**, **MySql.Data.MySqlClient**, **MySql.Data.MySqlClient.Properties**, dan **MySql.Data.Types**.
4. Ganti nama **form1.vb** menjadi **frmLaporan.vb**. Rancang desain form beserta kelengkapan kontrolnya seperti berikut ini.



Tuliskan kode program **frmLaporan.vb** di bawah ini:

frmLaporan.vb

```
Imports System.Globalization

Public Class frmLaporan
    Private MyDataSet As DataSet = New DataSet()
    Private MyAdapter As New MySqlDataAdapter()
    Private CMD As New MySqlCommand
    Private CN As New MySqlConnection
    Private SQL As String

    Private Sub frmLaporan_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load
        Me.Text = "Laporan"
        GroupBox1.Text = "Laporan Buku"
        GroupBox2.Text = "Laporan Penjualan"
    End Sub

    Private Sub btnLaporanBuku_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnLaporanBuku.Click
        Dim myConnectionString = "Database=dbpenjualanbuku;Data Source=localhost;User Id=root;Password="
    End Sub
```

```

CN = New MySqlConnection(myConnectionString)
CN.Open()

If rdJenis.Checked = True Then
    MyDataSet.Clear()
    SQL = "SELECT * FROM RPT_BUKU_VW;"

    Try
        CMD = New MySqlCommand(SQL, CN)
        MyAdapter.SelectCommand = CMD
        MyAdapter.Fill(MyDataSet, "RPT_BUKU_VW")

        Dim MyReport As New rptJenisBuku
        MyReport.SetDataSource(MyDataSet)
        'menghidupkan displaygroupdirectory
        ViewLaporan.CrystalReportViewer1.DisplayGroupTree =
True
        ViewLaporan.CrystalReportViewer1.ReportSource =
MyReport
        Catch ex As Exception
            MessageBox.Show(ex.Message, "Terjadi Kegagalan!",
MessageBoxButtons.OK, MessageBoxIcon.Error)
        Finally
            CN.Close()
            CN = Nothing
        End Try
    End If

    If rdPenerbit.Checked = True Then
        MyDataSet.Clear()
        SQL = "SELECT * FROM RPT_BUKU_VW;"

        Try
            CMD = New MySqlCommand(SQL, CN)
            MyAdapter.SelectCommand = CMD
            MyAdapter.Fill(MyDataSet, "RPT_BUKU_VW")

            Dim MyReport As New rptPenerbitBuku
            MyReport.SetDataSource(MyDataSet)
            'menghidupkan displaygroupdirectory
            ViewLaporan.CrystalReportViewer1.DisplayGroupTree =
True
            ViewLaporan.CrystalReportViewer1.ReportSource =
MyReport
            Catch ex As Exception
                MessageBox.Show(ex.Message, "Terjadi Kegagalan!",
MessageBoxButtons.OK, MessageBoxIcon.Error)
            Finally
                CN.Close()
                CN = Nothing
            End Try
        End If

        If rdPenulis.Checked = True Then

```

```

        MyDataSet.Clear()
        SQL = "SELECT * FROM RPT_BUKU_VW;"
        Try
            CMD = New MySqlCommand(SQL, CN)
            MyAdapter.SelectCommand = CMD
            MyAdapter.Fill(MyDataSet, "RPT_BUKU_VW")

            Dim MyReport As New rptPenulisBuku
            MyReport.SetDataSource(MyDataSet)
            'menghidupkan displaygroupdirectory
            ViewLaporan.CrystalReportViewer1.DisplayGroupTree =
True
            ViewLaporan.CrystalReportViewer1.ReportSource =
MyReport
            Catch ex As Exception
                MessageBox.Show(ex.Message, "Terjadi Kegagalan!", MessageBoxButtons.OK, MessageBoxIcon.Error)
            Finally
                CN.Close()
                CN = Nothing
            End Try
        End If
    End Sub

    Private Sub btnLaporanNota_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles btnLaporanNota.Click
        Dim myConnectionString = "Database=dbpenjualanbuku;Data
Source=localhost;User Id=root;Password="
        CN = New MySqlConnection(myConnectionString)
        CN.Open()
        Try
            MyDataSet.Clear()
            SQL = "SELECT * FROM RPT_PENJUALAN_VW "
            SQL &= "WHERE nota_jual BETWEEN @Nota1 AND @Nota2"
            CMD = New MySqlCommand(SQL, CN)
            CMD.Parameters.Add("@Nota1", MySqlDbType.VarChar,
10).Value = txtNota1.Text
            CMD.Parameters.Add("@Nota2", MySqlDbType.VarChar,
10).Value = txtNota2.Text
            MyAdapter.SelectCommand = CMD
            MyAdapter.Fill(MyDataSet, "RPT_PENJUALAN_VW")

            Dim MyReport As New rptPenjualanPerNota
            MyReport.SetDataSource(MyDataSet)
            ViewLaporan.CrystalReportViewer1.ReportSource = MyReport
        Catch ex As Exception
            MessageBox.Show(ex.Message, "Terjadi Kegagalan!", MessageBoxButtons.OK, MessageBoxIcon.Error)
        Finally
            CN.Close()
            CN = Nothing
        End Try
    End Sub

```

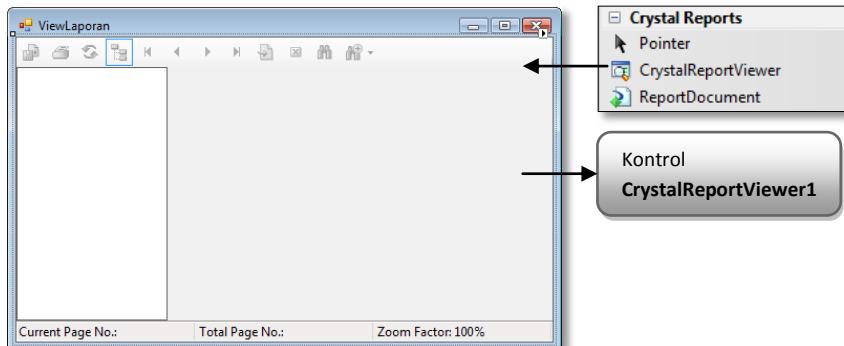
```

    Private Sub btnLaporanTanggal_Click(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
btnLaporanTanggal.Click
    Dim myConnectionString = "Database=dbpenjualanbuku;Data
Source=localhost;User Id=root;Password="
    CN = New MySqlConnection(myConnectionString)
    CN.Open()
    Try
        MyDataSet.Clear()
        SQL = "SELECT * FROM RPT_PENJUALAN_PERTGL_VW "
        SQL &= "WHERE tgl_jual BETWEEN @tgl1 AND @tgl2;"
        CMD = New MySqlCommand(SQL, CN)
        CMD.Parameters.Add("@tgl1", MySqlDbType.Date).Value =
dtJual1.Value.ToString("yyyy/MM/dd",
DateTimeFormatInfo.InvariantInfo)
        CMD.Parameters.Add("@tgl2", MySqlDbType.Date).Value =
dtJual2.Value.ToString("yyyy/MM/dd",
DateTimeFormatInfo.InvariantInfo)
        MyAdapter.SelectCommand = CMD
        MyAdapter.Fill(MyDataSet, "RPT_PENJUALAN_PERTGL_VW")

        Dim MyReport As New rptPenjualanPerTgl
        MyReport.SetDataSource(MyDataSet)
        ViewLaporan.CrystalReportViewer1.ReportSource = MyReport
    Catch ex As Exception
        MessageBox.Show(ex.Message, "Terjadi Kegagalan!",
MessageBoxButtons.OK, MessageBoxIcon.Error)
    Finally
        CN.Close()
        CN = Nothing
    End Try
End Sub
End Class

```

- 5.Tambahkan satu form baru sebagai tempat laporan. Buka menu
Project-> Add Windows Form, beri nama form dengan
ViewLaporan.vb. Tambahkan kontrol **CrystalReportViewer** dari
ToolBox ke dalam form **ViewLaporan.vb**.



6. Buat beberapa view yang akan digunakan sebagai table maya untuk ditampilkan dalam laporan. Buka program navicat melalui menu **Tools->Console** atau menu **Query->New Query**. Tuliskan script query di bawah ini.

```

DROP VIEW IF EXISTS RPT_BUKU_VW;
CREATE VIEW RPT_BUKU_VW
AS
SELECT buku_tb.kode_buku, buku_tb.judul,
kelompok_buku_tb.nama_kelompok, penerbit_tb.nama_penerbit,
penulis_tb.nama_penulis, buku_tb.tahun_terbit, buku_tb.foto
FROM penulis_tb INNER JOIN (penerbit_tb INNER JOIN
(kelompok_buku_tb INNER JOIN buku_tb ON
kelompok_buku_tb.kode_kelompok = buku_tb.kode_kelompok) ON
penerbit_tb.kode_penerbit = buku_tb.kode_penerbit) ON
penulis_tb.kode_penulis = buku_tb.kode_penulis;

DROP VIEW IF EXISTS RPT_PENJUALAN_PERTGL_VW;
CREATE VIEW RPT_PENJUALAN_PERTGL_VW
AS
SELECT penjualan_header_tb.tgl_jual,
penjualan_header_tb.not_a_jual,
Sum(penjualan_detil_tb.Jumlah*penjualan_detil_tb.harga) AS
SubTotal
FROM penjualan_header_tb INNER JOIN penjualan_detil_tb ON
penjualan_header_tb.not_a_jual = penjualan_detil_tb.not_a_jual
GROUP BY penjualan_header_tb.tgl_jual,
penjualan_header_tb.not_a_jual;

DROP VIEW IF EXISTS RPT_PENJUALAN_VW;
CREATE VIEW RPT_PENJUALAN_VW
AS

```

```

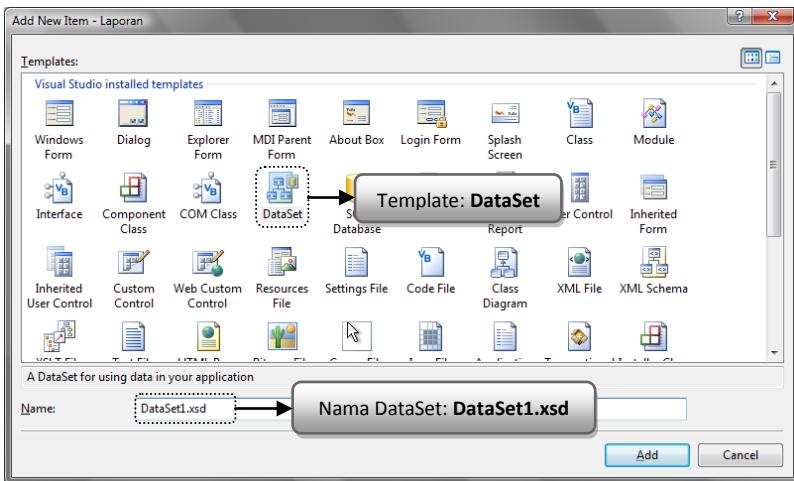
SELECT penjualan_header_tb.nota_jual,
penjualan_header_tb.tgl_jual, penjualan_detil_tb.kode_buku,
buku_tb.judul, penjualan_detil_tb.jumlah,
penjualan_detil_tb.harga,
penjualan_detil_tb.jumlah*penjualan_detil_tb.harga AS SubTotal
FROM penjualan_header_tb INNER JOIN (buku_tb INNER JOIN
penjualan_detil_tb ON buku_tb.kode_buku =
penjualan_detil_tb.kode_buku) ON penjualan_header_tb.nota_jual =
penjualan_detil_tb.nota_jual;

DROP VIEW IF EXISTS RPT_PENJUALANPERBUKU_VW;
CREATE VIEW RPT_PENJUALANPERBUKU_VW
AS
SELECT penjualan_detil_tb.kode_buku, buku_tb.judul,
penjualan_header_tb.nota_jual, penjualan_header_tb.tgl_jual,
penjualan_detil_tb.jumlah, penjualan_detil_tb.harga,
penjualan_detil_tb.jumlah*penjualan_detil_tb.harga AS subtotal
FROM buku_tb INNER JOIN (penjualan_header_tb INNER JOIN
penjualan_detil_tb ON penjualan_header_tb.nota_jual =
penjualan_detil_tb.nota_jual) ON buku_tb.kode_buku =
penjualan_detil_tb.kode_buku
ORDER BY penjualan_detil_tb.kode_buku,
penjualan_header_tb.nota_jual;

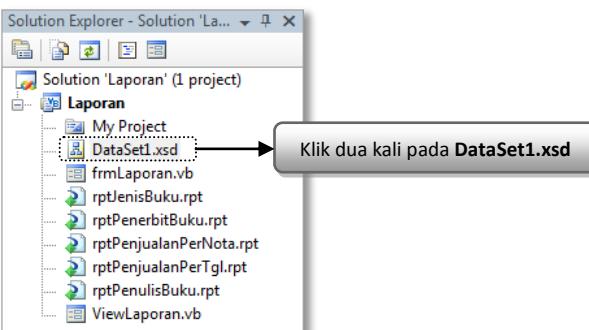
DROP VIEW IF EXISTS RPT_PENJUALANPERJENISBUKU_VW;
CREATE VIEW RPT_PENJUALANPERJENISBUKU_VW
AS
SELECT kelompok_buku_tb.kode_kelompok,
kelompok_buku_tb.nama_kelompok, buku_tb.kode_buku, buku_tb.judul,
penjualan_header_tb.nota_jual, penjualan_header_tb.tgl_jual,
penjualan_detil_tb.jumlah, penjualan_detil_tb.harga,
penjualan_detil_tb.jumlah*penjualan_detil_tb.harga AS subtotal
FROM penjualan_header_tb INNER JOIN ((kelompok_buku_tb INNER JOIN
buku_tb ON kelompok_buku_tb.kode_kelompok = buku_tb.kode_kelompok)
INNER JOIN penjualan_detil_tb ON buku_tb.kode_buku =
penjualan_detil_tb.kode_buku) ON penjualan_header_tb.nota_jual =
penjualan_detil_tb.nota_jual
ORDER BY kelompok_buku_tb.nama_kelompok,
penjualan_header_tb.nota_jual;

```

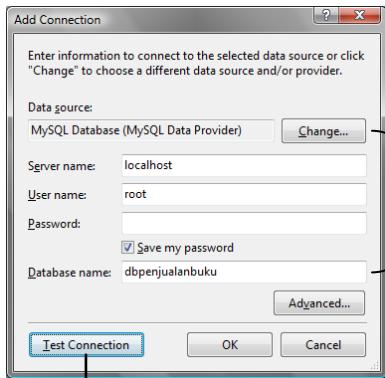
7. Tambahkan kontrol **DataSet**, buka menu **Project->Add Windows Form**, pilih template **Dataset** dan beri nama dengan **DataSet1.xsd**.



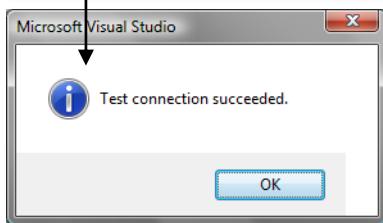
8. Klik dua kali **DataSet1.xsd** pada window Solution Explorer



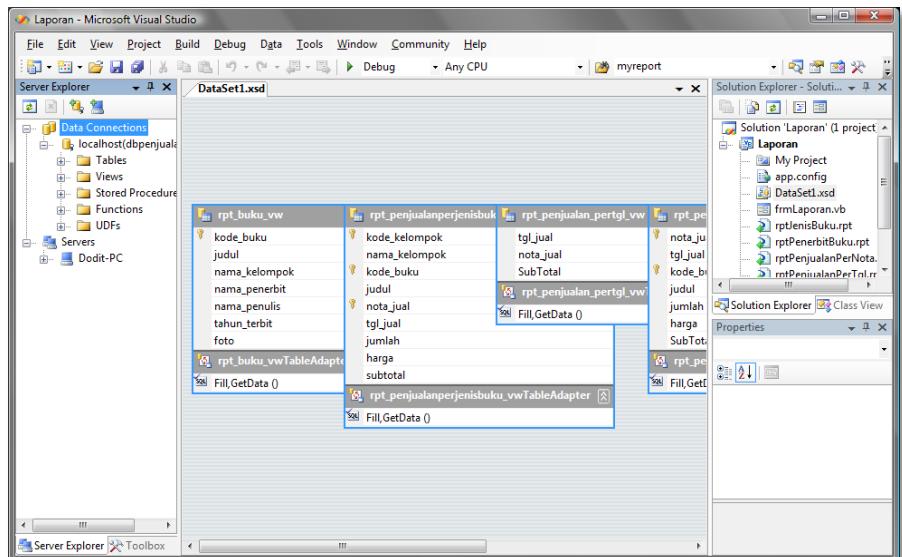
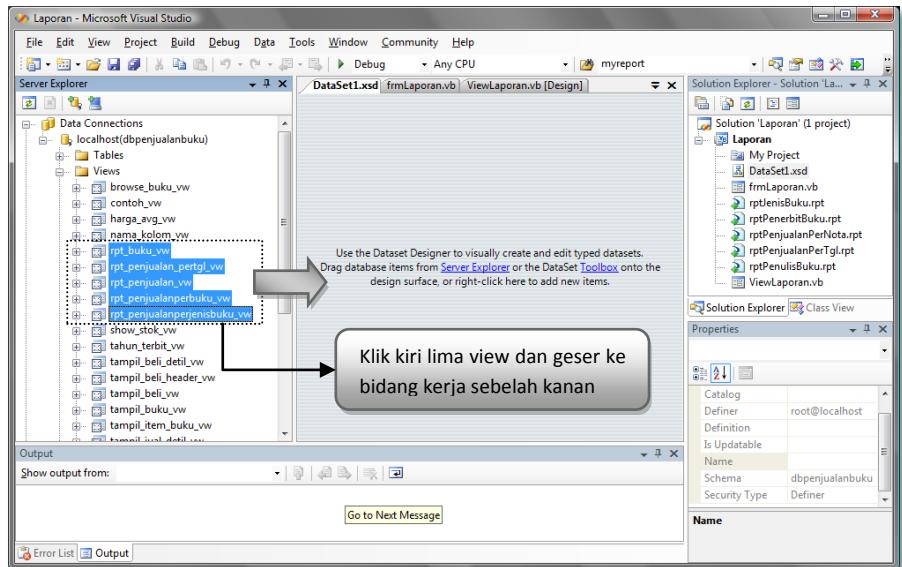
Buka menu **Tools->Connect To Database**, lakukan konfigurasi seperti di bawah ini. Setelah itu lakukan pengujian dengan menekan tombol **Test Connection**. Jika sukses lanjutkan dengan menekan tombol **OK**.



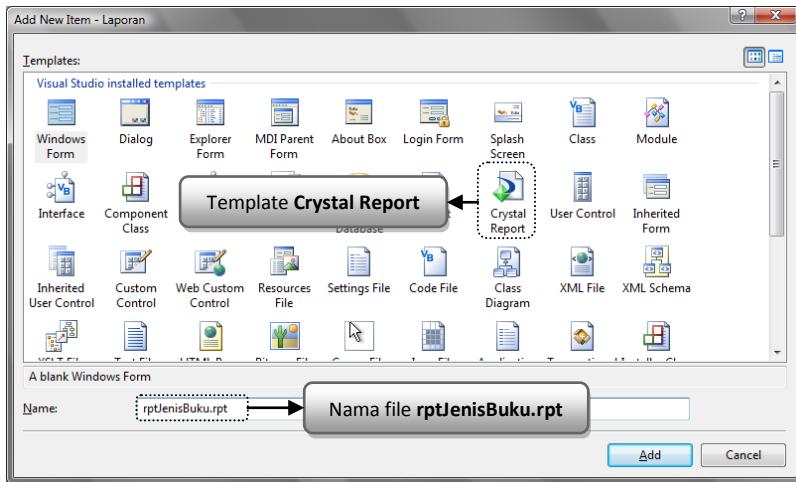
Data source=MySQL Database,
Server name=localhost, User
name=root, Password=<kosong>,
Centang Save my password,
Database name=dbpenjualanbuku



Lakukan drag-drop lima view antara lain **rpt_buku_vw**, **rpt_penjualanperjenisbuku_vw**, **rpt_penjualan_perbuku_vw**, **rpt_penjualan_vw** dari window **Server Explorer** root **Views** ke bidang kerja sebelah kanan.



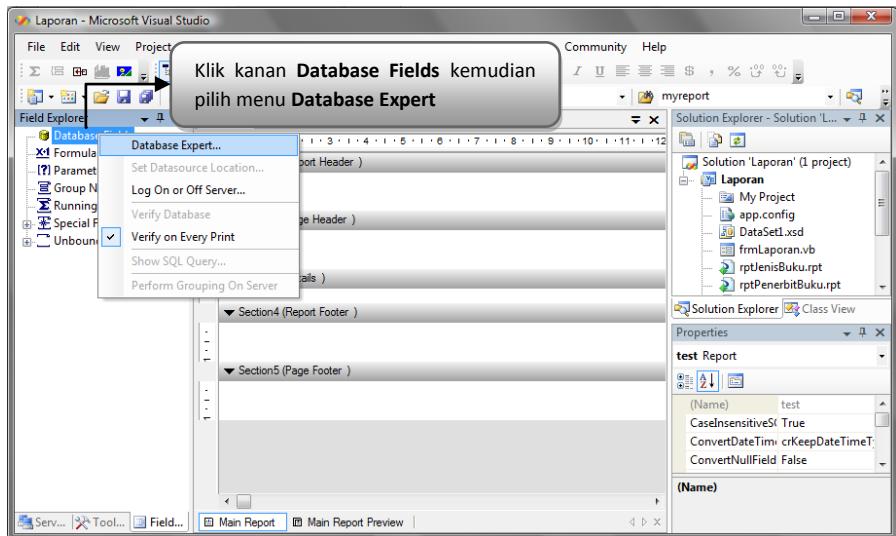
9. Buka menu Project->Add Windows Form, pilih template **CrystalReport** dan beri nama **rptJenisBuku.rpt**. Selanjutnya klik tombol **Add**.



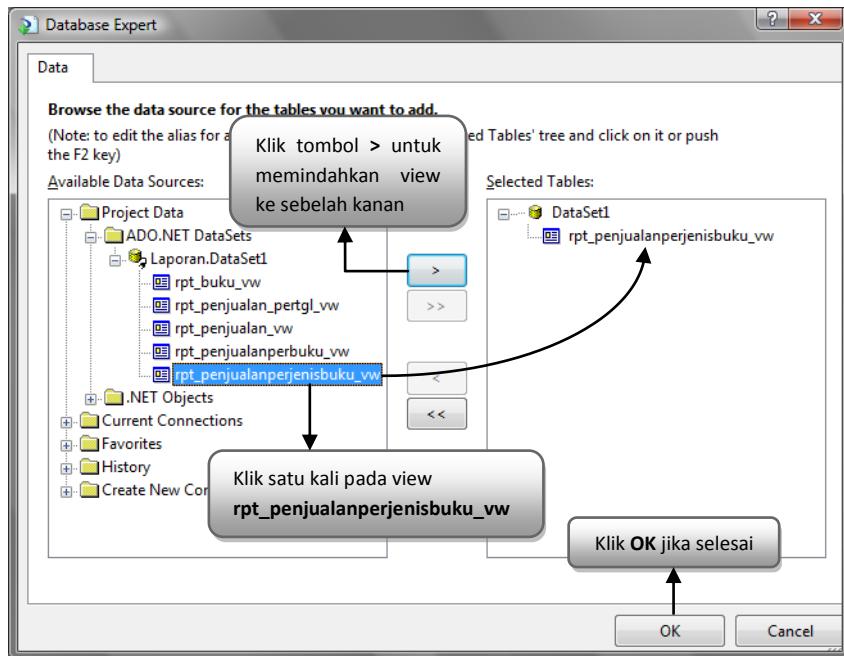
Kita akan mendesain laporan secara manual sehingga harus dipilih **As Blank Report** kemudian klik tombol **OK**.



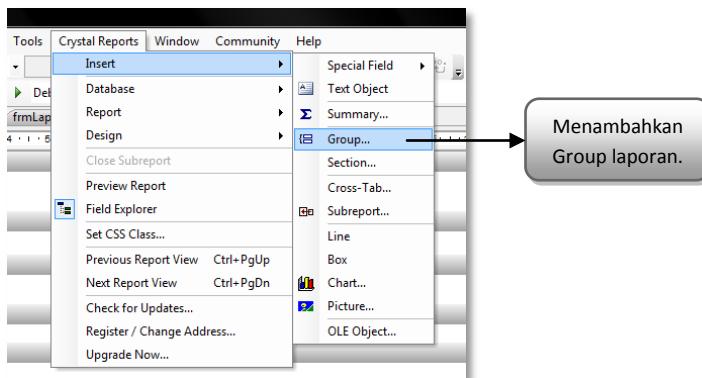
Klik kanan pada **Database Fields** pilih menu **Database Expert**

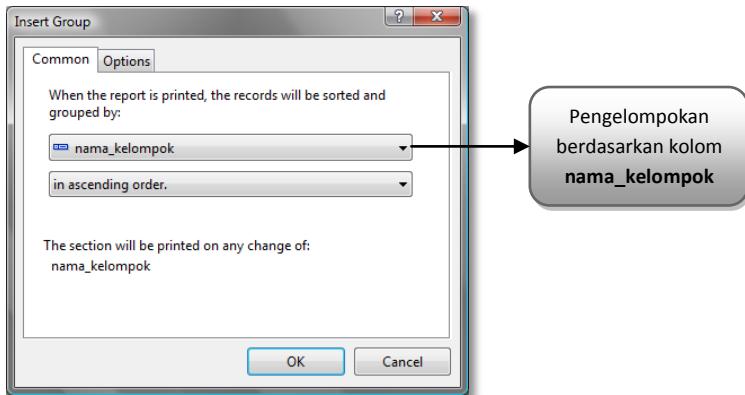


Klik satu kali view **rpt_penjualanperjenisbuku_vw** yang berada pada area **Available Data Sources** sebelah kiri, kemudian klik tombol > untuk memindahkan view tersebut ke area **Selected Tables** sebelah kanan. Selanjutnya klik **OK**.



10. Buka menu **Crystal Reports->Insert->Group** untuk menambahkan *Group* laporan.



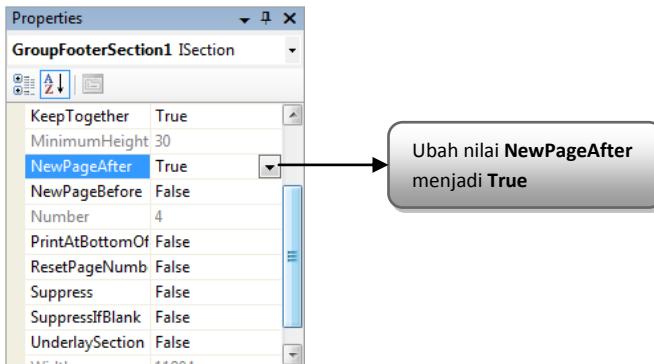


11. Lakukan *drag-drop* terhadap kolom **kode_buku**, **judul**, **nama_kelompok**, **nama_penulis**, **tahun_terbit**, dan **foto** dari view **rpt_buku_vw** ke bidang **Section3 (Details)**. Tambahkan **Text Object** (Crystal Reports->Insert->Text Object) pada **Section1 (Report Header)** sebagai judul laporan. Tambahkan **Line** (Menu Crystal Reports->Insert->Line) pada **Section2 (Page Header)** dan **GroupFooterSection1**. Ganti judul kolom pada **Section2 (Page Header)** dengan cara klik kanan pada labelnya masing-masing kemudian pilih **Edit Object**. Atur sedemikian rupa sehingga tampilannya seperti berikut ini.

Daftar Buku Berdasarkan Jenis

Kode	Judul Buku	Penerbit	Penulis	Tahun	Foto	
Group #1 Name						
Section3 (Details)	kode_buku	judul	nama_penerbit	nama_penulis	in_terbit	foto
GroupFooterSection1 (Group Footer #1: rpt_buku_vw.nama_kelompok - A)						
Section4 (Report Footer)						
Section5 (Page Footer)						
Halaman						
					Print Date	Print Time

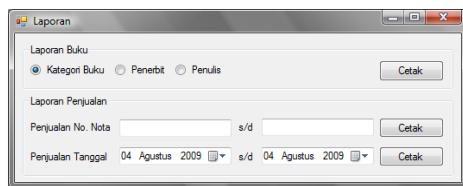
Tambahkan **Page Number** dari **Fields Explorer->Special Fields->Page Number**, tambahkan **Print Date** dari **Fields Explorer->Special Fields->Print Date**, tambahkan pula **Print Time** dari **Fields Explorer->Special Fields->Print Time**. Letakkan semuanya di **Section5 (Page Footer)**.



Klik satu Kali pada GroupFooterSection1 kemudian ubah nilai dari **Properties->NewPageAfter** menjadi **True**.

12. Cara-cara pembuatan form laporan di atas berlaku bagi semua laporan yang belum dibuat. Anda dapat mencoba laporan yang lain berdasarkan CD program contoh yang disertakan dalam buku.

13. Jalankan programnya dengan menekan tombol F5 atau Ctrl+F5.



Contoh laporan buku berdasarkan **Kategori Buku**:

Kode	Judul Buku	Penerbit	Penulis	Tahun	Foto
9789793767079	Efek Partikel 3ds max 6	Maxikom CV.	Handi Chandra	2004	
9789793767543	7 Jam Belajar Visual Basic .NET Untuk Orang Awam	Maxikom CV.	Firdaus	2006	
9789793767666	7 Jam Belajar Interaktif Autocad Untuk Orang Awam	Maxikom CV.	Handi Chandra	2006	
9789799992260	30 Menit Menjadi Webmaster	Oase Media	Haris Supriyansyah	2007	

Contoh Laporan Buku berdasarkan **Penerbit**:

ViewLaporan

Main Report

Daftar Buku Berdasarkan Penerbit

Kode	Judul Buku	Kelompok	Penulis	Terbit	Foto
Andi Offset					
9789797633134	Kumpulan Latihan Pemrograman Delphi	Komputer	Jaja Jamaludin Malik	2006	
9789795336730	Internet Marketing	Marketing	Riyuke Ustadiyanto	2001	
Informatika Bandung					
9789793338125	Esensi-esensi Bahasa Pemrograman JAVA Disertai Lebih Dari 100 Contoh Program	Komputer	Bambang Haryanto	2005	
9789793338729	Membuat Aplikasi Web Interaktif dengan ASP	Komputer	Bernard Renaldy Suteja	2006	

Current Page No.: 1 Total Page No.: 1+ Zoom Factor: 100%

Contoh laporan buku berdasarkan Penulis:

ViewLaporan

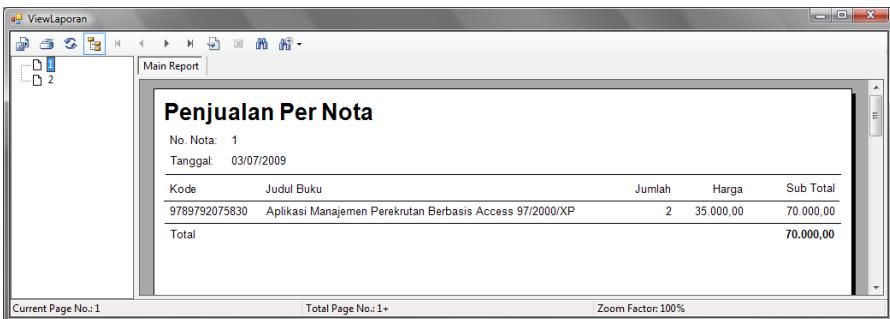
Main Report

Daftar Buku Berdasarkan Penulis

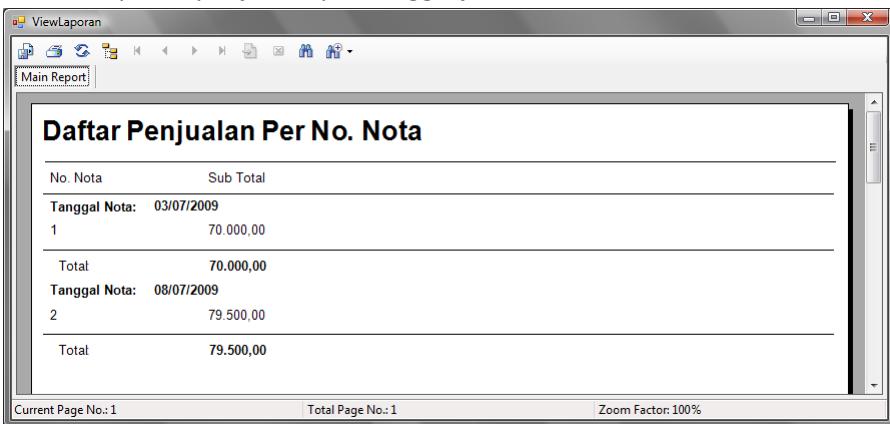
Kode	Judul Buku	Jenis Buku	Penerbit	Terbit	Foto
Bambang Hariyanto					
9789793338125	Esensi-esensi Bahasa Pemrograman JAVA Disertai Lebih Dari 100 Contoh Program	Komputer	Informatika Bandung	2005	

Current Page No.: 1 Total Page No.: 1+ Zoom Factor: 100%

Contoh laporan penjualan per nota:



Contoh Laporan penjualan per tanggal jual:



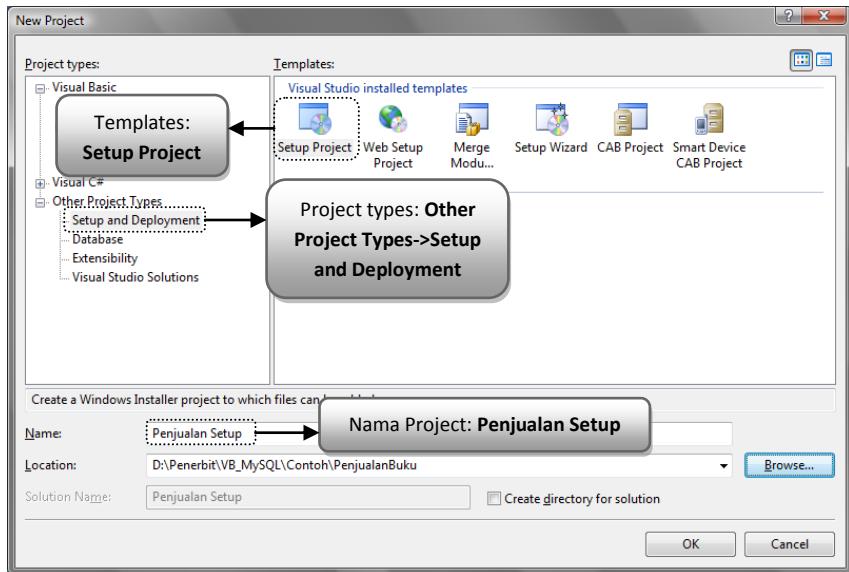
BAB 6

Setup Dan Deployment

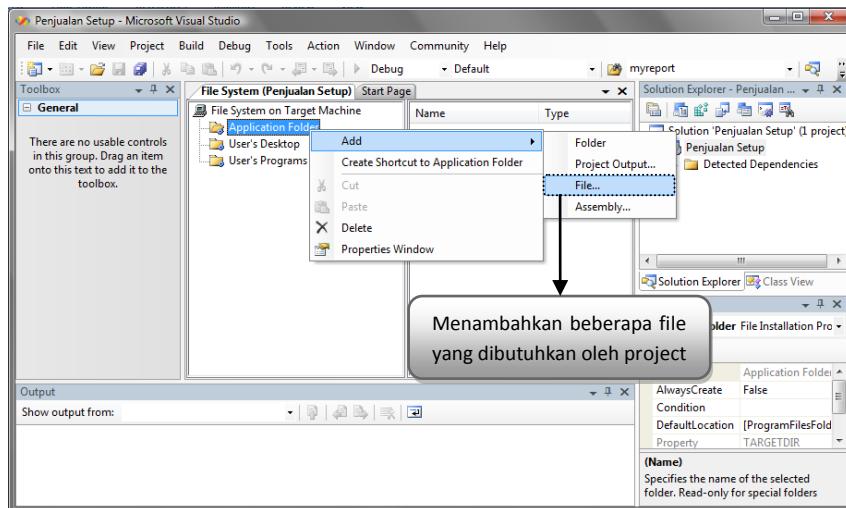
Setup dan Deployment adalah metode untuk menyatukan file-file project yang telah dibuat sebagai satu file paket yang siap untuk diinstal. Dengan setup dan deployment maka tidak dibutuhkan lagi file sumber yang berbasis teks dan ketergantungan file (*dependencies file*) dalam project karena seluruh file yang dibutuhkan sudah terkompilasi menjadi file biner (*binary file*). Dengan Setup dan Deployment maka pengguna dengan mudah menginstall layak program profesional.

Sebagai percobaan kita akan menggunakan project **PenjualanBuku** untuk dibuat setup dan deployment.

1. Buka program Microsoft Visual Studio 2005
2. Buka menu **File->New->Project**, pilih **Project Type** dengan **Other Project Types->Setup and Deployment**, pilih **Templates** dengan **Setup Project**. Beri nama project dengan **Penjualan Setup**, klik tombol **OK** untuk melanjutkan.

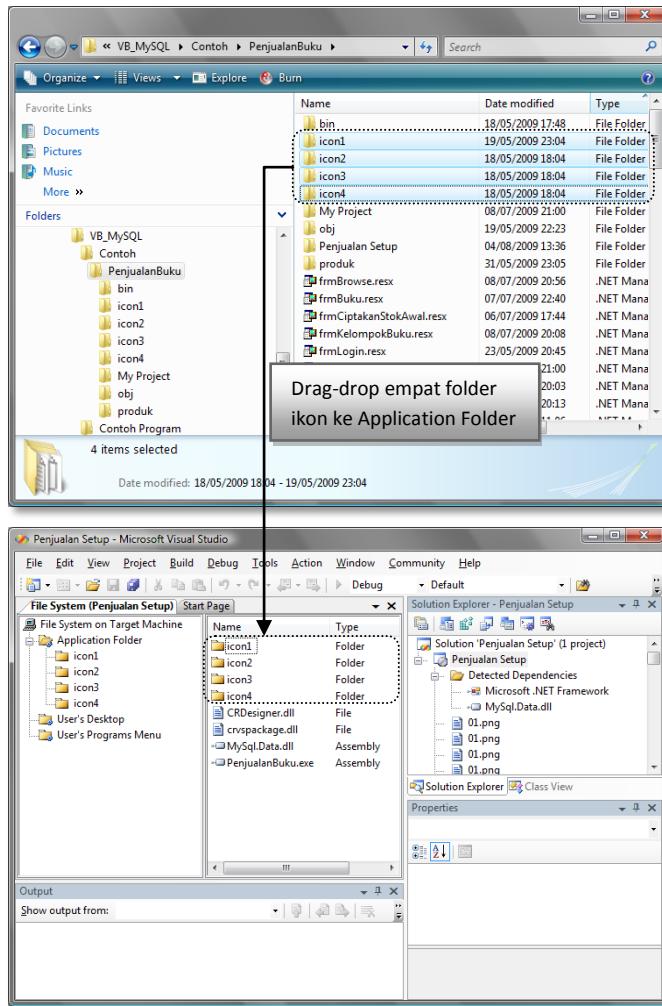


3. Pada area kerja **File System (Penjualan Setup)**, klik kanan **Application Folder** kemudian pilih menu **Add->File** untuk menambahkan file berekstensi **.EXE**. Sebagai contoh adalah **PenjualanBuku.exe** yang terletak pada folder **D:\Penerbit\VB_SQL\Contoh\PenjualanBuku\bin\Debug**.



Dengan cara yang sama tambahkan file **CRDesigner.dll** dan **crvspackage.dll** yang diperoleh dari folder **C:\Program Files\Microsoft Visual Studio 8\Crystal Reports**. Kedua file tersebut merupakan file biner yang digunakan sebagai pendukung laporan.

Tambahkan folder ikon yang dibutuhkan oleh program aplikasi dengan cara drag-drop beberapa folder ikon dari **Windows Explorer** ke area kerja **File System On Target Machine->Application Folder** seperti tampak gambar berikut.



4. Jika semua file yang dibutuhkan telah diletakkan di **Application Folder** selanjutnya adalah membentuk file Setup Deployment dengan cara memuka menu **Build->Build Penjualan Setup**.
5. Proses **Build** menghasilkan file **Penjualan Setup.msi** dan **setup.exe** yang siap didistribusikan dan diinstall pada komputer lain, seperti tampak gambar berikut ini:

