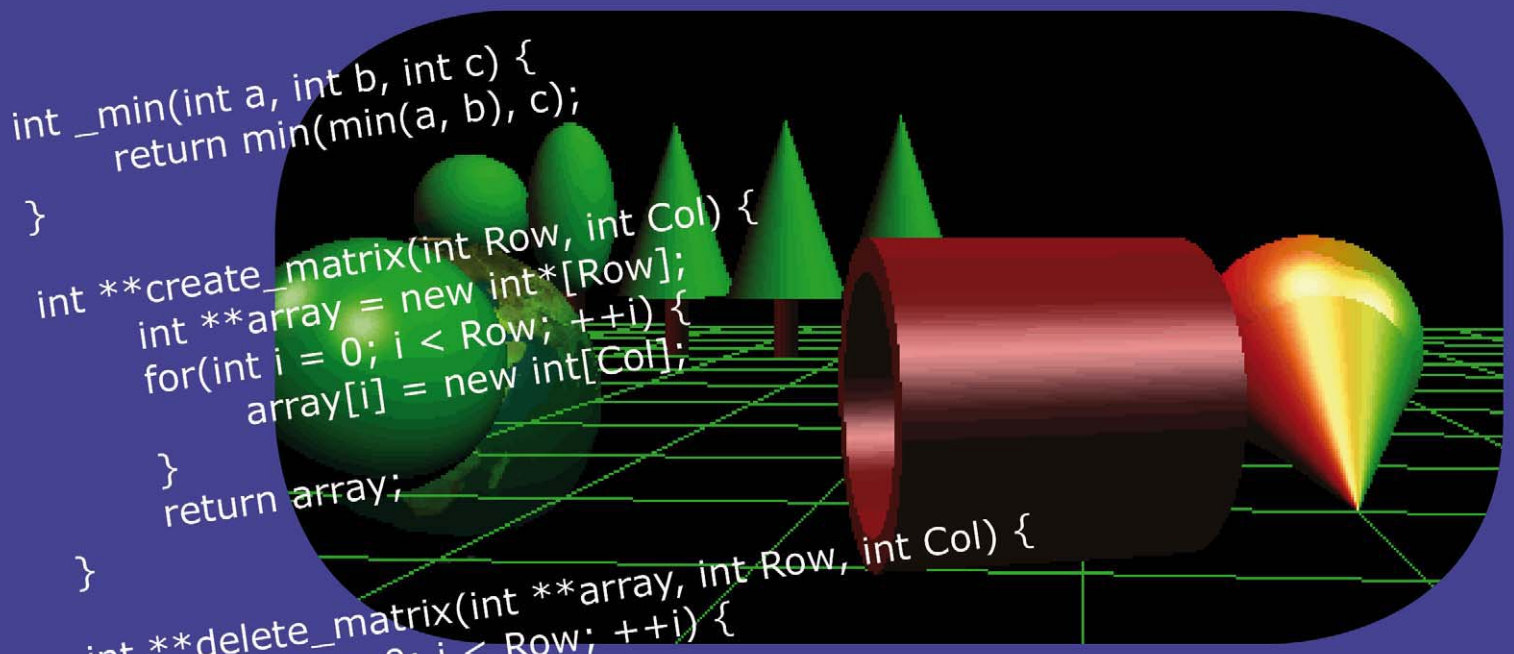
JILID 3

Aunur Rofiq Mulyanto, dkk.

REKAYASA PERANGKAT

untuk
Sekolah Menengah Kejuruan



Direktorat Pembinaan Sekolah Menengah Kejuruan
Direktorat Jenderal Manajemen Pendidikan Dasar dan Menengah
Departemen Pendidikan Nasional

Aunur R. Mulyanto

REKAYASA PERANGKAT LUNAK
JILID 3

SMK

Direktorat Pembinaan Sekolah Menengah Kejuruan
Direktorat Jenderal Manajemen Pendidikan Dasar dan Menengah
Departemen Pendidikan Nasional

REKAYASA PERANGKAT LUNAK JILID 3

Untuk SMK

Penulis : Aunur R. Mulyanto
Perancang Kulit : Tim

Ukuran Buku : 17,6 x 25 cm

MUL MULYANTO,Aunur R.
Rekayasa Perangkat Lunak Jilid 1 untuk SMK /oleh Aunur R. Mulyanto ---- Jakarta : Direktorat Pembinaan Sekolah Menengah Kejuruan, Direktorat Jenderal Manajemen Pendidikan Dasar dan Menengah, Departemen Pendidikan Nasional, 2008.
v. 159 hlm
Daftar Pustaka : A1-A2
Glosarium : B1-B6
ISBN : 978-979-060-007-2
ISBN : 978-979-060-010-2

Diterbitkan oleh
Direktorat Pembinaan Sekolah Menengah Kejuruan
Direktorat Jenderal Manajemen Pendidikan Dasar dan Menengah
Departemen Pendidikan Nasional
Tahun 2008

KATA SAMBUTAN

Puji syukur kami panjatkan kehadirat Allah SWT, berkat rahmat dan karunia Nya, Pemerintah, dalam hal ini, Direktorat Pembinaan Sekolah Menengah Kejuruan Direktorat Jenderal Manajemen Pendidikan Dasar dan Menengah Departemen Pendidikan Nasional, telah melaksanakan kegiatan penulisan buku kejuruan sebagai bentuk dari kegiatan pembelian hak cipta buku teks pelajaran kejuruan bagi siswa SMK. Karena buku-buku pelajaran kejuruan sangat sulit di dapatkan di pasaran.

Buku teks pelajaran ini telah melalui proses penilaian oleh Badan Standar Nasional Pendidikan sebagai buku teks pelajaran untuk SMK dan telah dinyatakan memenuhi syarat kelayakan untuk digunakan dalam proses pembelajaran melalui Peraturan Menteri Pendidikan Nasional Nomor 45 Tahun 2008 tanggal 15 Agustus 2008.

Kami menyampaikan penghargaan yang setinggi-tingginya kepada seluruh penulis yang telah berkenan mengalihkan hak cipta karyanya kepada Departemen Pendidikan Nasional untuk digunakan secara luas oleh para pendidik dan peserta didik SMK.

Buku teks pelajaran yang telah dialihkan hak ciptanya kepada Departemen Pendidikan Nasional ini, dapat diunduh (*download*), digandakan, dicetak, dialihmediakan, atau difotokopi oleh masyarakat. Namun untuk penggandaan yang bersifat komersial harga penjualannya harus memenuhi ketentuan yang ditetapkan oleh Pemerintah. Dengan ditayangkan *soft copy* ini diharapkan akan lebih memudahkan bagi masyarakat khsusnya para pendidik dan peserta didik SMK di seluruh Indonesia maupun sekolah Indonesia yang berada di luar negeri untuk mengakses dan memanfaatkannya sebagai sumber belajar.

Kami berharap, semua pihak dapat mendukung kebijakan ini. Kepada para peserta didik kami ucapkan selamat belajar dan semoga dapat memanfaatkan buku ini sebaik-baiknya. Kami menyadari bahwa buku ini masih perlu ditingkatkan mutunya. Oleh karena itu, saran dan kritik sangat kami harapkan.

Jakarta, 17 Agustus 2008
Direktur Pembinaan SMK

PENGANTAR PENULIS

Dengan segala kerendahan hati, kami mengucapkan syukur kepada Allah SWT. Karena hanya dengan lindungan, rahmat dan karuniaNya-lah maka buku ini dapat diselesaikan.

Buku yang berjudul 'Rekayasa Perangkat Lunak' merupakan buku yang disusun untuk memenuhi kebutuhan buku pegangan bagi siswa Sekolah Menengah Kejuruan. Khususnya pada program keahlian Rekayasa Perangkat Lunak. Buku ini memuat uraian yang mengacu pada standar kompetensi dan kompetensi dasar Rekayasa Perangkat Lunak untuk siswa SMK mulai dari kelas X, XI sampai dengan kelas XII.

Tiap bab berisi teori yang harus dipahami secara benar oleh peserta didik dan disertai dengan contoh-contoh soal yang relevan dengan teori tersebut. Selain itu terdapat juga soal-soal yang didasarkan pada konsep dan teori yang dibahas sebagai alat uji untuk mengukur kemampuan peserta didik dalam penguasaan materi tersebut.

Dalam mengembangkan buku ini, penulis berupaya agar materi yang disajikan sesuai dengan kebutuhan kompetensi yang harus dicapai. Oleh karenanya, selain dari hasil pemikiran dan pengalaman penulis sebagai pengajar dan praktisi Rekayasa Perangkat Lunak, materi yang dikembangkan juga diperkaya dengan referensi-referensi lain yang sesuai.

Pada kesempatan ini penulis ingin menyampaikan rasa terima kasih kepada semua pihak yang mendukung buku ini dapat diterbitkan. Mudah-mudahan buku ini dapat bermanfaat bagi peserta didik dalam mengembangkan kemampuannya. Penulis menyadari bahwa buku ini masih perlu dikembangkan terus menerus, sehingga saran dari berbagai pihak pengguna buku ini sangat diharapkan.

Penulis

GET IT ON Google Play

Download di App Store

iOS segera hadir

Unduh buku lainnya melalui aplikasi. Gratis.

Buku BSE dilengkapi dengan daftar isi untuk memudahkan navigasi. Tersedia juga majalah, tabloid, buku dan koran yang lebih hemat hingga 80% dibanding edisi cetak.

Unduh aplikasi myedisi reader gratis

myedisi.com/reader

myedisi

Buku BSE terbaru belum tersedia di myedisi? Sampaikan melalui email **bse@myedisi.com**

DAFTAR ISI

PETUNJUK PENGGUNAAN BUKU

A. Deskripsi Umum

Buku ini diberi judul "Rekayasa Perangkat Lunak",sama dengan salah satu program keahlian pada Sekolah Menengah Kejuruan (SMK). Meskipun demikian, sebenarnya isi dari buku ini tidak secara khusus membahas tentang Rekayasa Perangkat Lunak. Dari sisi pandang bidang IImu Komputer ada lima sub-bidang yang tercakup dalam dalam buku ini, yaitu sub-bidang Rekayasa Perangkat Lunak, Sistem Operasi, Algoritma dan Struktur Data, Bahasa Pemrograman dan Basis Data. Hal ini disesuaikan dengan kurikulum tingkat SMK untuk Program Keahlian Rekayasa Perangkat Lunak.

Pokok bahasan tentang Rekayasa Perangkat Lunak secara umum membahas dasar-dasar pengertian Rekayasa Perangkat Lunak, masalah dan pemecahan masalah, dan metode-metode pengembangan perangkat lunak. Pembahasan tentang sub-bidang Sistem Operasi berisi sistem computer, sistem operasi dan bekerja dalam jaringan computer. Cakupan materi algoritma meliputi algoritma dasar dan algoritma lanjutan. Sub bidang Bahasa Pemrograman mengambil porsi yang cukup besar, meliputi pemrograman GUI dengan VB & VB.Net, pemrograman Java, pemrograman C++, pemrograman berorientasi obyek dan Pemrograman berbasis web. Sub-bidang terakhir yang menjadi bagian dari buku ini adalah Basis Data dengan cakupan tentang system basis data, pemodelan konseptual, basis data relasional, Microsoft Access dan SQL.

B. Peta Kompetensi

Secara umum, buku ini mengacu pada Standar Kompetensi dan Kompetensi Dasar (SKKD) bagi SMK seperti berikut.

1. Menggunakan algoritma pemrograman tingkat dasar
2. Menggunakan algoritma pemrograman tingkat lanjut
3. Mengoperasikan aplikasi basis data
4. Membuat aplikasi berbasis Microsoft Access
5. Menguasai teknik elektronika dasar
6. Menguasai teknik elektronika digital
7. Membuat file dengan HTML sesuai spesifikasi
8. Menerapkan dasar-dasar pembuatan web statis tingkat dasar
9. Membuat program aplikasi menggunakan VB dan VB.NET
10. Membuat paket software aplikasi
11. Melakukan pemrograman data deskripsi (SQL – Stuctured Query Language) tingkat dasar
12. Mengoperasikan bahasa pemrograman data deskripsi (SQL) tingkat lanjut
13. Membuat halaman web dinamis tingkat dasar
14. Membuat halaman web dinamis tingkat lanjut
15. Membuat program aplikasi web menggunakan JSP

16. Membuat program aplikasi basis data menggunakan XML
17. Membuat program basis data menggunakan Microsoft (SQL Server)
18. Membuat program basis data menggunakan PL/SQL (Oracle)
19. Membuat program aplikasi menggunakan C++
20. Menjelaskan sistem peripheral
21. Membuat program dalam bahasa pemrograman berorientasi obyek
22. Membuat program aplikasi menggunakan Java
23. Mengoperasikan sistem operasi komputer berbasis teks dan GUI

Dalam penyajian buku ini, bab-bab tidak disusun berdasarkan SKKD, akan tetapi disusun berdasarkan urutan materi pokok bahasan. Sehingga di beberapa bab berisi gabungan dari beberapa standar kompetensi. Atau satu kompetensi dasar mungkin berada tidak pada kelompok standar kompetensi seperti pada daftar SKKD, tetapi berada pada sub bab yang lain.

Kesesuaian SKKD dan isi bab dapat dilihat pada table berikut ini.

Kode Kompetensi	Kompetensi	Bab Terkait
ELKA-MR.UM.001.A	Menguasai Teknik Dasar Elektronika	**3**
ELKA.MR.UM.004.A	Menguasai Dasar Elektronika Digital dan Komputer	**3**
TIK.PR02.001.01	Menggunakan algoritma pemograman tingkat dasar	**5**
TIK.PR02.002.01	Menggunakan algoritma pemograman tingkat lanjut	**6**
HDW.OPR.103.(1).A	Mengoperasikan sistem operasi jaringan komputer berbasis teks	**4**
HDW.OPR.104.(1).A	Mengoperasikan sistem operasi jaringan komputer berbasis GUI	**4**
TIK.PR02.020.01	Mengoperasikan aplikasi basis Data	**10 dan 11**
TIK.PR08.004.01	Membuat aplikasi Berbasis Microsoft Acces	**11**
TIK.PR08.024.01	Membuat dokumen dengan HTML sesuai spesifikasi	**13**
TIK.PR08.027.01	Menerapkan dasar-dasar pembuatan web statis tingkat dasar.	**13**
TIK.PR08.003.01	Membuat program aplikasi menggunakan VB & VB.NET	**7**
TIK.PR02.016.01	Membuat paket software Aplikasi	**7**
TIK.PR03.001.01	Mengoperasikan bahasa pemrograman data deskripsi (SQL) tingkat dasar	**12**
TIK.PR03.002.01	Mengoperasikan bahasa pemrograman data deskripsi (SQL) tingkat Lanjut	**12**
TIK.PR04.002.01	Membuat Halaman Web dinamis tingkat dasar	**13**
TIK.PR04.003.01	Membuat Halaman Web dinamis tingkat Lanjut.	**13**

Kode Kompetensi	Kompetensi	Bab Terkait
TIK.PR02.009.01	Mengoperasikan bahasa pemograman berorientasi obyek	**8**
TIK.PR08.012.01	Membuat program aplikasi menggunakan Java	**8**
TIK.PR08.001.01	Membuat program aplikasi menggunakan C++	**9**
TIK.PR06.003.01	Menjelaskan sistem Peripheral	**3**
TIK.PR08.005.01	Membuat program basis data menggunakan PL/SQL	**10 dan 12**
TIK.PR08.006.01	Membuat program basis data menggunakan SQL Server	**12**
TIK.PR08.008.01	Membuat program aplikasi web berbasis JSP	**14**

C. Cara Menggunakan Buku

Buku ini secara khusus ditujukan kepada siswa dan guru SMK untuk program keahlian RPL. Namun demikian, buku ini juga terbuka bagi pembaca umum yang berminat dalam dunia RPL, Algoritma dan Pemrograman, Basis Data dan Internet. Bagi siswa, buku ini dapat dijadikan buku pegangan, karena ini buku ini menyediakan bahan-bahan pelajaran yang cukup lengkap untuk mata pelajaran selama tiga tahun di bangku sekolah. Beberapa bagian dari buku ini mungkin memerlukan buku-buku bantu lainnya untuk lebih memperkaya wawasan dan peningkatan kemampuan. Sedangkan bagi guru, buku ini dapat digunakan sebagai buku referensi untuk menyusun modul-modul ajar bagi anak didiknya.

Buku ini disusun sedemikian rupa agar siswa dapat belajar secara mandiri dan terdorong untuk mencoba secara langsung. Oleh karena itu dalam buku ini, akan banyak dijumpai ilustrasi baik yang berupa gambar, skema maupun *listing program.* Hal ini dimaksudkan agar siswa dapat dengan mudah memahami penjelasan ataupun penerapan suatu konsep tertentu. Bahkan pada bagian akhir bab diakhiri dengan soal-soal latihan dari pokok bahasan pada bab tersebut.

BAB 11 APLIKASI BASIS DATA BERBASIS MICROSOFT ACCESS

Gambar 11.1. Microsoft Access 2007.

Gambar di atas adalah tampilan awal dari Microsoft Access terbaru yaitu versi 2007. Perangkat lunak yang termasuk dalam *Micosoft Office Suite* ini mungkin perangkat lunak yang jarang digunakan orang meskipun telah tersedia pada paket Micosoft Office. Padahal perangkat lunak ini sangat bermanfaat banyak bila digunakan.

Bab ini membahas dua standar kompetensi yaitu mengoperasikan aplikasi basis data dan membuat aplikasi berbasis Microsoft Access. Standar kompetensi mengoperasikan aplikasi basis data terdiri dari tiga kompetensi dasar, yaitu menjelaskan menu aplikasi basis data, membuat tabel dan membuat view atau query. Sedangkan standar kompetensi membuat aplikasi berbasis Microsoft Access terdiri dari empat kompetensi dasar, yaitu menjelaskan *Database Management System*, menjelaskan *Data Definition Language*, menerapkan *query* dan menerapkan *reporting*. *Database Management System* telah kita bahas pada Bab 10, sedangkan *Data Definition Language* akan kita bahas pada Bab 12. Sebelum mempelajari kompetensi ini ingatlah kembali tentang prinsip pemecahan masalah, sistem operasi, dan dasar-dasar basis data pada bab-bab sebelumnya.

TUJUAN

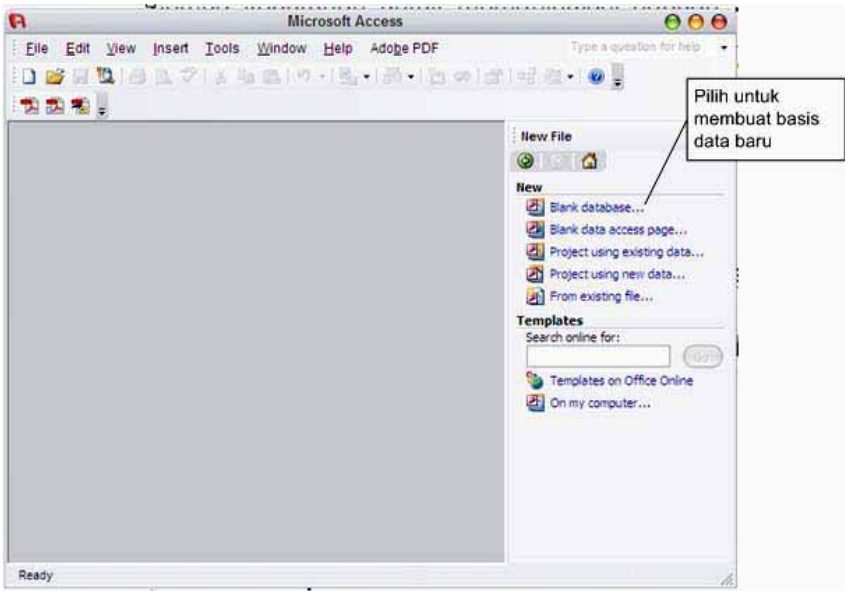
Setelah mempelajari bab ini diharapkan pembaca akan mampu :

- Menjelaskan menu-menu umum aplikasi basis data
- Membuat Tabel
- Membuat dan menerapkan *View / Query*
- Membuat *Form*
- Membuat *Report*

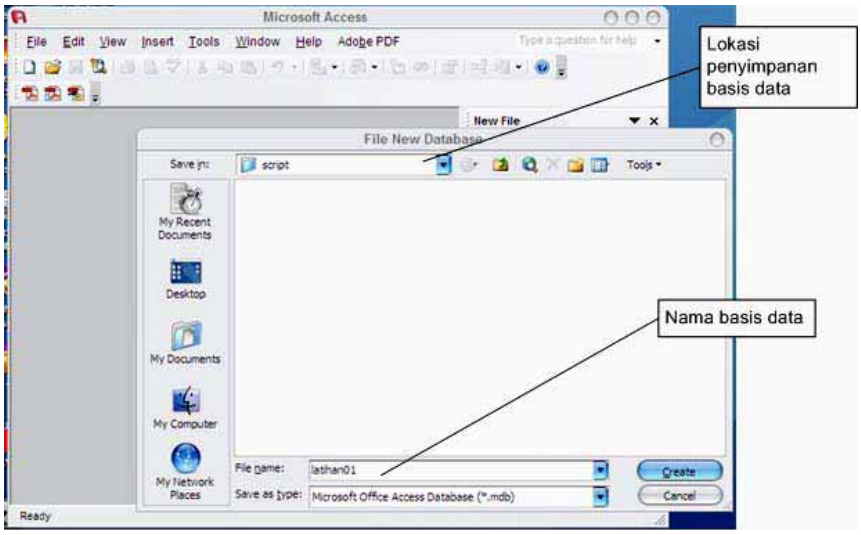
11.1. MENU-MENU UMUM APLIKASI BASIS DATA

Seperti telah disebutkan pada bab sebelumnya, Microsoft Access adalah DBMS keluaran dari Microsoft. Versi terbaru dari Access adalah versi 2007 yang termasuk dalam aplikasi Microsoft Office 2007. Format data default untuk versi terbaru ini berbeda dengan versi sebelumnya. Ekstension file sebelumnya adalah .mdb, namun sekarang berganti dengan .accdb. File basis data yang dibuat oleh versi terbaru ini tidak dapat dibaca oleh versi sebelumnya. Namun versi terbaru ini dapat membaca file basis data versi sebelumnya.

Untuk memulai Microsoft Access, kita dapat melakukan klik Start pada Windows pilih Programs lalu pilih Microsoft Access. Tampilan awal Microsoft Access akan tampak seperti Gambar 11.2. Microsoft Access memberikan beberapa opsi dalam pembuatan basis data seperti tampak pada Gambar 11.2. Namun opsi yang paling sering kita gunakan adalah *Blank database*. Apabila kita klik pada bagian ini maka akan muncul permintaan untuk mendefinisikan basis data seperti terlihat pada Gambar 11.3. Nama dan lokasi penyimpanan basis data harus ditentukan.

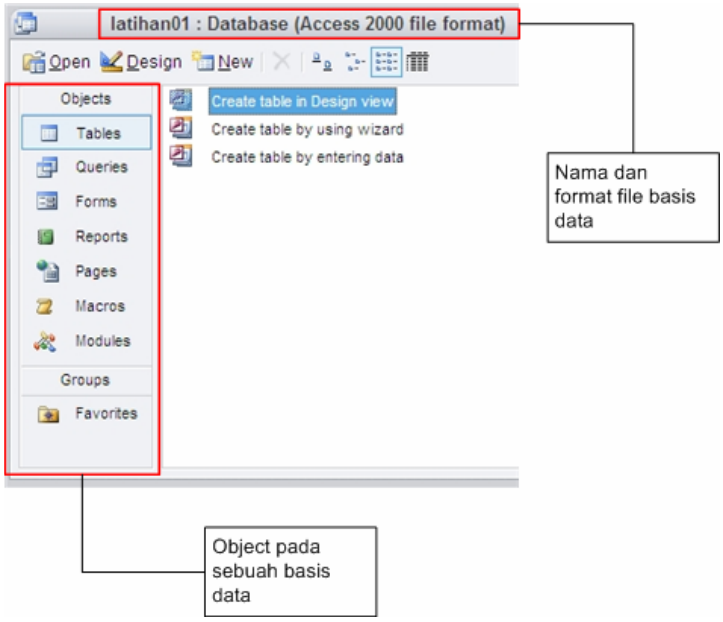


Gambar 11.2. Tampilan awal Microsoft Access.



Gambar 11.3. Penentuan nama dan lokasi basis data.

Setelah kita tekan tombol *Create* berarti kita telah mempunyai sebuah basis data, namun masih belum terisi tabel atau data apapun (Gambar 11.4). Pada gambar tersebut dapat kita lihat nama file basis data adalah *latihan01* dan format file basis data menggunakan Access 2000. Pada gambar tersebut juga tampat bagian bagian (*object*) basis data tersebut.

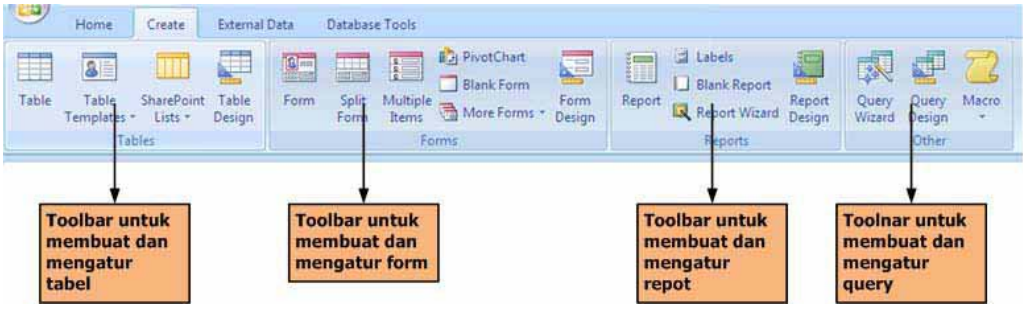


Gambar 11.4. Bagian-bagian sebuah basis data pada Microsoft Access.

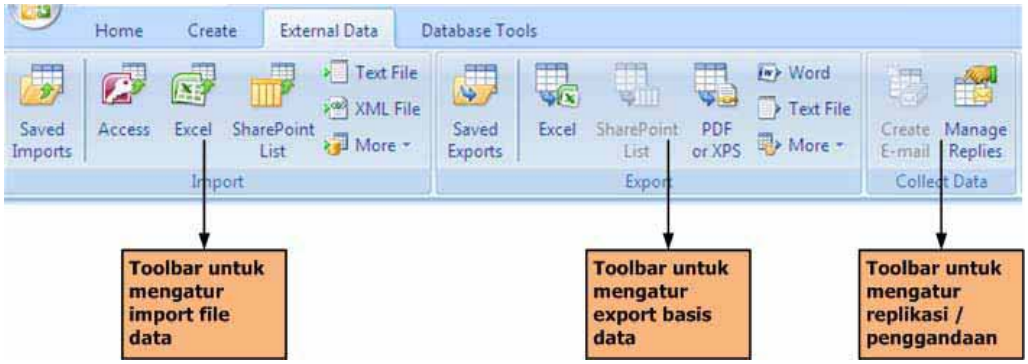
Ada enam obyek penting Access yang menjadi fitur utama dari DBMS ini, yaitu:

- *Table.* Tabel adalah tempat dimana kita menyimpan data. Semua tabel di dalam Access mengikuti aturan basis data relasional yang terdiri dari baris dan kolom. Setiap basis data bisa berisi lebih dari satu tabel.
- *Queries*. Fitur *queries* disediakan untuk memilih data yang akan kita tampilkan. Queries pada Access disediakan baik dalam bentuk GUI maupun dalam bentuk bahasa SQL.
- *Forms*. Fitur form disediakan untuk membuat tampilan dari basis data yang dibuat menjadi lebih menarik. Baik ketika mengedit data maupun tampilan output data di layar monitor.
- *Reports.* Fitur ini disediakan untuk membuat format pencetakan pada media kertas melalui printer.
- *Macros.* Fitur *macro* merupakan fitur yang digunakan untuk menyimpan perintah-perintah otomatis tertentu yang berhubungan dengan basis data yang dibuat. Dibutuhkan kemampuan pemrograman untuk menggunakan fasilitas ini.
- *Modules*. Fitur ini lebih luas dari *macro* karena kita dapat melakukan pemrograman pada banyak aspek dalam Microsoft Access.

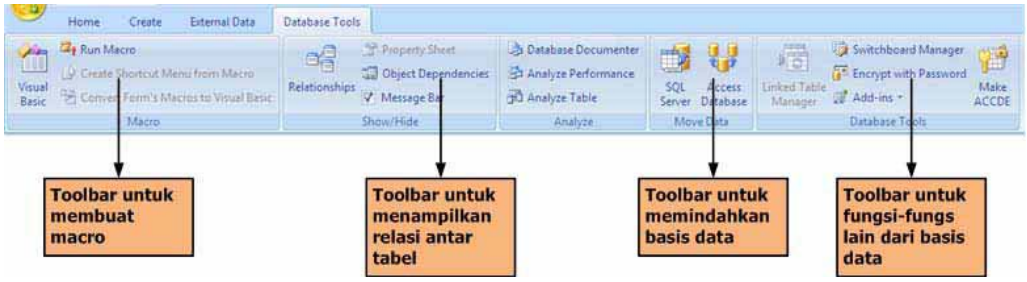
Selain obyek-obyek utama di atas Microsoft Access juga menyediakan seperangkat alat untuk mendukung kemudahan dalam membuat basis data dan aplikasinya. Berikut ini gambar-gambar *toolbar* pada Microsoft Access beserta kegunaannya.



Gambar 11.5. Toolbar pada menu Create.



Gambar 11.6. Toolbar pada menu External Data.



Gambar 11.7. Toolbar pada menu Database Tool.

Pada buku ini kita akan mempelajari empat buah obyek yaitu, *tabel, query, form* dan *report*. Namun, sebelum kita memulai dengan pembahasan tentang bagaimana menggunakan obyek-obyek dalam Microsoft Access tersebut, kita akan sekilas membahas tentang contoh kasus basis data yang akan kita buat. Kasus yang akan kita buat adalah *Basis Data Penjualan Buku*. Pada kasus ini setiap pembeli akan melakukan pembelian terhadap buku yang diinginkan dan membayar sejumlah uang sesuai buku yang dibeli. Data pembeli akan dicatat. Demikian juga setiap data pesanan, baik itu pemesanan ringkasan maupun item-item pemesanannya. Data pesanan berisi pembeli yang melakukan pemesanan, total pembelian dan tanggal pembelian. Sedangkan data item pemesanan berisi data buku yang dipesan dan jumlahnya untuk tiap pemesanan.

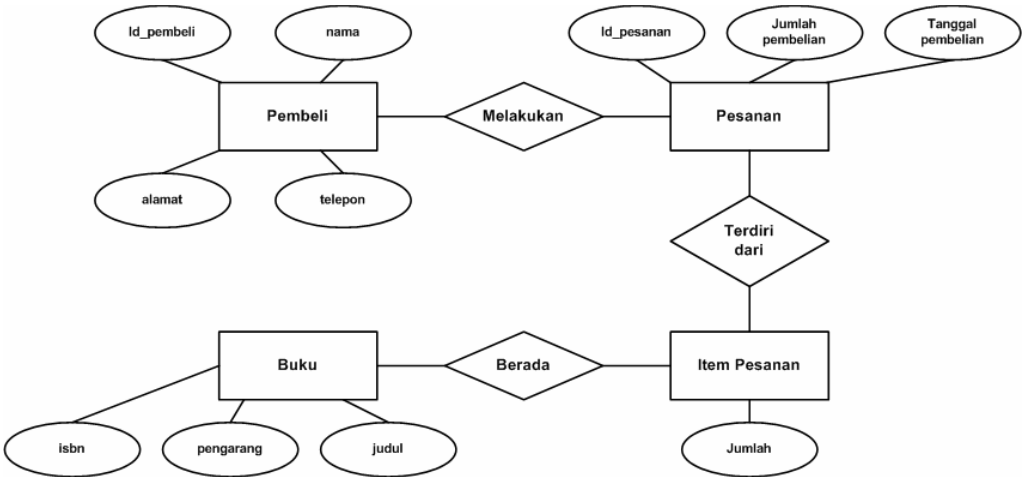
Dengan membaca kasus di atas, maka apabila kita akan membuat ER Diagramnya maka langkah pertama adalah identifikasi kandidat entitas yang terlibat. Dari teks di atas kita dapat mengidentifikasi ada minimal 4 kandidat entitas yaitu *pembeli, buku, pesanan* dan *item pemesanan*. Sedangkan relasinya dapat kita identifikasi sebagai berikut :

- pembeli melakukan pemesanan
- pada setiap pesanan terdapat item-item pesanan
- pada item-item pesanan terdapat daftar buku-buku yang dipesan.

Atribut-atribut untuk masing-masing kandidat entitas dapat kita tentukan sebagai berikut:

- *Entitas Pembeli* dengan atribut *id_pembeli, nama, alamat*, dan *telepon*.
- *Entitas Buku* dengan atribut nomor *ISBN, pengarang*, dan *judul*
- *Entitas Pesanan* dengan atribut *id_pesanan, jumlah pembelian, tanggal pembelian.*
- *Entitas Item Pemesanan* dengan atribut jumlah masing-masing buku yang dipesan.

Setelah semua informasi lengkap maka kita dapat menggambarkan ER Diagram untuk kasus di atas seperti pada Gambar 11.8.



Gambar 11.8. ER Diagram untuk kasus Basis Data Penjualan Buku.

Berdasarkan ER Diagram di atas kita dapat menentukan tabel-tabel apa yang dibutuhkan pada basis data penjualan buku. Ada empat tabel yaitu: tabel *pembeli, buku, pesanan*, dan *item pesanan*. Untuk mengakomodasi relasi yang ada pada ER Diagram, maka akan dibuat atribut-atribut yang berperan sebagai perwujudan relasi. Tabel 11.1 menunjukkan tabel dan atribut yang ada pada *basis data penjualan buku*.

Tabel 11.1. Tabel dan atribut pada Basis Data Penjualan Buku.

Tabel	Atribut	Keterangan
Pembeli	o id_pembeli o nama o alamat o telepon	Id_pesanan adalah atribut kunci (primary key) karena bersifat unik.
Buku	o isbn o pengarang o judul	Isbn adalah primary key karena bersifat unik

Pesanan	o id_pesanan o id_pembeli o jumlah_pembelian o tanggal_pembelian	Id_pesanan adalah primary key karena bersifat unik. Sedangkan id_pembeli adalah atribut penghubung (foreign key) entitas Pembeli dengan Pesanan. Hal ini merupakan perwujudan dari relasi Pembeli melakukan Pesanan.
Item_Pesanan	o id_pesanan o isbn o jumlah	Id_pesanan dan isbn secara bersama-sama adalah primary key untuk tabel ini. Id_pesanan sendiri adalah foreign key sebagai perwujudan relasi Pesanan terdiri dari Item_Pesanan. Sedangkan isbn adalah perwujudan relasi Buku berada dalam Item_Pesanan.

Tabel 11.1 dapat kita lengkapi dengan tipe data dan constraint/domain seperti pada Tabel 11.2. Hal ini untuk mempermudah pembuatan tabel pada DBMS. Apabila kita memeriksa apakah tabel-tabel yang terbentuk sudah dalam bentuk normal atau belum, maka kita akan menjumpai semua tabel sudah dalam bentuk normal bentuk ketiga (3NF).

Tabel 11.2. Tabel, atribut, tipe data dan constraint/domain pada Basis Data Penjualan Buku.

Tabel	Atribut	Tipe Data	Constraint/Domain
Pembeli	o id_pembeli o nama o alamat o telepon	Integer Char/Text (30) Char/Text (60) Char/Text (15)	Not Null
Buku	o isbn o pengarang o judul o harga	Char/Text (15) Char/Text (30) Char/Text (50) Real/Float (10,2)	Not Null
Pesanan	o id_pesanan o id_pembeli o jumlah_pembelian o tanggal_pembelian	Integer Integer Real/Float (10,2) Date	Not Null Not Null
Item_Pesanan	o id_pesanan o isbn o jumlah	Integer Char/Text (30) Integer	Not Null Not Null Not Null dan > 0

11.2. TABEL

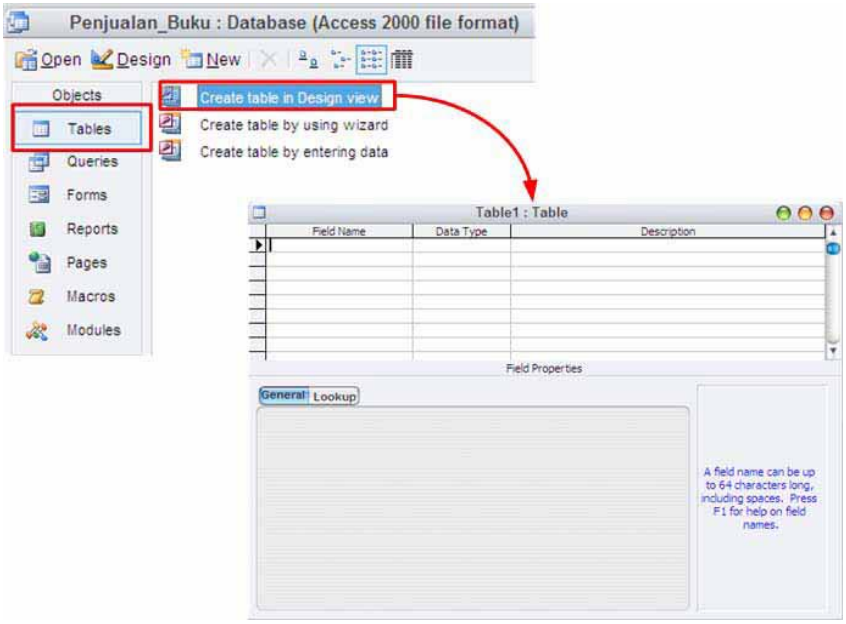
Ketika kalian selesai membuat file basis data (lihat Gambar 11.3 dan 11.4, kita sudah punya sebuah basis data namun belum berisi apa-apa karena kita belum membuat tabel-tabel dalam basis data tersebut. Seperti telah dijelaskan pada Bab 10, tabel akan berisi kolom dan baris. Kolom di sebut *field* dan baris disebut *record* dalam Microsoft Access.

11.2.1 Pendefinisian Field dan Tipe Data.

Tahap pertama dalam membuat tabel adalah mendefinisikan field-field yang dibutuhkan baru kemudian mengisi baris-baris data.

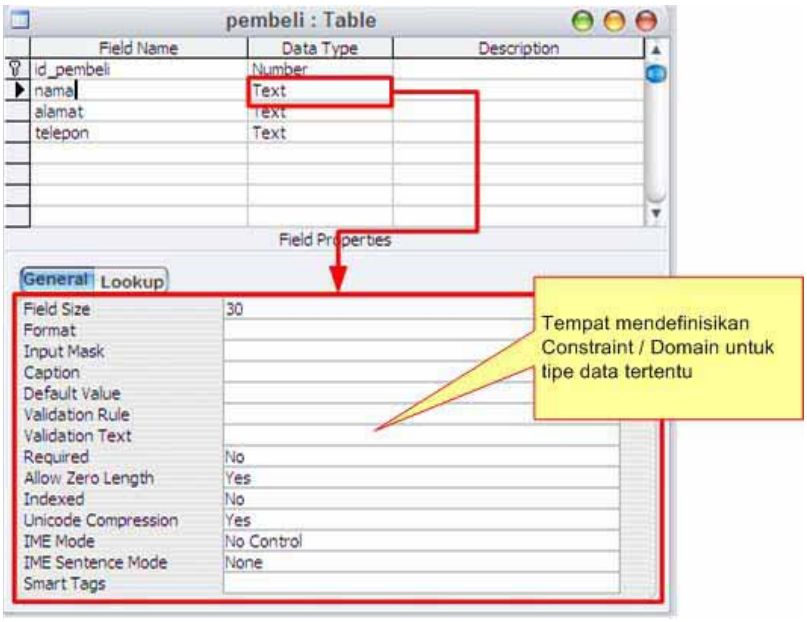
Langkah-langkah pembuatan tabel adalah sebagai berikut:

1. Pada jendela *Database* click pada *Table* (Gambar 11.9).
2. *Double click* pada *Create table in Design View* (Gambar 11.9). Kemudian akan muncul jendela untuk mendefinisikan *field-field* yang dibutuhkan (Gambar 11.9)



Gambar 11.9. Tahap awal pembuatan tabel.

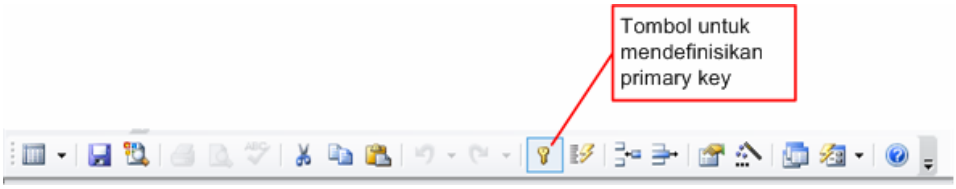
3. Kita dapat mulai memasukkan *field-field* yang dibutuhkan. Untuk contoh awal kita akan memasukkan *field-field* untuk tabel *Pembeli* seperti yang sudah didefinisikan pada Tabel 11.2. Perhatikan Gambar 11.10 berikut ini.



Gambar 11.10. Pendefinisian *field*, tipe data, *constraint* dan *domain*.

Microsoft Access menyediakan fasilitas yang sangat baik untuk mendefinisikan field-field suatu tabel. Pada Gambar 11.10 tampak proses pendefinisian field. Bagian atas adalah untuk menentukan nama field, tipe data dan keterangan. Sedangkan bagian bawah merupakan tempat menentukan lebar data, format, domain atau constraint dari suatu tipe data. Pada gambar tersebut terlihat bahwa field *nama* didefinisikan bertipe data teks, dengan lebar data 30 (lihat bagian bawah gambar).

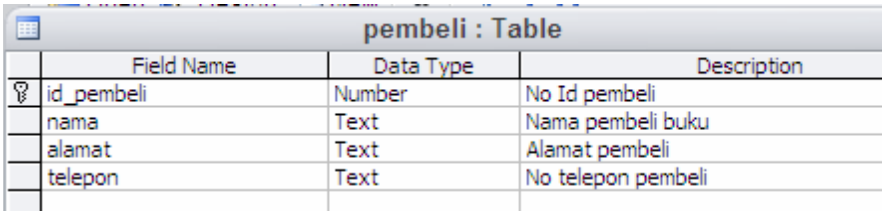
4. Setelah semua field untuk tabel *pembeli* selesai didefinisikan maka kita harus menentukan field mana yang berperan sebagai *primary key*. Pada penjelasan kasus di atas kita sudah menetapkan bahwa *id_pembeli* akan menjadi *primary key*. Pilih / sorot baris *id_pembeli*, kemudian click tombol bergambar kunci (Gambar 11.11) pada toolbar Microsoft Access.



Gambar 11.11. Toolbar Microsoft Access.

5. Kita dapat menyimpan tabel yang sudah kita definisikan dan memberi nama tabel tersebut dengan cara menekan tombol bergambar disket (lihat Gambar 11.11). Kemudian kita dapat menutup jendela *Design View*

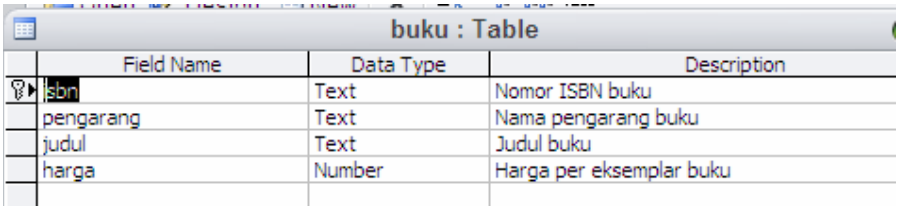
tersebut. Dengan cara yang sama, tabel-tabel lainnya yaitu tabel *buku*, tabel *pesanan*, dan tabel *item_pesanan* dapat kita definisikan. Gambar-gambar berikut menunjukkan hasil pendefinisian keseluruhan tabel.



pembeli : Table

Field Name	Data Type	Description
id_pembeli	Number	No Id pembeli
nama	Text	Nama pembeli buku
alamat	Text	Alamat pembeli
telepon	Text	No telepon pembeli

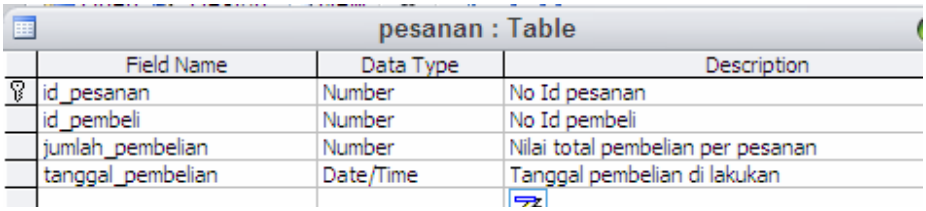
Gambar 11.12. Struktur tabel *pembeli*.



buku : Table

Field Name	Data Type	Description
isbn	Text	Nomor ISBN buku
pengarang	Text	Nama pengarang buku
judul	Text	Judul buku
harga	Number	Harga per eksemplar buku

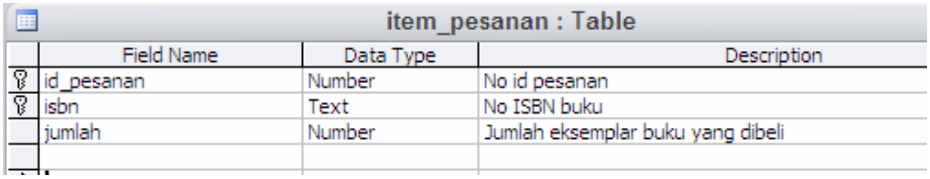
Gambar 11.13. Struktur tabel *buku*.



pesanan : Table

Field Name	Data Type	Description
id_pesanan	Number	No Id pesanan
id_pembeli	Number	No Id pembeli
jumlah_pembelian	Number	Nilai total pembelian per pesanan
tanggal_pembelian	Date/Time	Tanggal pembelian di lakukan

Gambar 11.14. Struktur tabel *pesanan*.



item_pesanan : Table

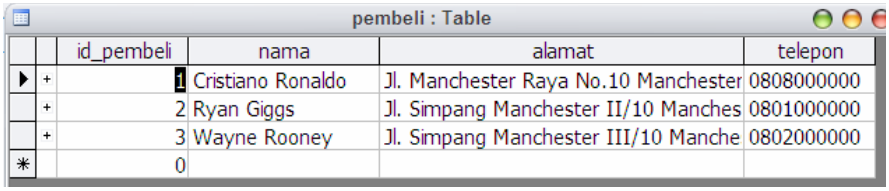
Field Name	Data Type	Description
id_pesanan	Number	No id pesanan
isbn	Text	No ISBN buku
jumlah	Number	Jumlah eksemplar buku yang dibeli

Gambar 11.15. Struktur tabel *item_pesanan*.

11.2.2.Pengisian Data Pada Tabel.

Tabel-tabel yang telah kita buat di atas, belum mempunyai isi data apa-apa. Hanya strukturnya yang telah kita buat. Untuk mengisikan data pada tabel, caranya cukup mudah, yaitu: *double click* pada nama tabel yang kita ingin

isikan datanya. Setelah terbuka jendela seperti pada Gambar 11.16, kita dapat segera mengisikan datanya. Cara pengisian datanya sama seperti kalau kita bekerja dengan Microsoft Excell atau *software spreadsheet* lainnya. Kemudian kita dapat menyimpan hasil pengisian data dengan menekan tombol bergambar disket pada *toolbar*.



pembeli : Table

id_pembeli	nama	alamat	telepon
1	Cristiano Ronaldo	Jl. Manchester Raya No.10 Manchester	0808000000
2	Ryan Giggs	Jl. Simpang Manchester II/10 Manches	0801000000
3	Wayne Rooney	Jl. Simpang Manchester III/10 Manche	0802000000
0			

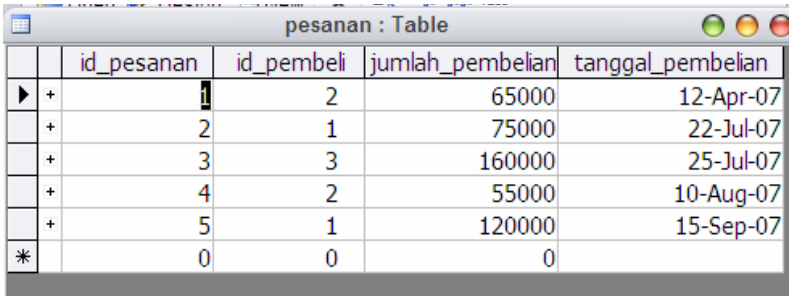
Gambar 11.16. Hasil pengisian data pada tabel *pembeli*.

Gambar 11.16 menunjukkan isi data pada tabel *pembeli.* Dengan cara yang sama kita dapat mengisikan data pada tabel-tabel lain. Perhatikan hasil pengisian data pada gambar-gambar berikut.

buku : Table

isbn	pengarang	judul	harga
0-672-31337-	Juande Ramos	Sejarah Sepakbola	105000
0-672-31637-	Harry Redknapp	Teori Sepakbola Pra Sejarah	55000
0-672-31667-	Arsene Wenger	Teori Sepakbola Kuno	75000
0-672-31667-	Harry Redknapp	Teori Sepakbola Modern	65000
			0

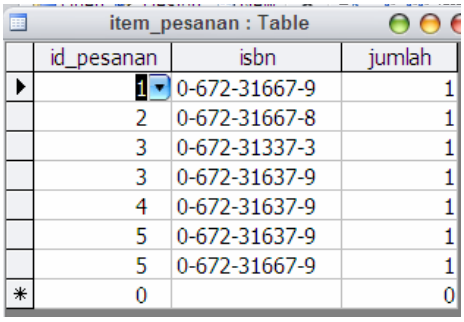
Gambar 11.17. Hasil pengisian data pada tabel *buku*.



pesanan : Table

id_pesanan	id_pembeli	jumlah_pembelian	tanggal_pembelian
1	2	65000	12-Apr-07
2	1	75000	22-Jul-07
3	3	160000	25-Jul-07
4	2	55000	10-Aug-07
5	1	120000	15-Sep-07
0	0	0	

Gambar 11.18. Hasil pengisian data pada tabel *pesanan*.



item_pesanan : Table

id_pesanan	isbn	jumlah
1	0-672-31667-9	1
2	0-672-31667-8	1
3	0-672-31337-3	1
3	0-672-31637-9	1
4	0-672-31637-9	1
5	0-672-31637-9	1
5	0-672-31667-9	1
0		0

Gambar 11.19. Hasil pengisian data pada tabel *item_pesanan*.

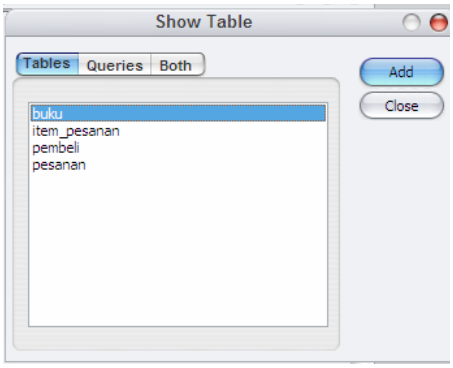
11.3 QUERY

Pada bagian ini kita akan menerapkan teori-teori *query* yang telah kita telah singgung sebelumnyanya. Seperti telah dijelaskan, *query* adalah 'permintaan data'. Dengan *query* kita dapat menampilkan data-data tertentu dari satu atau lebih tabel, atau melakukan perhitungan pada data di dalam tabel. Namun sebelum mempelajari bagaimana membuat *query*, kita akan pelajari dulu bagaimana membuat relasi antar tabel agar ketika membuat *query* menjadi lebih mudah.

11.3.1. Membuat Relasi Antar Tabel.

Pada Gambar 11.8, kita telah mendefinisikan hubungan antar entitas dalam ER Diagram. Gambar ini merupakan dasa dalam membuat relasi antar tabel. Untuk membuat relasi antar tabel, pilih menu Tools kemudian Relationship sehingga akan muncul jendela seperti pada Gambar 11.20. Pada jendela *relationships* tersebut klik kanan sehingga muncul menu pilihan dan pilih *Show Tabel*. Setelah muncul jendela seperti Gambar 11.21, kita dapat mulai menentukan tabel mana saja yang akan kita relasikan.

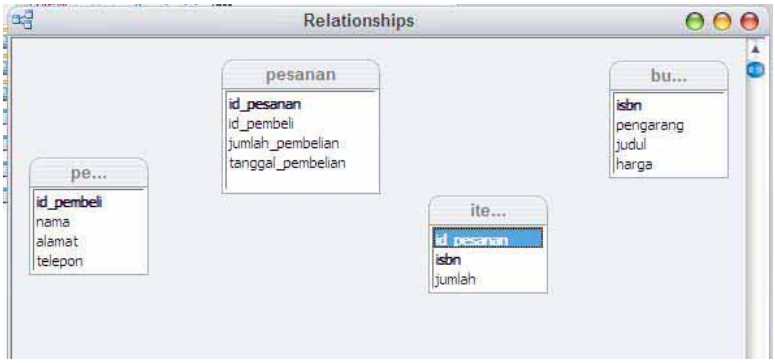
Gambar 11.20. Jendela *Relationships*.



Gambar 11.21. Jendela *Show Table.*

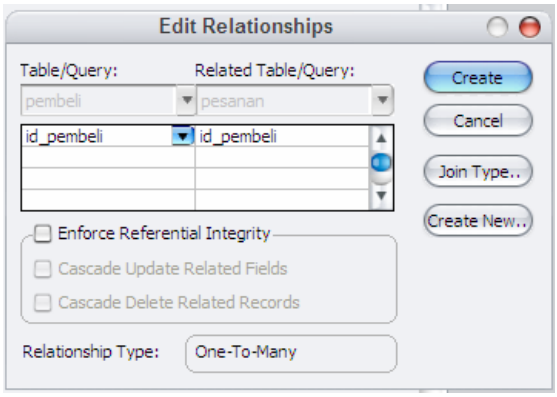
Pada Gambar 11.21, pilih tabel yang akan direlasikan kemudian *click* tombol **Add.** Setelah selesai proses pemilihan *click* tombol **Close**. Pada kasus

yang akan kita terapkan ini keempat tabel itu berhubungan langsung maupun tidak langsung. Sehingga kita memilih seluruh tabel untuk direlasikan. Gambar 11.22 menunjukkan hasil pemilihan tabel pada jendela *Relationships*. Pada gambar ini yang diperlihatkan adalah struktur dari masing-masing tabel, bukan isinya. Nama kolom yang dicetak tebal menunjukkan kolom tersebut adalah primary key. Kita dapat mengatur posisi tabel dengan cara *drag & drop*.

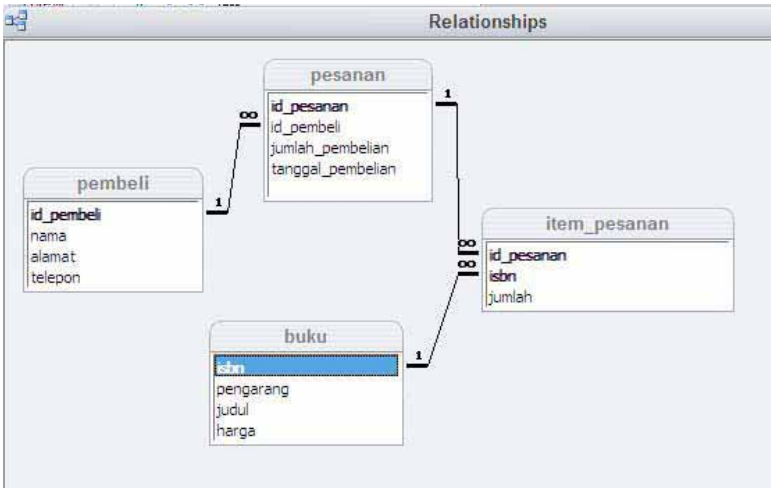


Gambar 11.22. Tabel-tabel yang akan direlasikan.

Untuk membuat relasi antar tabel, dapat dilakukan dengan memilih primary key pada suatu tabel kemudian seret mouse menuju key dengan nama yang sama pada tabel lainnya (foreign key pada tabel lain). Sebagai contoh pada tabel *pembeli*, primary key – nya *id_pembeli* dan pada tabel *pesanan*, *id_pembeli* adalah *foreign key*. *Click id_pembeli* pada tabel *pembeli* kemudian seret mouse menuju *id_pembeli* pada tabel *pesanan*. Apabila prosedur ini benar dilakukan, maka akan muncul jendela seperti Gambar 11.23. *Click* pada bagian **Enforce Referential Integrity** dan *click* tombol **Create.** Kita dapat melakukan prosedur ini pada relasi-relasi yang lain. Sehingga pada jendela *Relationships* akan tampak seperti pada Gambar 11.24.



Gambar 11.23. Jendela untuk edit relationships



Gambar 11.24. Relasi untuk keseluruhan tabel.

Gambar 11.24 menunjukkan relasi antar tabel yang dapat kita bandingkan dengan ER Diagram pada Gambar 11.8. Pada relasi antar tabel ini, kita juga menentukan kardinalitas antar tabel. Perhatikan pada garis yang menghubungkan tabel *pembeli* dengan *pesanan*. Di ujung yang berada pada tabel *pembeli* ditandai dengan angka 1 dan di ujung yang ada pada tabel *pesanan* ditandai dengan notasi ∞. Hal ini menunjukkan adanya hubungan *one-to-many* antara tabel *pembeli* dengan tabel *pesanan*. Demikian juga dengan relasi antar tabel yang lain.

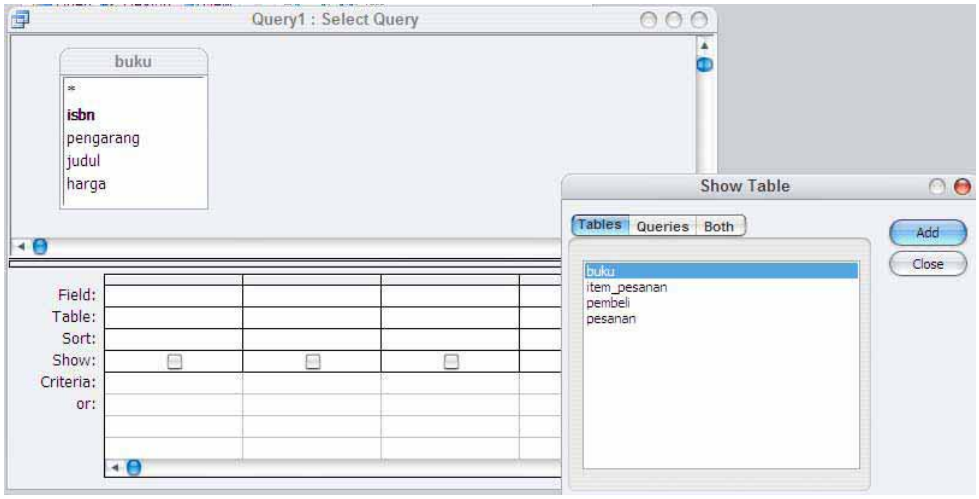
11.3.2. Membuat Query

Microsoft Access menyediakan fasilitas *query* yang sangat baik dan memudahkan pengguna. Selain karena berbasis GUI juga prosedurnya tidak terlalu rumit. Berikut contoh-contoh membuat *query* di Microsoft Access.

- *Query* pada satu tabel

  Query pada satu tabel hanya akan melibatkan satu tabel saja. Misalkan kita akan melakukan *query* pada tabel buku. Prosedur yang ditempuh adalah sebagai berikut:

  1. Pada jendela *Database* pilih *Query* lalu klik dua kali *Create Query In Design View.* Sehingga akan muncul jendela seperti pada Gambar 11.25. Pilih tabel *buku* pada jendela *Show Tabel*, kemudian click **Add** dan kemudian **Close**. Jendela *Show Table* akan tertutup dan kita dapat mulai melakukan *query*.



Gambar 11.25. Jendela *query* pada mode design view.

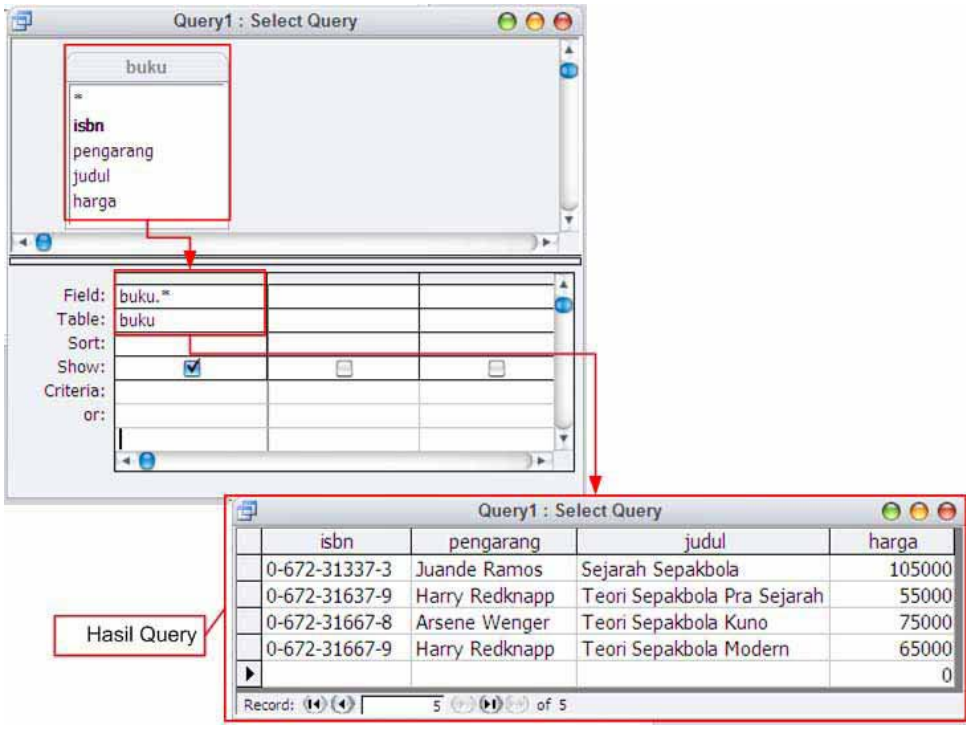
2. Pada jendela *Query* bagian bawah (lihat Gambar 11.25), ada beberapa hal penting yang harus diketahui dan berguna dalam *query* yaitu

   Field : Nama Field yang ingn ditampilkan
   Tabel : Nama Tabel dari Filed tersebut
   Sort : Mengurutkan Data hasil *query*
   Show : Mengatur Field ditampikan atau tidak
   Criteria : Syarat dari data yang ingin ditampilkan

3. Contoh *query* yang pertama adalah bagaimana menampilkan semua data, misalnya:

   Tampilkan semua data yang ada di tabel buku.

   Untuk menampilkan seluruh data pada tabel *buku*, pada Field, click tombol panah ke bawah pilih **buku.*** (lihat Gambar 11.26). Kita dapat mengeksekusi *query* dengan memilih menu *Query* kemudian click *Run*, atau *click* langsung tombol tanda seru () yang ada di *toolbar*. Hasil *query* dapat dilihat pada Gambar 11.26. Simpan *query* dengan nama yang diinginkan (misalnya: *query_buku_semua_data*) kemudian tutup jendela *Query*.

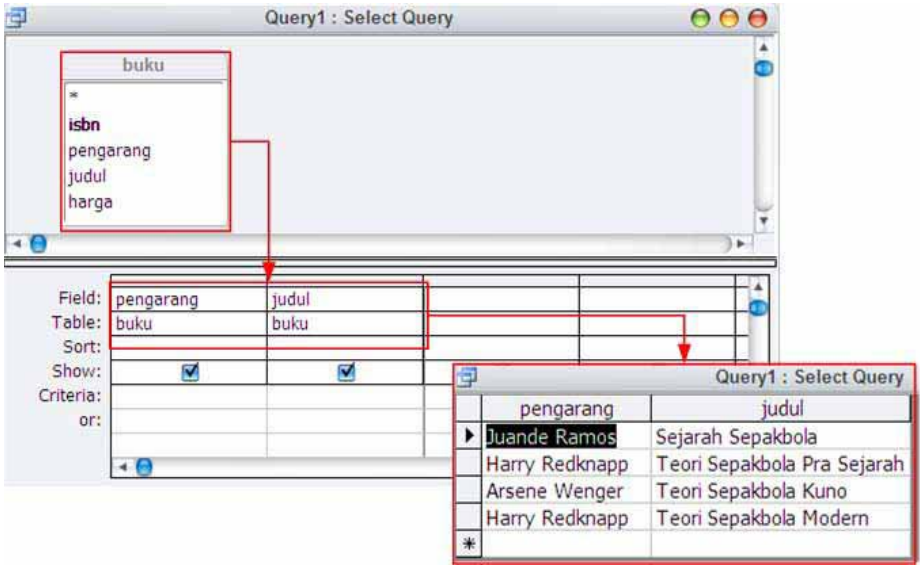


Gambar 11.26. Prosedur dan hasil *query* tabel *buku*.

4. Contoh *query* yang kedua adalah memilih kolom mana saja yang akan ditampilkan, misalnya :

   Tampilkan semua nama pengarang dan judul buku yang dikarangnya

   Query ini tidak menampilkan seluruh data tetapi hanya data dari kolom *pengarang* dan *judul* buku saja. Pada Gambar 11.27 terlihat bagaimana *query* dilakukan. Tabel buku tetap dipilih dari jendela *Show Table*, kemudian di jendela *Query*, pada bagian Field dipilih field *pengarang* dan *judul*. Hasil eksekusi *query* adalah daftar nama seluruh pengarang dan buku yang dikarangnya.

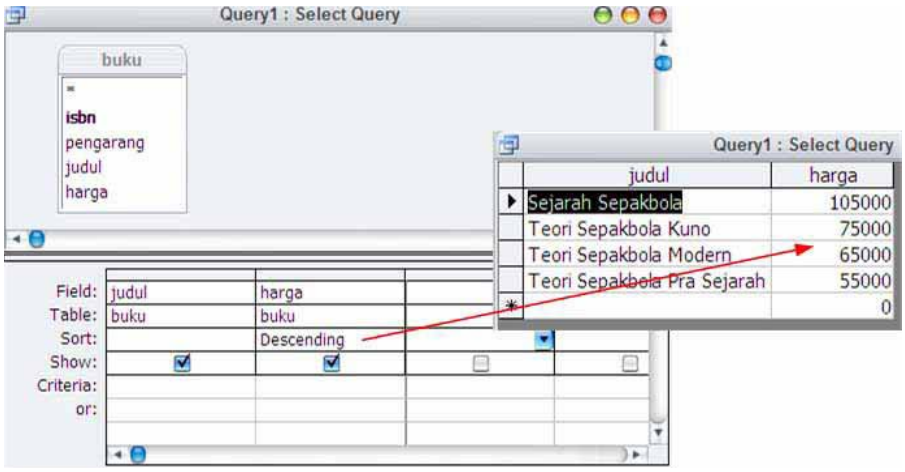


Gambar 11.27. Query nama pengarang dan bukunya.

5. Contoh *query* yang ketiga adalah bagaimana membuat tampilan data urut sesuai yang dikehendaki, misalnya:

   Tampilkan semua judul buku dan harganya dengan urutan harga yang paling mahal lebih dahulu

   Query ini juga tidak menampilkan seluruh data tetapi hanya data dari *judul buku* dan *harga* saja. Namun urutan tampilan dirubah. Pada Gambar 11.28 terlihat bagaimana query dilakukan. Tabel *buku* tetap dipilih dari jendela *Show Tabel*, kemudian di jendela *Query*, pada bagian Field dipilih field *judul* dan *harga*. Pada bagian **Sort**, pada kolom yang sama dengan harga, kita gunakan opsi **Descending** untuk mengurutkan dari besar ke kecil. Hasil eksekusi *query* adalah daftar seluruh judul buku dan harganya dengan urutan judul buku yang berharga paling mahal di atas.. Bandingkan urutan baris pada hasil dengan hasil *query* pada Gambar 11.26.

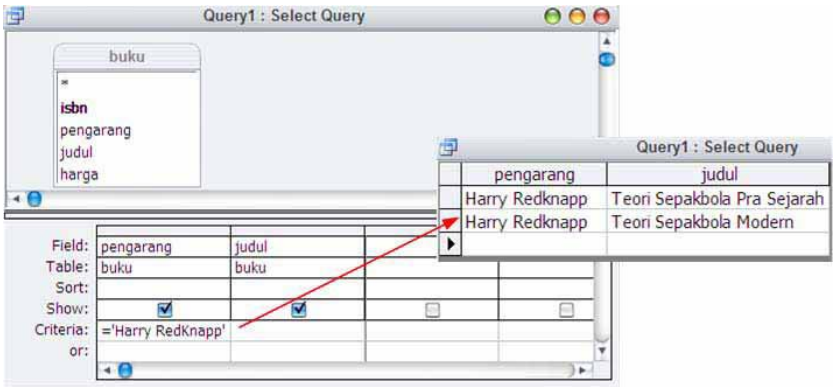


Gambar 11.28. Query judul buku dan harga dengan urutan.

6. Contoh *query* yang ketiga adalah bagaimana memilih baris-baris mana saja yang akan ditampilkan, misalnya:

   Tampilkan semua judul buku yang pengarangnya adalah Harry Redknapp.

   Query ini juga hanya berhubungan dengan data dari field *judul buku* dan pengarang saja. Namun tidak seluruh judul, tetapi hanya judul buku yang ditulis 'Harry Redknapp'. Pada Gambar 11.28 terlihat bagaimana query dilakukan. Tabel *buku* tetap dipilih dari jendela *Show Tabel*, kemudian di jendela Query, pada bagian *Field* dipilih field *nama pengarang* dan *judul*. Pada bagian criteria, kita masukkan criteria yang kita maksudkan, yaitu = 'Harry Redknapp'. Hasil eksekusi *query* adalah daftar seluruh judul buku yang ditulis 'Harry Redknapp'.

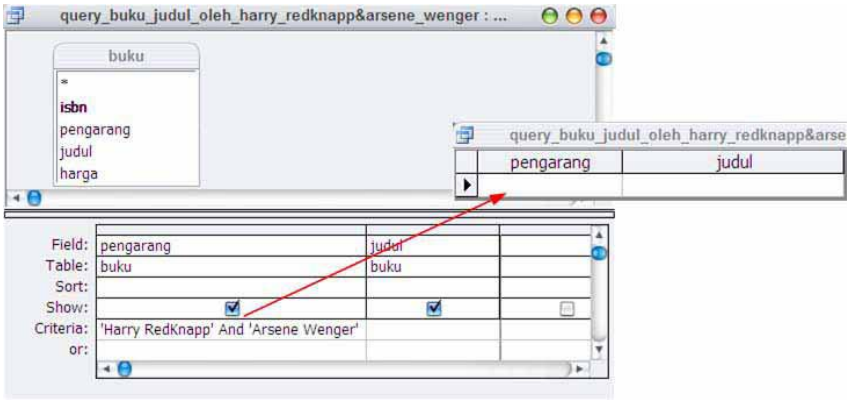


Gambar 11.29. Query dengan criteria tertentu.

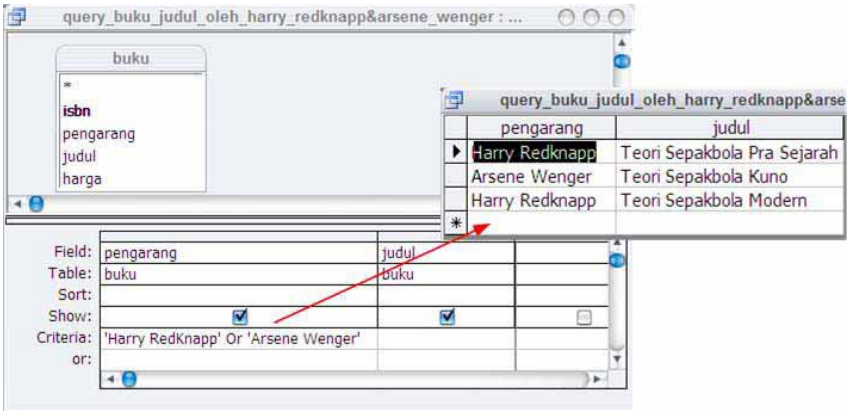
7. Contoh query yang ketiga adalah bagaimana kita menggunakan operator pada kriteria, misalnya:

   Tampilkan semua judul buku yang pengarangnya adalah Harry Redknapp dan Arsene Wenger.

   Seperti pada no 6, *query* ini juga hanya berhubungan dengan data dari field judul dan pengarang saja. Namun tidak seluruh judul, tetapi hanya judul buku yang ditulis secara bersama oleh 'Harry Redknapp dan Arsene Wenger'. Pada Gambar 11.30 terlihat bagaimana *query* dilakukan. Gunakan cara yang sama seperti pada no 6, hanya pada bagian criteria, kita masukkan criteria yang kita maksudkan, yaitu = *'Harry Redknapp'* ***and*** *'Arsene Wenger'*. Hasil eksekusi menunjukkan tidak ada satu recordpun yang memenuhi. Hal ini karena memang tidak ada buku yang ditulis secara bersama oleh 'Harry Redknapp' dan 'Arsene Wenger'. Bagaimana kalau kita ganti ***and*** dengan ***or***? Perhatikan pada Gambar 11.31. Ada 3 record yang sesuai.



Gambar 11.30. *Query* dengan menggunakan operator ***and***.



Gambar 11.31. *Query* dengan menggunakan operator ***or***.

o *Query* pada lebih dari satu tabel

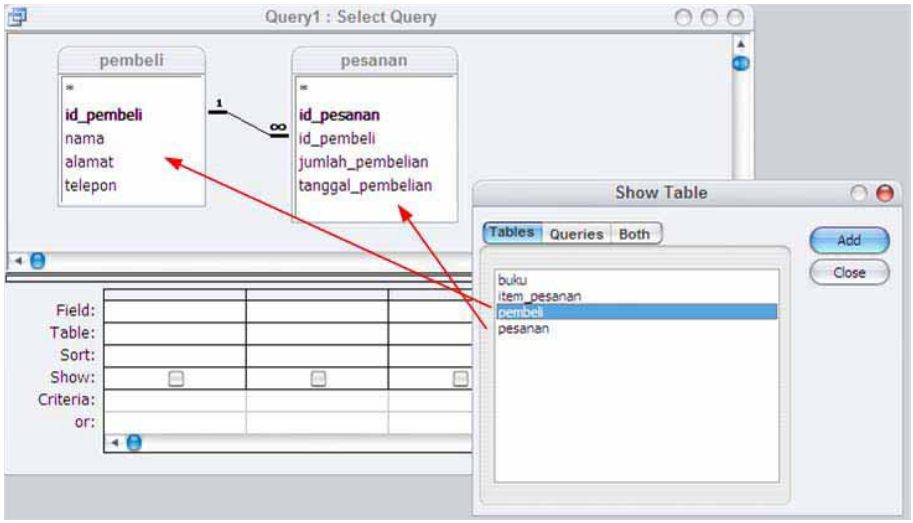
Query pada lebih dari satu tabel, relasi antar tabel yang telah kita buat sebelumnya akan menjadi sangat penting. Karena relasi ini akan menentukan bagaimana hasil dari *query*. Prosedur untuk membuat *query* sama seperti membuat *query* satu tabel, namun tabel yang dipilih pada jendela Show Tabel tidak lagi satu, tetapi mungkin dua, tiga atau lebih, sesuai dengan kebutuhan. Berikut ini beberapa contoh *query* dengan lebih dari satu tabel.

1. Contoh *query* yang pertama adalah *query* yang melibatkan dua tabel, misalnya:

   Tampilkan nama dan alamat pembeli yang jumlah pembeliannya lebih dari 100000.

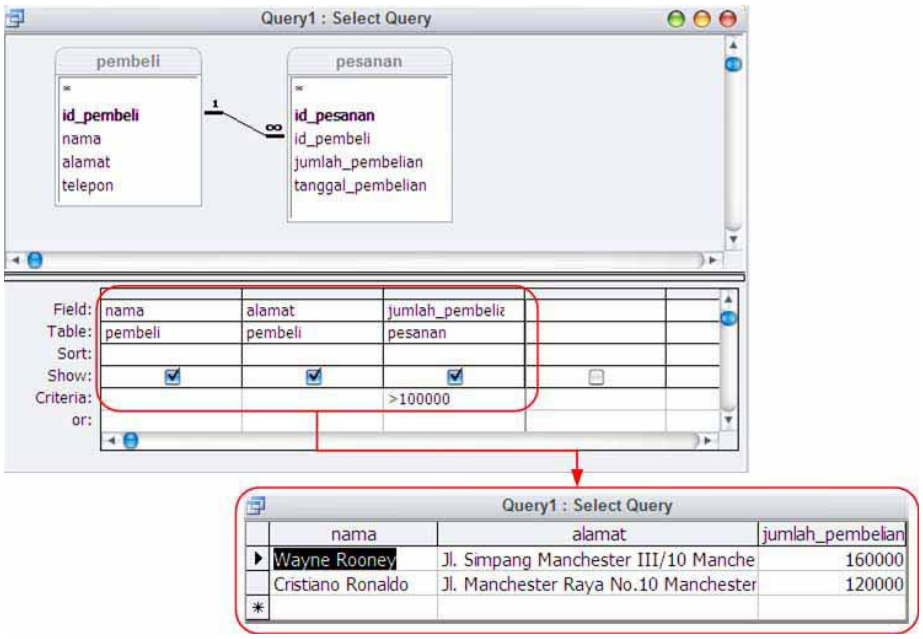
   Pada *query* ini kita membutuhkan tabel pembeli karena kolom nama dan alamat pembeli ada pada tabel pembeli. Kita juga membutuhkan tabel pesanan karena kolom jumlah pembelian ada pada tabel ini. Dari relasi antar tabel pada Gambar 11.24, kita tahu bahwa ada relasi antara tabel pembeli dan tabel pesanan dengan kardinalitas one-to-many. Prosedur membuat *query* akan seperti berikut:

   Pada jendela *Database* pilih *Query* lalu klik dua kali *Create Query In Design View.* Sehingga akan muncul jendela Show Tabel. Pilih tabel *pembeli dan pesanan* pada jendela *Show Table* (Gambar 11.32), kemudian click **Add** dan kemudian **Close**. Jendela *Show Table* akan tertutup dan kita dapat mulai melakukan *query*.



Gambar 11.32. Pemilihan tabel untuk *query* dua tabel.

Ketika kita memilih tabel pembeli dan pesanan seperti tampak pada Gambar 11.33, secara otomatis Microsoft Access akan menampilkan garis relasi antara kedua tabel tersebut. Apabila kita belum membuat relasi, maka garis relasi tidak akan muncul. Pada Gambar 11.33 terlihat pada bagian Field untuk kolom nama dan alamat, Tabel nya adalah pembeli. Sedangkan pada jumlah_pembelian, tabelnya adalah pesanan. Selain itu pada kolom jumlah_pembelian, kita juga membuat criteria, yaitu yang lebih besar dari 100000. Hasil eksekusi *query* menunjukkan ada dua orang pembeli yang jumlah_pembeliannya lebih dari 100000.



Gambar 11.33. *Query* dua tabel

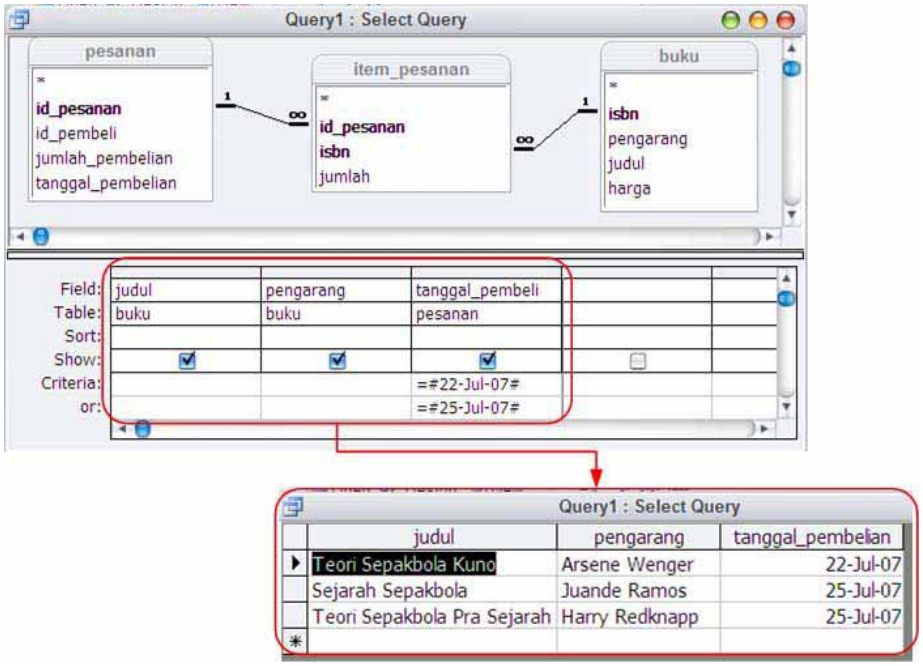
2. Contoh *query* yang kedua adalah *query* yang melibatkan tiga tabel, misalnya:

   Tampilkan judul buku dan pengarangnya yang dibeli pada tanggal 22 Juli 2007 atau 25 Juli 2007.

   Pada *query* ini kita membutuhkan tabel buku karena kolom judul dan pengarang ada pada tabel buku. Kita juga membutuhkan tabel pesanan karena kolom tanggal pembelian ada pada tabel ini. Namun dari relasi antar tabel pada Gambar 11.24, kita tahu bahwa tabel pesanan dan tabel buku tidak ada relasi langsung. Tabel pesanan berhubungan langsung dengan tabel item_pesanan dan tabel item_pesanan berhubungan langsung dengan tabel buku. Semua relasinya berkardinalitas one-to-many. Sehingga pembuatan *query* akan seperti berikut:

   Pilih tabel buku, item_pesanan dan pesanan pada jendela Show Tabel. Kemudian click **Add** dan kemudian **Close**. Seperti tampak pada Gambar

11.34 terlihat relasi antara ketiga tabel tersebut. Hasil eksekusi *query* ini menghasilkan tiga record seperti terlihat pada gambar.

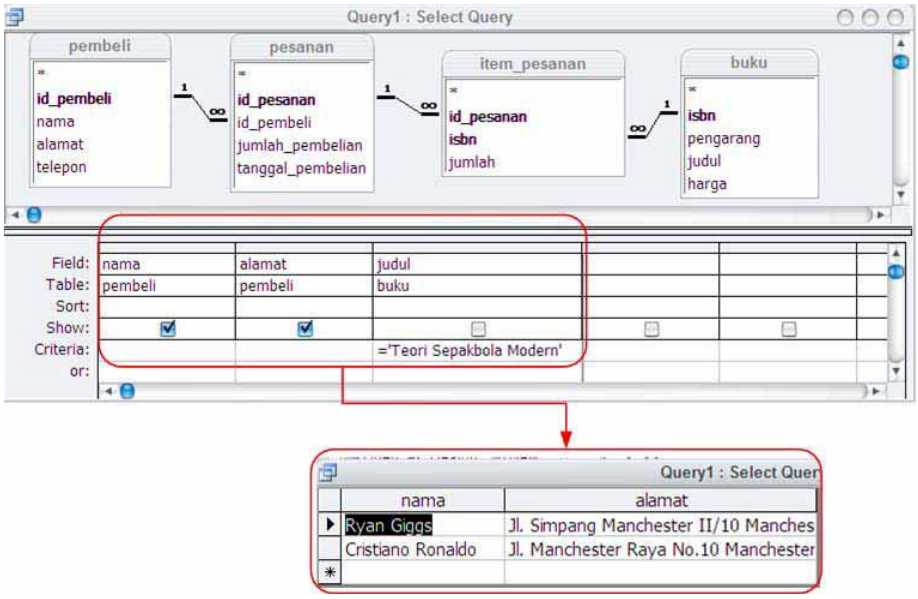


Gambar 11.34. *Query* tiga tabel.

3. Contoh *query* yang ketiga adalah *query* yang melibatkan semua tabel pada basis data penjualan buku, misalnya:

   Tampilkan nama dan alamat pembeli yang membeli buku dengan judul Teori Sepakbola Modern.

   Pada *query* ini kita membutuhkan tabel pembeli karena kolom nama dan alamat ada pada tabel pembeli. Kita juga membutuhkan tabel buku karena kolom judul hanya ada pada tabel ini. Dari relasi antar tabel pada Gambar 11.24, kita tahu bahwa tabel pembeli dan tabel buku tidak ada relasi langsung, namun harus melalui tabel pesanan dan tabel item_pesanan. Oleh karena itu kita membutuhkan keempat tabel tersebut dalam *query* ini. Pada Gambar 11.35 terlihat bagaimana *query* ini harus dibuat. Hasil eksekusi *query* ini menghasilkan tiga record seperti terlihat

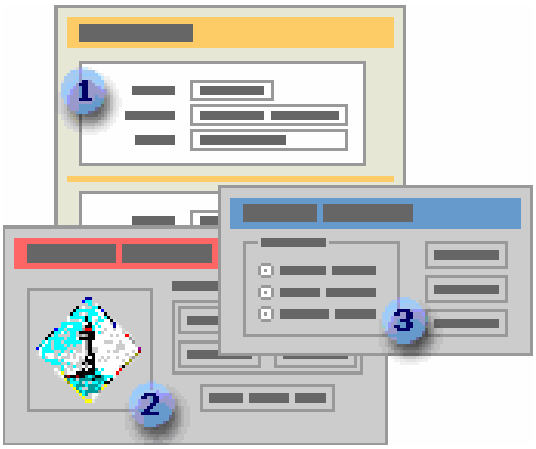


Gambar 11.35. *Query* empat tabel.

11.4. FORM

Form adalah salah satu obyek basis data dalam Microsoft Access yang digunakan sebagai antar muka bagi pengguna untuk memasukkan data atau menampilkan data. Bagi pengguna awam, memasukkan data seperti pada Gambar 11.16 sampai dengan 11.19 agak menyulitkan. Jauh lebih mudah menggunakan form. Pada Microsoft Access, dikenal ada tiga model form, yaitu: form data entry (Gambar 11.36 no 1), form switchboard (no 2) dan form custom dialog (no. 3).

Ada dua cara pembuatan form pada Micosoft Access, yaitu dengan menggunakan Wizard dan dengan menggunakan Design View. Wizard merupakan cara yang paling mudah, karena kita Microsoft Access akan melakukan pembuatan form secara otomatis. Sedangkan pada Design View kita melakukan rancangan form secara manual. Kita dapat memodifikasi hasil dari Wizard dengan Design View.

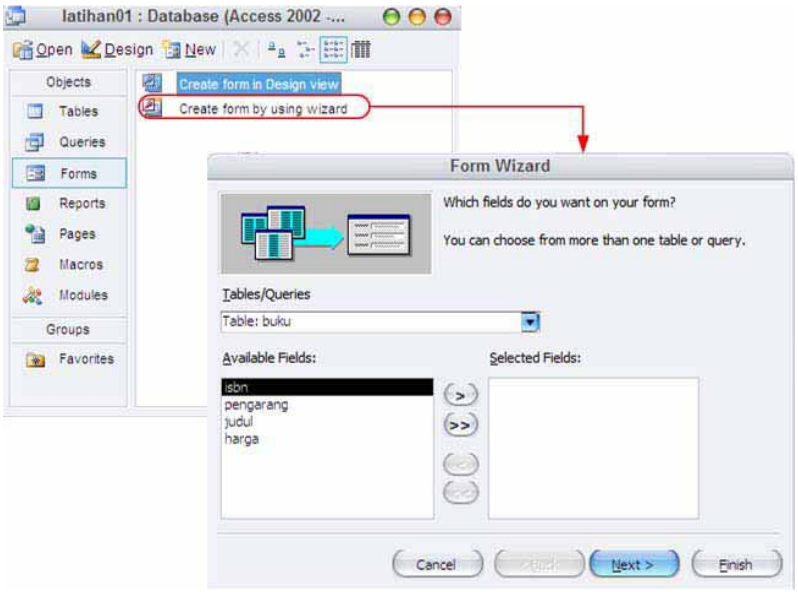


Gambar 11.36. Jenis-jenis form.

11.4.1. Membuat Form

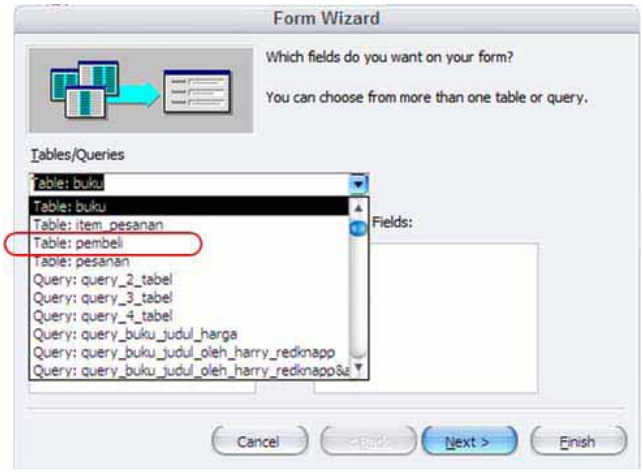
Pada bagian ini kita akan membuat form dengan mode Wizard. Form pertama yang akan kita buat adalah form untuk input data *Pembeli*.

1. Pada jendela Database, click pada object Form, kemudian double click pada Create form by using Wizard. Jendela Form Wizard akan terbuka (Gambar 11.37)

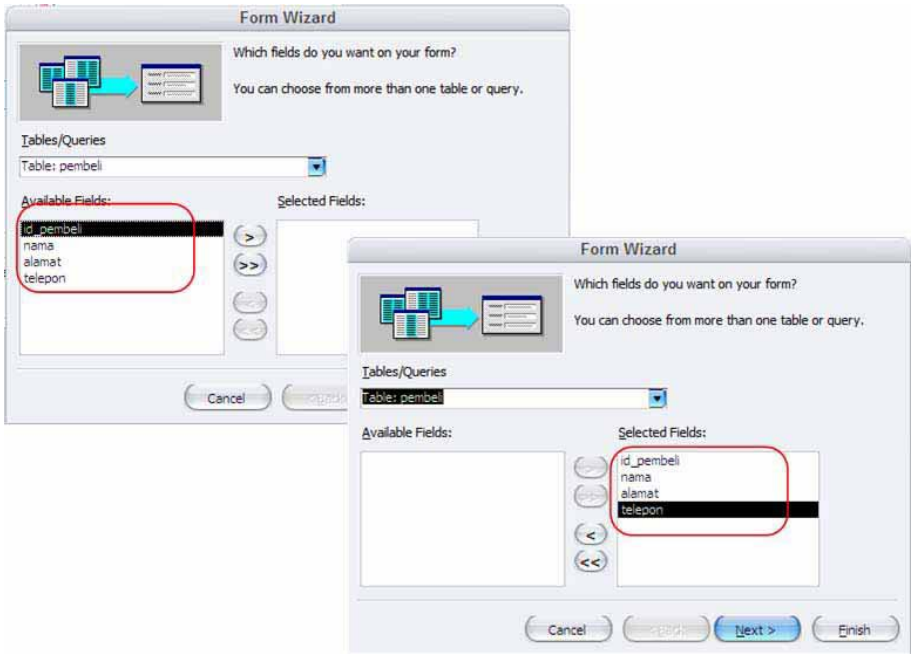


Gambar 11.37. Membuka jendela Form Wizard.

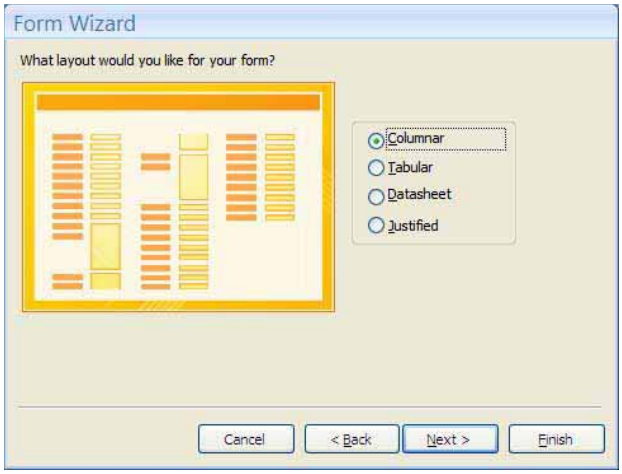
2. Pada jendela Form Wizard, click pada combo box Tabel/Queries dan pilih tabel pembeli (Gambar 11.38), kemudian click tombol >> sehingga semua field yang ada pada bagian Available Fields berpindah ke Selected Fields (lihat Gambar 11.39). Kemudian click Next untuk memunculkan jendela untuk memilih model tampilan form (Gambar 11.40)



Gambar 11.38. Pemilihan tabel yang akan dibuat formnya.

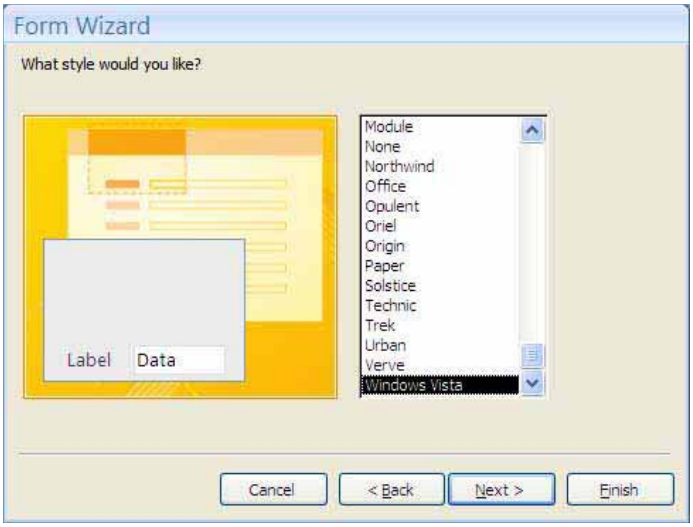


Gambar 11.39. Pemilihan field untuk form.

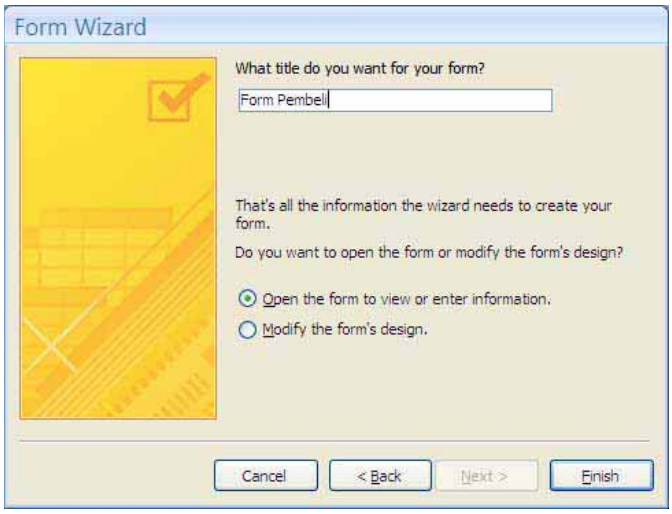


Gambar 11.40. Jendela untuk memilih model tampilan form.

3. Pada Gambar 11.40, tersedia beberapa opsi model tampilan form. Kita akan mencoba membuat dengan model Columnar. Namun pembaca dapat mencoba membuat dengan model tampilan lain yang tersaji pada pilihan. Click tombol Next maka akan muncul jendela untuk memilih style (Gambar 11.41). Pada jendela ini kita dapat memilih sesuai keinginan kita dan setelah selesai kita click Next sehingga muncul jendela untuk member Title atau judul form yang telah kita buat. Isikan nama form pada Textbox yang telah disediakan (Gambar 11.42).

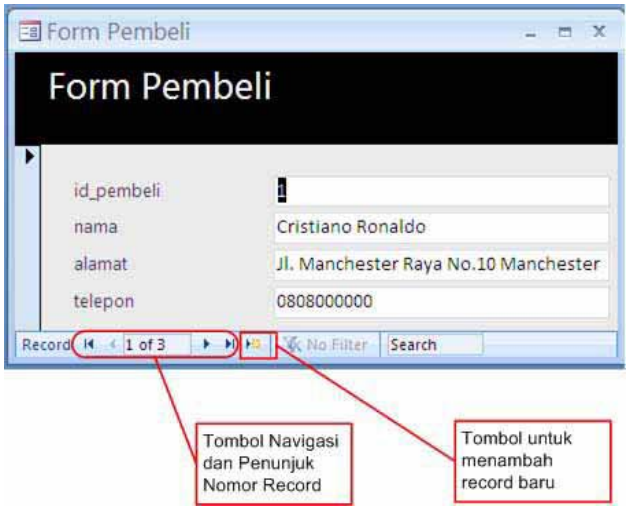


Gambar 11.41. Jendela untuk memilih style form.



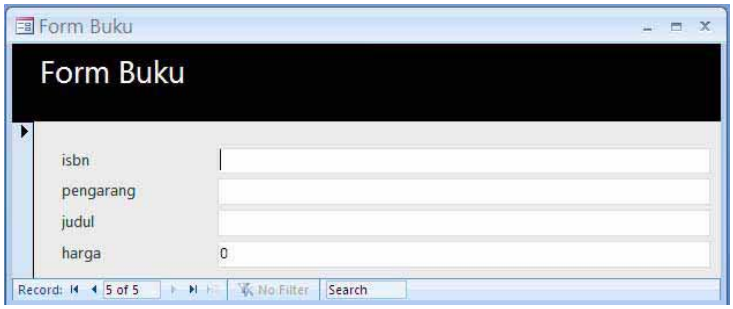
Gambar 11.42. Jendela untuk memberi nama form.

4. Click tombol Finish pada Gambar 11.42. Form pembeli telah selesai kita buat dan hasilnya tampak pada Gambar 11.43. Ada bagian yang sangat penting pada Form telah kita buat. Yaitu tombol navigasi dan tombol menambah record baru. Tombol navigasi berfungsi untuk melihat data yang telah kita buat secara urut berdasarkan id_pembeli. Sedangkan apabila tombol menambah form di-*click*, maka teks yang ada pada seluruh textbox akan dikosongkan dan kita dapat mulai mengisi data baru.



Gambar 11.43. Form Pembeli.

Dengan cara yang sama kita dapat membuat form-form yang lain. Sebagai contoh berikut ini tampilan akhir untuk Form Buku.

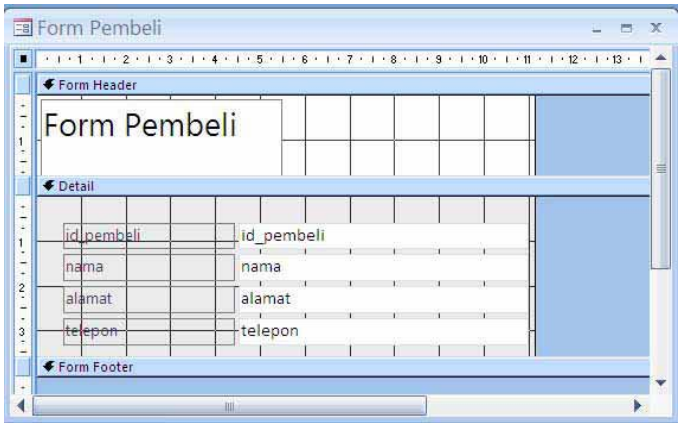


Gambar 11.44. Form Buku.

11.4.2. Memodifikasi Form

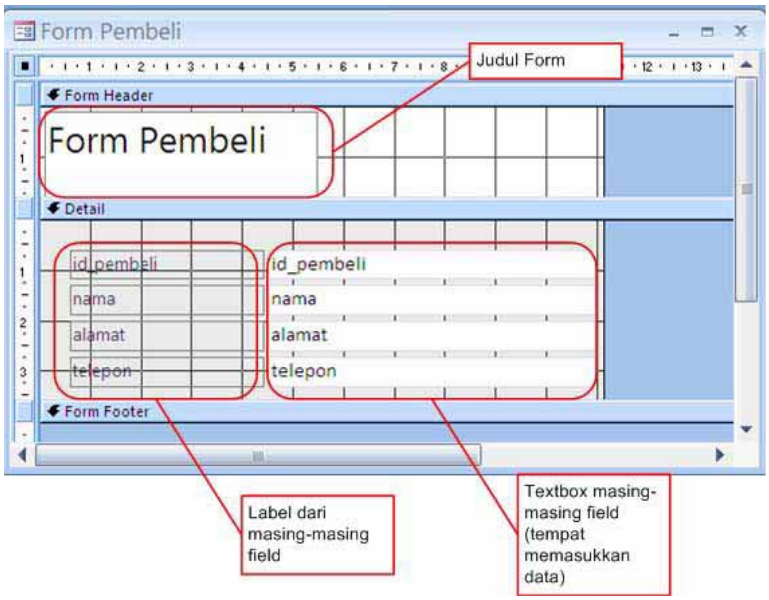
Membuat form dengan menggunakan mode wizard memiliki keuntungan yaitu, kemudahan dan kecepatan dalam pembuatan. Namun disisi lain juga memiliki kerugian karena semua format telah ditentukan secara otomatis. Sebagai contoh baik pada Gambar 11.43 dan 11.44, kita melihat nama-nama fields pada form dibuat sama persis seperti pada nama fields pada tabelnya. Untungnya pada Microsoft Access menyediakan fasilitas untuk memodifikasi form. Berikut ini langkah-langkah untuk memodifikasi form.

1. Pada jendela Database, pilih Form. Kemudian click kanan pada nama form yang ingin kita modifikasi. Pilih Design View pada pop menu yang muncul, sehingga jendela form yang akan kita modifikasi muncul seperti pada Gambar 11.45.



Gambar 11.45. Jendela Form Pembeli pada mode Design View.

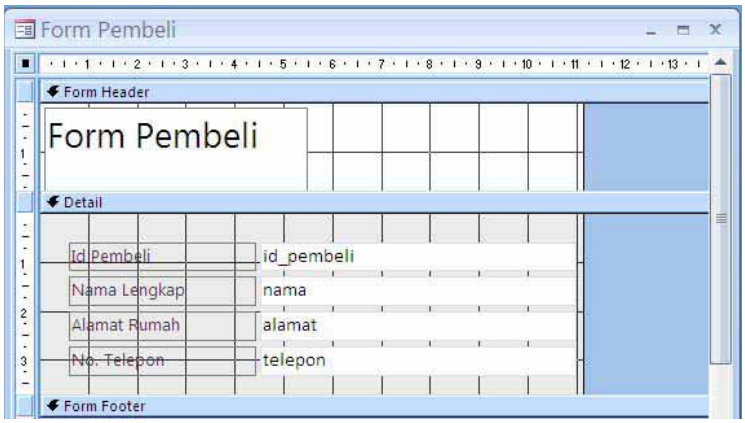
2. Pada Gambar 11.45, yang akan kita akan modifikasi adalah Form Pembeli. Sebelum kita modifikasi kita perlu tahu bagian-bagian dari suatu form. Gambar 11.46. menunjukkan bagian-bagian utama dari suatu form. Secara umum ada tiga bagian pada form, yaitu Form Header, Detail dan Form Footer. Form Header biasanya berisi judul form dan logo tertentu. Bagian Detail merupakan bagian utama karena disinilah tempat kita meletakkan field-field yang akan diisi datanya. Bagian Form Footer bersifat opsional (boleh dipakai boleh tidak). Biasanya Form Footer akan berisi keterangan-keterangan lain misalnya nama pembuat, nama pengisi data dan lain-lain.



Gambar 11.46. Bagian-bagian suatu form.

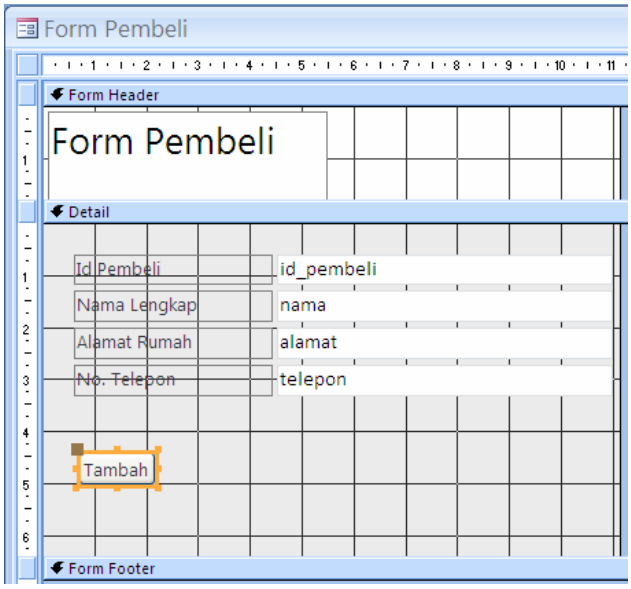
3. Microsoft Access menggunakan control-control yang sama seperti pada bahasa pemrograman berbasis GUI (lihat kembali Bab 13). Pada form kita akan menjumpai control Label, Text Box, Combo Box, Command Button dan lain-lain.

4. Modifikasi pertama yang kita lakukan adalah memperbaiki tampilan pada nama-nama fields. Caranya dengan click pada Label yang ingin kita perbaiki. Kemudian ganti atau ubah dengan cara mengetikkan nama yang baru. Setelah semua selesai tampilan Form Pembeli akan tampak seperti pada Gambar 11.47. Perhatikan perbedaan Label fields pada Gambar 11.45 dan Gambar 11.47.



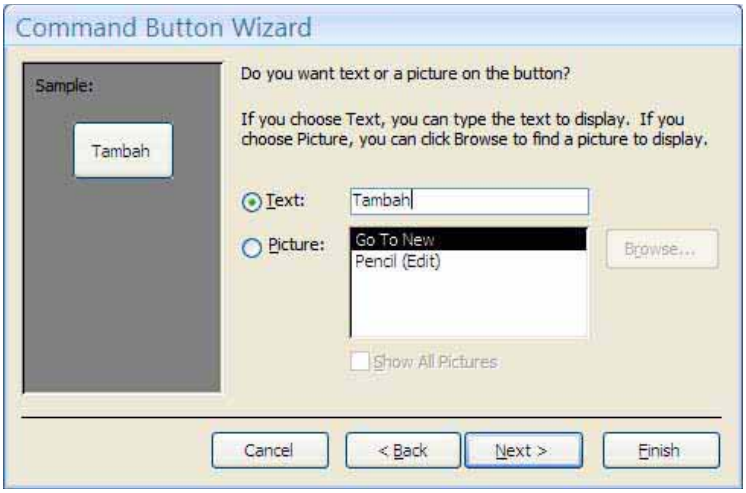
Gambar 11.47. Perubahan pada Label fields pada Form Pembeli.

5. Modifikasi berikutnya yang akan kita lakukan adalah menambahkan tombol-tombol untuk memudahkan pengguna dalam mengisi data. Untuk melakukan ini kita harus mengatur form menjadi seperti pada Gambar 11.48. Kemudian kita tambahkan tombol dengan cara click pada icon Command Button pada Toolbar, lalu letakan pada Form. Jendela Command Buttom Wizards akan muncul seperti pada Gambar 11.49. Pada *Categories* pilih *Record Operations* dan pada *Actions* pilih *Add New Record*. Click Next untuk melanjutkan. Pada jendela berikutnya (Gambar 11.50) pilih Text dan ketikkan *Tambah* lalu klik Next. Pada jendela berikutnya (Gambar 11.51) ketikkan nama untuk command button ini, kita berikan nama misalnya *cmdTambah* lalu klik Finish.

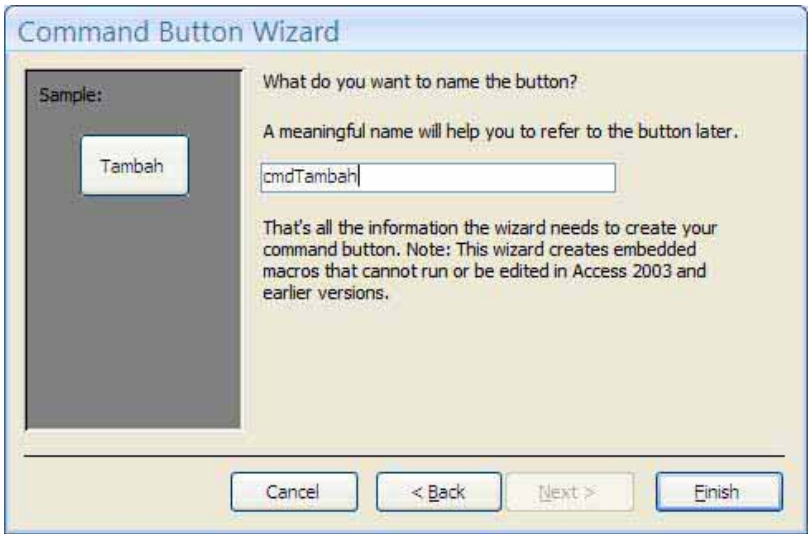


Gambar 11.48. Modifikasi tampilan form.

Gambar 11.49. Mendefinisikan aksi untuk suatu Command Button.

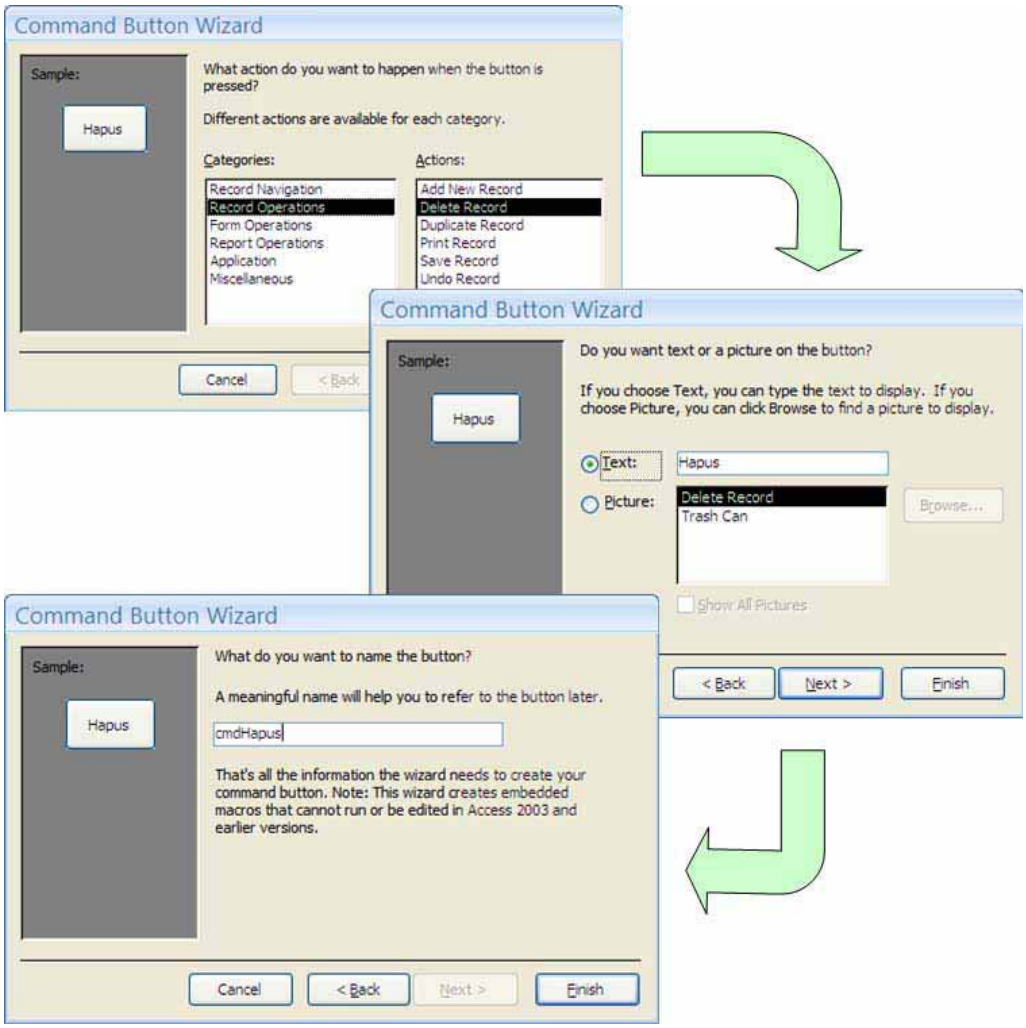


Gambar 11.50. Mendefinisikan teks pada Command Button.



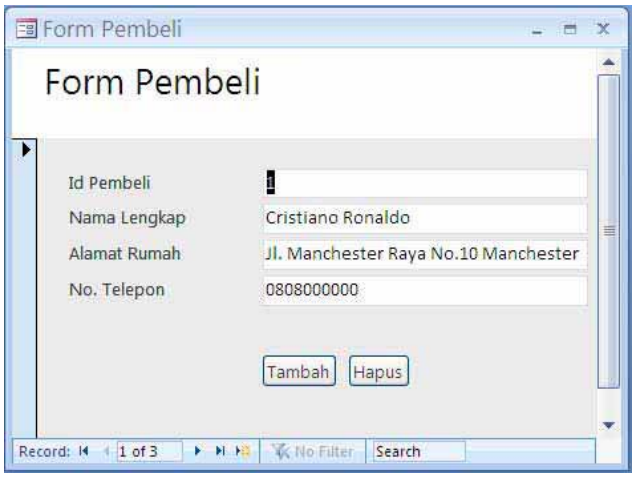
Gambar 11.51. Mendefinisikan nama Command Button.

6. Kita tambahkan tombol yang berfungsi menghapus data. Cara pembuatannya persis seperti di atas hanya saja pada jendela Command Buttom Wizards, Categories dipilih *Record Operations* dan *Actions* yang dipilih adalah *Delete Record* (Gambar 11.52). Tentukan teks dan nama command button yang sesuai.



Gambar 11.52. Mendefinisikan nama Command Button.

7. Tampilan akhir dari form setelah dimodifikasi akan tampak seperti pada Gambar 11.53.



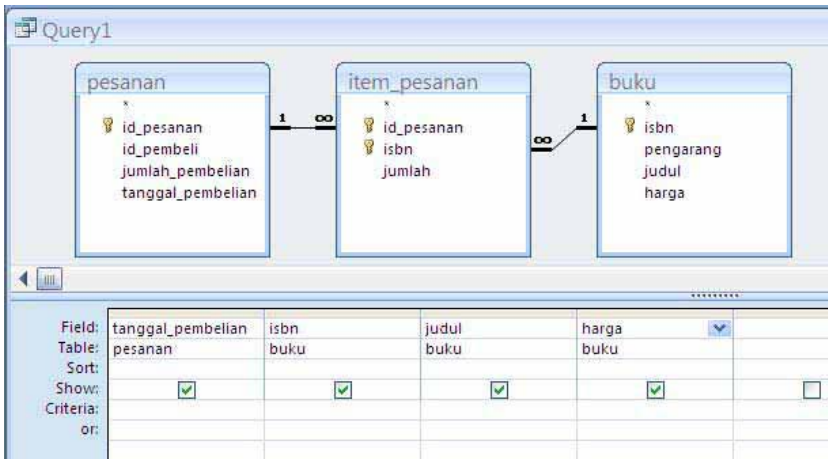
Gambar 11.53. Hasil modifikasi Form Pembelian.

Kita dapat menambahkan tombol-tombol lain atau control-control lain dengan cara yang kurang lebih sama.

11.5. REPORT

Report, seperti halnya form, digunakan untuk merepresentasikan hasil olahan data menjadi informasi yang siap di cetak di lembaran kertas. Kita dapat saja mencetak langsung dari tabel database namun hasil cetakannya tidak seperti laporan yang diinginkan. Cara yang terbaik adalah dengan membuat model laporan dengan fasilitas Report. Report dapat dibuat dengan dua cara yaitu manual dan wizards. Seperti halnya pada form, wizard memberikan kemudahan dalam pembuatan report, karena semuanya sudah diatur otomatis. Berikut ini kita akan membuat laporan penjualan harian. Laporan ini berisi tanggal transaksi, ISBN dan judul buku yang terjual, harga masing-masing buku dan total nilai penjualan per hari/tanggal.

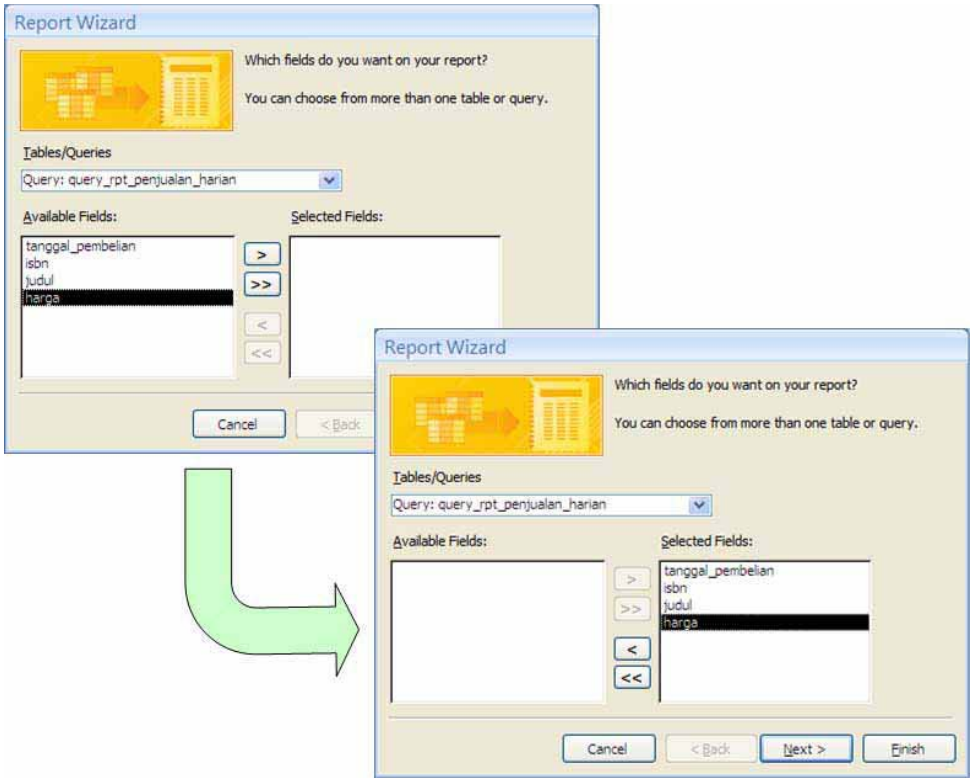
1. Untuk membuat report ini kita tidak bias langsung dari tabel, karena report ini berisi gabungan dari beberapa tabel yang ada. Sehingga kita harus membuat *query* sebagai sumber data untuk laporan. Cara membuat *query* sama dengan yang kita lakukan sebelumnya. Perhatikan Gambar 11.54 berikut ini. Pada Gambar tersebut kita memilih tabel pesanan, item_pesanan dan buku. Kemudian kita memilih tanggal_pembelian, isbn, judul, dan harga pada bagian Field. Kemudian kita simpan dengan nama *query*_rpt_penjualan_harian (atau dengan nama yang lain).



Gambar 11.54. Jendela *query* untuk sumber report.

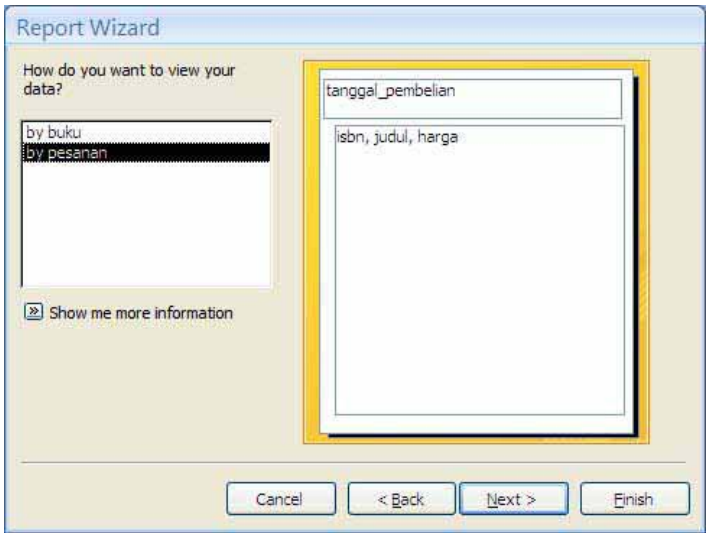
2. Pada jendela Database pilih object Report dan double click pada Create Report by using Wizard. Jendela Report Wizard akan terbuka dan pada bagian Tabels/Queries pilih *query* yang telah kita buat pada bagian 1 (lihat Gambar 11.55). Kemudian click tombol >> sehingga semua field yang ada pada bagian Available Fields berpindah ke Selected Fields (lihat Gambar 11.56)

Gambar 11.55. Pemilihan *query* sebagai sumber data laporan.



Gambar 11.56. Pemilihan fields yang terlibat.

3. Click Next untuk membuka jendela berikutnya (Gambar 11.57). Pada jendela ini kita menentukan dasar tampilan laporan, apakah berdasarkan buku atau pesanan. Karena kita akan membuat laporan harian maka tampilan berdasarkan pesanan yang kita pilih (ingat, field tanggal_pembelian ada pada tabel pesanan).
4. Click Next untuk melanjutkan dengan jendela berikutnya (Gambar 11.58). Jendela ini digunakan untuk mengelompokkan (grouping) data pada field yang sama. Pada contoh kali ini kita tidak melakukan grouping sehingga kita tidak perlu mengatur apa-apa pada jendela ini. Kita dapat langsung click Next untuk melanjutkan pada jendela berikutnya.

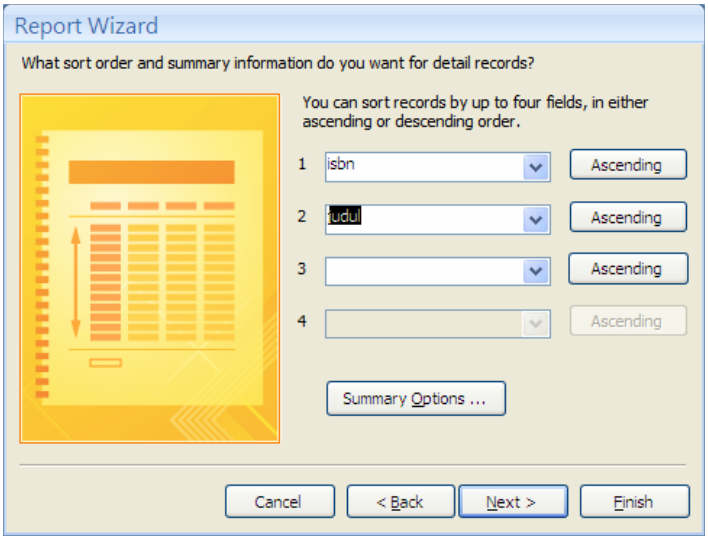


Gambar 11.57. Jendela untuk menentukan dasar tampilan report.

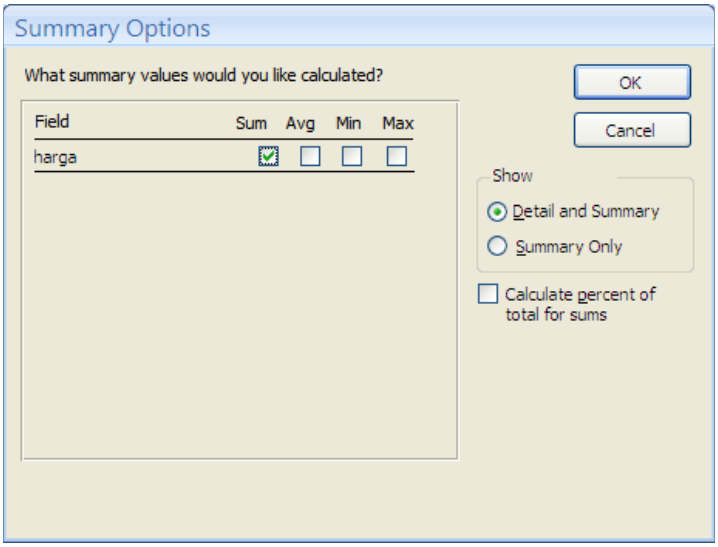
Gambar 11.58. Jendela untuk menentukan grouping data.

5. Pada Gambar 11.59, kita dapat memilih melakukan pengurutan data atau tidak. Pada bagian ini kita akan mengurutkan berdasarkan nomor ISBN dan kemudian berdasarkan judul buku. Pada Combo Box no 1 kita pilih isbn dan pada Combo Box no 2 kita pilih judul. Selain itu pada bagian ini kita juga mengatur apakah kita membuat ringkasan laporan atau tidak. Click pada Summary Options untuk membuka jendela pengaturan ringkasan (Gambar 11.60). Pada field harga kita check pada kotak di bawah Sum. Maksud dari bagian ini adalah kita akan

menampilkan jumlah pembelian atau transaksi tiap harinya dan total transaksi. Jika sudah selesai kita dapat menutup jendela Summary Option dengan meng-click tombol OK. Kemudian kita click Next untuk melanjutkan dengan jendela berikutnya.



Gambar 11.59. Jendela untuk menentukan urutan data.

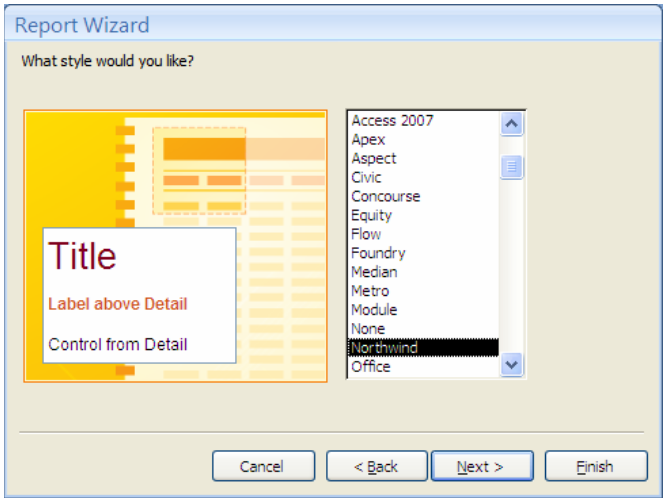


Gambar 11.60. Jendela untuk mengatur tampilan ringkasan.

6. Gambar 11.61 menunjukkan jendela untuk mengatur lay-out dari laporan. Kita dapat mencoba-coba lay-out mana yang sesuai. Pada

contoh ini kita akan mencoba dengan lay-out Block dan Orientation Landscape. Click Next jika sudah selesai dan jendela untuk mengatur style dari laporan (Gambar 11.62). Pada bagian ini kita dapat memilih sesuai keinginan kita. Setelah selesai click Next untuk membuka jendela terakhir dari Report Wizard. Pada bagian Text Box yang tersedia kita tentukan judul dari laporan yang telah kita buat. Click tombol Finish untuk mengakhiri proses pembuatan laporan. Hasil pembuatan laporan akan ditampilkan seperti pada Gambar 11.61.

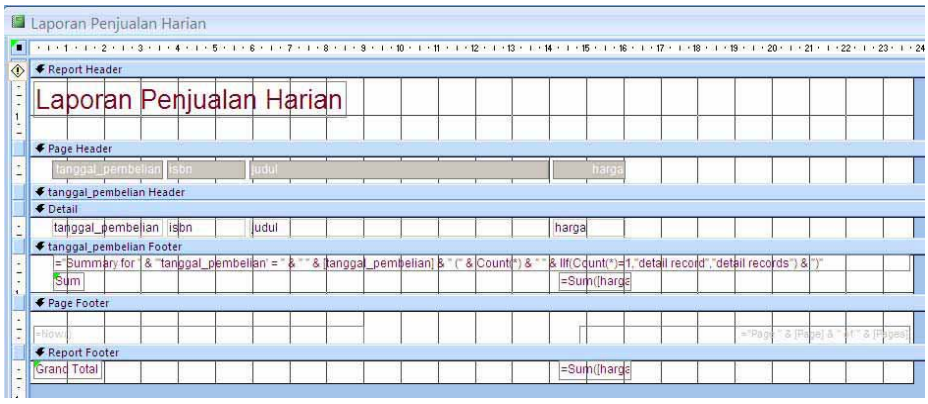
Gambar 11.61. Jendela untuk mengatur lay-out dan orientation.



Gambar 11.62. Jendela untuk mengatur style laporan.

Gambar 11.63. Hasil pembuatan laporan menggunakan Wizard.

7. Pada Gambar 11.63, kita melihat laporan yang tidak terlalu bagus bila dicetak. Judul kolom masih menggunakan nama field pada tabel. Selain itu kata-kata Summary for tanggal pembelian dan seterusnya, agak mengganggu tampilan laporan. Untuk memperbaiki tampilan laporan, click kanan pada nama report di jendela object Report kemudian pilih Design View. Jendela seperti pada Gambar 11.64 akan terbuka.

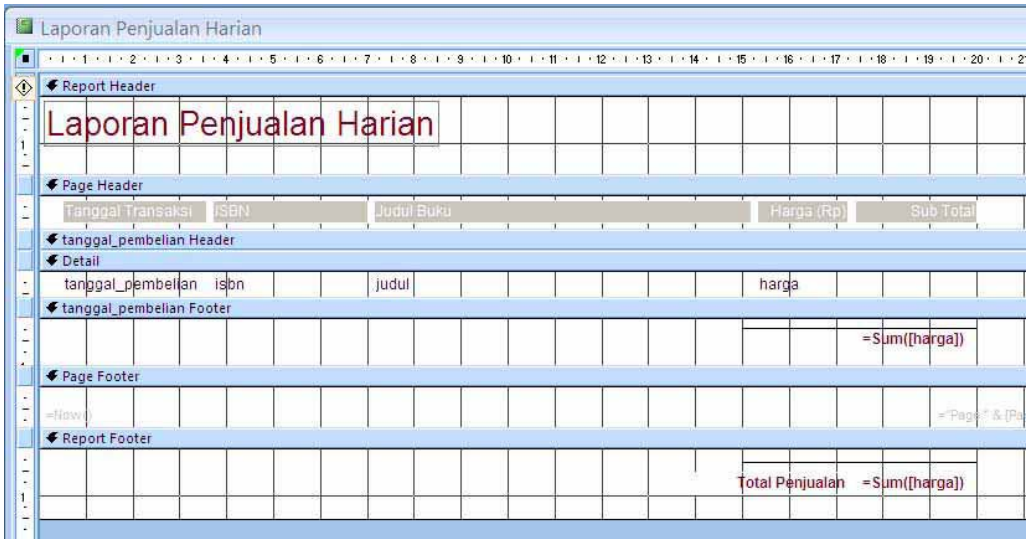


Gambar 11.64. Laporan dalam mode Design View.

8. Perbaikan pertama yang akan kita lakukan adalah pada judul kolom. Kita akan ganti menjadi seperti pada Gambar 11.65. Untuk mengganti judul kolom, double-click pada judul kolom yang ingin diganti kemudian

ketikan nama yang baru. Kita juga dapat mengatur lebar kolom dengan cara click pada bagian paling atas kolom untuk memilih seluruh kolom kemudian letakkan kursor pada samping kanan kolom dan geser untuk memperlebar kolom (sama persis dengan mengatur lebar tabel pada Microsoft Word).

9. Perbaikan yang kedua adalah kita menghilangkan kata-kata Summary for tanggal pembelian dan seterusnya yang tampak pada Gambar 11.64. Caranya dengan click pada bagian tersebut kemudian tekan tombol Del pada keyboard. Selain itu kita hilangkan kata Sum dan kita ganti kata Grand Total dengan Total Penjualan. Penambahan garis juga dapat dilakukan dengan memilih *control line* pada Toolbar Report, kemudian diletakkan pada posisi yang diinginkan. Tampilan akhir pada Design View akan tampak seperti pada Gambar 11.65. Sedangkan hasil Print Preview dari laporan akan tampak seperti pada Gambar 11.66.



Gambar 11.65. Design laporan setelah dilakukan perbaikan.

Gambar 11.66. Print Preview laporan setelah perbaikan.

11.6. RINGKASAN

- Ada enam obyek penting Microsoft Access, yaitu Table, Queries, Forms, Reports, Macros dan Modules.
- *Tabel* dalam Microsoft Access terdiri dari record (baris) dan field (kolom). Tabel dapat dibuat dengan mendefinisikan field-field yang dibutuhkan dilengkapi dengan tipe data, *domain* dan penentuan *primary key* nya.
- Query atau permintaan data dapat menampilkan data dari satu tabel atau beberapa tabel yang saling berhubungan. *Query* dalam Microsoft Acces dapat dilakukan dengan menggunakan fasilitas GUI.
- *Form* adalah salah satu obyek basis data dalam Microsoft Access yang digunakan sebagai antar muka bagi pengguna untuk memasukkan data atau menampilkan data. Form dapat dibuat melalui metode *Wizard* maupun manual dengan menggunakan Design View.
- *Report* digunakan untuk merepresentasikan hasil olahan data menjadi informasi yang siap di cetak di lembaran kertas. *Report* juga dapat dibuat dengan cara manual maupun dengan fasilitas *Wizard*.

11.7. SOAL-SOAL LATIHAN

1. Perhatikan tabel-tabel yang ada pada contoh di atas. Tambahkan satu buah tabel dengan nama penerbit. Atribut-atributnya adalah ID Penerbit, nama penerbit, alamat, kota dan contact person. Tentukan sendiri tipe data dan lebar data untuk tabel tersebut. Isilah tabel buku tersebut dengan penerbit-penerbit yang anda ketahui (coba lihat buku-buku diperpustakaan kalian dan cermati informasi tentang penerbit).
2. Pada kondisi nyata, penerbit akan berhubungan dengan buku. Sebuah penerbit dapat menerbitkan lebih dari satu buku. Tetapi satu buku hanya bisa diterbitkan oleh satu penerbit. Buatlah hubungan antar tabel baru antara tabel buku dengan penerbit. Jika perlu lakukan perubahan pada tabel buku.
3. Buatlah *query-query* berikut ini dan cermati hasil yang dperoleh.
    a. Tampilkan semua data penerbit.
    b. Tampilkan data penerbit yang berasal dari kota Bandung.
    c. Tampilkan data penerbit yang mempunyai buku berharga di atas Rp. 150.000,-.
    d. Tampilkan nama penerbit yang bukunya dibeli oleh Cristiano Ronaldo.
    e. Tampilkan buku yang dibeli oleh Cristiano Ronaldo dan Wayne Rooney.
4. Buatlah form untuk memasukkan data-data penerbit.
5. Buatlah report untuk menampilkan daftar penerbit dan buku yang diterbitkannya.

12

BAB 12 BASIS DATA BERBASIS SQL

Gambar 12.1. Perangkat komputer server.

Gambar 12.1 di samping ini adalah gambar sekumpulan komputer *server* yang biasanya ada pada perusahaan penyedia jasa internet atau pada bagian teknologi informasi perusahaan-perusahaan besar. Salah satu fungsi peralatan ini adalah sebagai pusat data yang dibutuhkan penggunanya. Peran sebagai pusat data ini membutuhkan DBMS dan aplikasi basis data yang kuat. Perangkat DBMS seperti Microsoft SQL Server, Oracle, MySQL, PostgreSQL biasanya menjadi pilihan. Fungsi-fungsi lanjut yang disediakan oleh DBMS tersebut sangat membantu kecepatan kerja, keamanan, dan keakuratan data yang disampaikan.

Bab ini membahas tiga standar kompetensi yaitu melakukan pemrograman data deskripsi (SQL) tingkat dasar, mengoperasikan bahasa pemrograman data deskripsi (SQL) tingkat lanjut dan membuat program basis data dengan Microsoft SQL Server. Kedua standar kompetensi ini disatukan karena kedekatan materi bahasan. Susunan sub bab tidak langsung mengacu pada kompetensi dasar, tapi lebih pada kecocokan materi. Ringkasan akan disampaikan di akhir bab kemudian dilanjutkan dengan soal-soal latihan. Sebelum kalian mempelajari bab ini buka kembali bab-bab tentang pemecahan masalah, sistem operasi, algoritma lanjutan, dasar-dasar basis data, dan aplikasi basis data berbasis Microsoft Access pada bab-bab sebelumnya.

TUJUAN

Setelah mempelajari bab ini diharapkan pembaca akan mampu :

- Mempersiapkan dan mengoperasikan perangkat lunak SQL
- Mengenali menu aplikasi SQL
- Membuat dan mengisi tabel
- Mengoperasikan tabel dan *view*
- Menggunakan T-SQL
- Menggunakan fitur-fitur lanjutan Microsoft SQL Server seperti procedures, fungsi, trigger dan XML support
- Menerapkan administrasi Microsoft SQL Server

12.1. SEKILAS TENTANG SQL

Dalam DBMS biasanya tersedia paket bahasa yang digunakan untuk mengorganisasi basis data yang ada, yaitu *Data Definition Language* (DDL) dan *Data Manipulation Language* (DML).

12.1.1. Data Definition Language (DDL)

Data Definition Language (DDL) adalah satu paket bahasa DBMS yang berguna untuk melakukan spesifikasi terhadap skema basis data. Hasil kompilasi dari DDL adalah satu set tabel yang disimpan dalam file khusus yang disebut *Data Directory/Dictionary*. Secara umum perintah perintah dalam DDL berhubungan dengan operasi-operasi dasar seperti membuat basis data baru, menghapus basis data, membuat tabel baru, menghapus tabel, membuat indeks, mengubah struktur tabel. Contoh perintah DDL misalnya, *Create Table, Create Index, Alter, dan Drop Database*.

12.1.2. Data Manipulation Language

Data Manipulation Language (DML) adalah satu paket DBMS yang memperbolehkan pemakai untuk mengakses atau memanipulasi data sebagaimana yang telah diorganisasikan sebelumnya dalam model data yang tepat. Dengan DML dapat dilakukan kegiatan :

- Mengambil informasi yang tersimpan dalam basis data.
- Menyisipkan informasi baru dalam basis data.
- Menghapus informasi dari tabel.

Terdapat dua tipe DML yaitu prosedural dan non prosedural. Prosedural DML membutuhkan pemakai untuk menspesifikasikan data apa yang dibutuhkan dan bagaimana cara mendapatkannya, sedang non prosedural DML membutuhkan pemakai untuk menspesifikasikan data apa yang dibutuhkan tanpa tahu bagaimana cara mendapatkannya.

SQL merupakan kependekan dari *Structured Query Language* (Bahasa Query Terstruktur). SQL lebih dekat dengan DML dari pada DDL. Namun tidak

berarti SQL tidak menyediakan perintah DDL. SQL lebih menekankan pada aspek pencarian dari dalam tabel. Aspek pencarian ini sedemikian penting karena di sinilah sebenarnya inti dari segala upaya kita melakukan pengelolaan data. Data dalam basis data diorganisasi sedemikian rupa dengan tujuan untuk memudahkan pencarian di kemudian hari.

Sebagai sebuah bahasa, SQL telah distandarisasi dan mengalami beberapa perubahan atau penyempurnaan. SQL muncul pertama kali pada tahun 1970 dengan nama Sequel (nama yang masih sering digunakan hingga saat ini). Standarisasi yang pertama dibuat pada tahun 1986 oleh ANSI (*American National Standards Institute*) dan ISO (*International Standard Organization*), yang disebut SQL-86. Pada tahun 1989 SQL-86 diperbaharui menjadi SQL-89. Standar terakhir yang dibuat adalah SQL-92.

Pernyataan-pernyataan SQL digunakan untuk melakukan beberapa tugas seperti : update data pada basis data, atau menampilkan data dari basis data. Beberapa *software* RDBMS yang dapat menggunakan SQL, seperti : Oracle, Sybase, Microsoft SQL Server, MySQL, Microsoft Access, Ingres, dsb. Setiap *software* basis data mempunyai mungkin bahasa perintah / sintaks yang berbeda, namun pada prinsipnya mempunyai arti dan fungsi yang sama.

Perintah utama dalam SQL adalah **select**. Struktur utama perintah adalah sebagai berikut:

```
Select <kolom>
From <table>
Where <kondisi>
```

Kita akan menggunakan perintah-perintah ini pada bagian-bagian berikut.

12.2. MEMPERSIAPKAN PERANGKAT LUNAK BERBASIS SQL

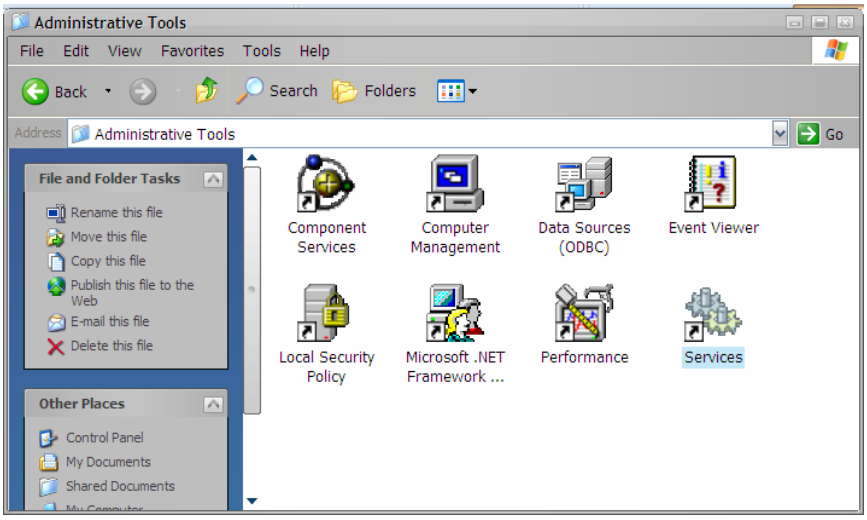
Seperti disebutkan di bagian awal bab, perangkat lunak yang akan kita gunakan adalan Microsoft SQL Server. Pada kesempatan ini versi yang dipakai adalah versi 2005 Developer Edition. Penjelasan sekilas tentang Microsoft SQL Server dapat dilihat pada bab 10.

12.2.1.Kebutuhan Operasi

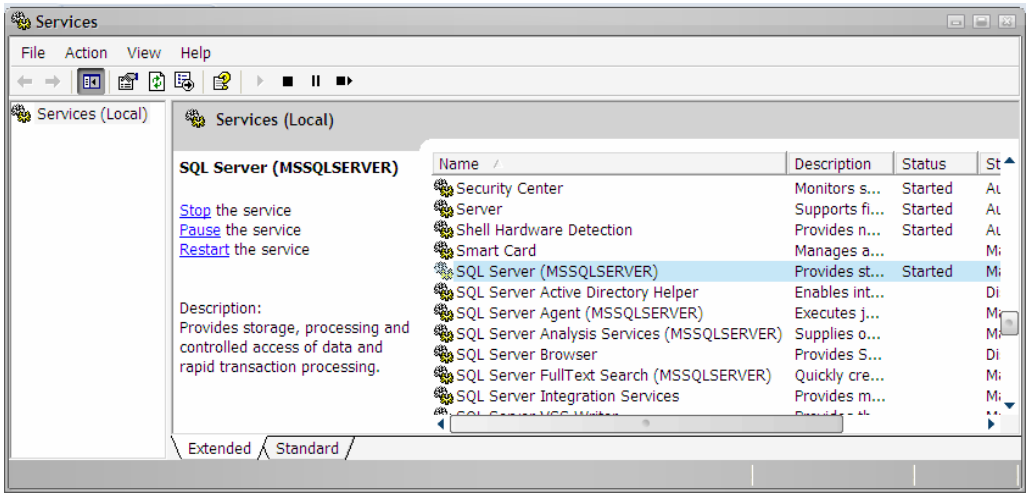
Microsoft SQL Server adalah perangkat lunak yang berjalan pada platform Microsoft Windows, sehingga kebutuhan utama dari perangkat lunak ini adalah tersedianya komputer yang telah terinstal sistem operasi Windows, terutama versi Server. Hal ini karena Windows versi *Server* memberikan integrasi yang lebih baik. Selain itu dari sisi keamanan dan alokasi kerja model *client-server*, versi *server* memberikan unjuk kerja yang lebih baik.

Microsoft SQL Server adalah perangkat lunak yang harus diaktifkan sebagai *service* pada Windows. Jika belum dijalankan maka kita tidak akan dapat menggunakannya. *Default*-nya, ketika Microsoft SQL Server diinstal maka secara otomatis, *service* ini akan dibuat dan dijalankan setiap kali Windows

dihidupkan. Namun untuk memastikan kalian dapat memeriksanya dengan membuka *Control Panel*, kemudian pilih *Administrative Tool*. Pada jendela *Administrative Tool* (Gambar 12.2) klik ganda pada ikon *Service* sehingga terbuka jendela seperti pada Gambar 12.3.



Gambar 12.2. Jendela *Administrative Tool*.

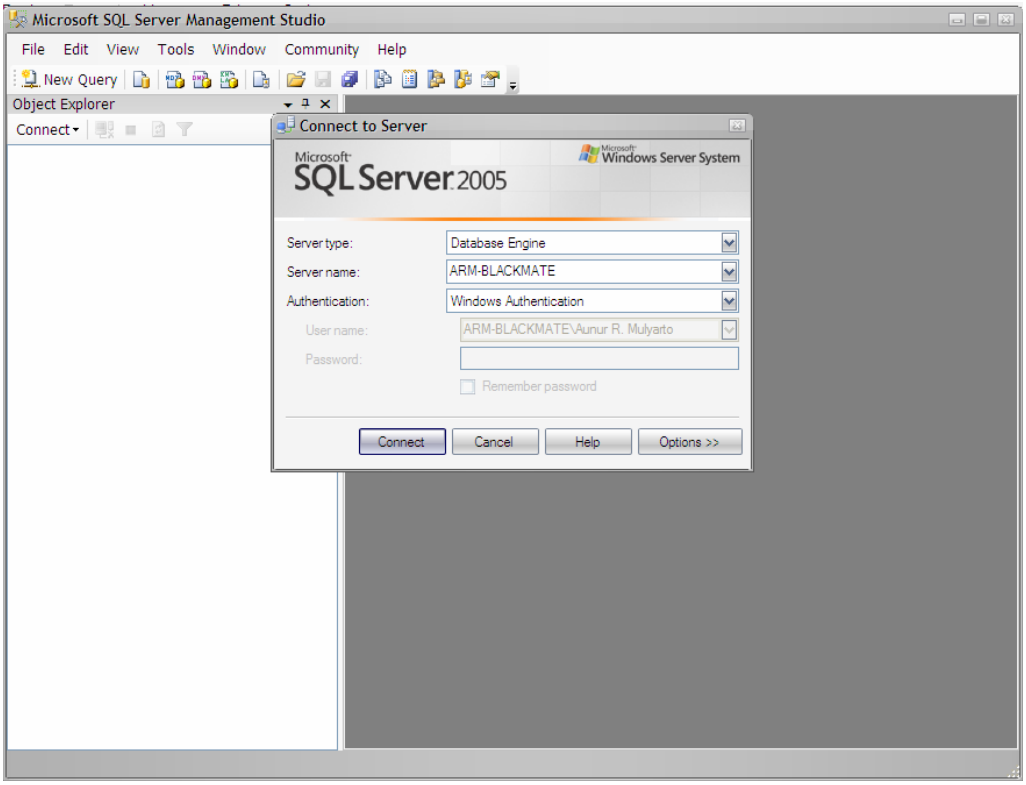


Gambar 12.3. Jendela *Services*.

Pada jendela *Service* carilah nama service SQL Server kemudian periksalah di kolom status, apakah sudah started atau belum. Jika belum jalankan (*start*) *service* ini.

12.2.2. Menjalankan Perangkat Lunak Basis Data Berbasis SQL

Setelah terinstal maka kita dapat menggunakan Microsoft SQL Server dengan memanggil *GUI client* dari Microsoft SQL Server yang disebut sebagai *SQL Server Management Studio*. Ketika pertama kali dipanggil maka tampilan akan tampak seperti pada Gambar 12.4.



Gambar 12.4. Tampilan autentikasi SQL Server Management Studio

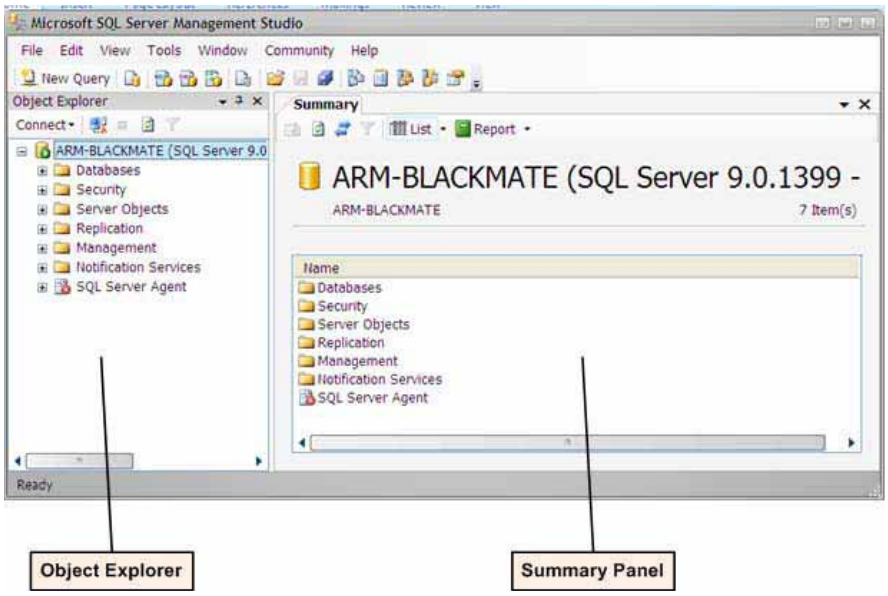
Setiap kali kita membuka SQL Server Management Studio, kita akan diperiksa apakah kita berhak menggunakan SQL Server atau tidak. Jika SQL Server dijalankan pada mesin lokal dan menggunakan *Windows Authentication* maka kita dapat langsung menekan tombol *Connect* untuk masuk ke lingkungan SQL Server. Tetapi bila SQL Server dijalankan dari komputer lain (server) kita akan diminta memasukkan *user name* dan *password* yang telah didaftarkan.

Ada informasi penting dari Gambar 12.4 di atas yang kalian perlu ketahui. Pada gambar tersebut terlihat bahwa kita akan mengakses tipe server *Database Engine*, yaitu server yang berhubungan dengan masalah aplikasi basis data. SQL Server menyediakan beberapa tipe server yang dapat diakses. Namun tidak dijelaskan pada buku ini. Selain itu nama server yang akan diakses adalah ARM-BLACKMATE. Hal ini karena SQL Server diinstal pada komputer dengan nama

ARM-BLACKMATE sehingga kita perlu nama server sama dengan nama dimana SQL Server berjalan.

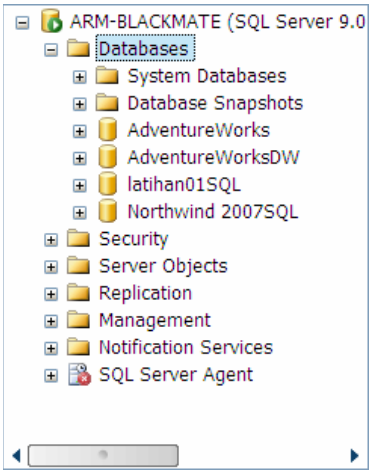
12.3. MENU / FITUR UTAMA

Setelah kita berhasil masuk ke dalam lingkungan SQL Server maka kita akan disuguhi tampilan awal dari SQL Server Management Studio seperti pada Gambar 12.5.



Gambar 12.5. Tampilan awal SQL Server Management Studio.

Ada dua bagian penting yang ada pada bagian awal ini yaitu *Object Explorer* dan *Summary Panel*. *Object Explorer* adalah tempat kita melihat obyek-obyek apa saja yang ada di dalam SQL Server. Pada gambar di atas kalian bisa melihat obyek-obyek yang tersedia. Untuk sementara yang paling penting adalah Databases. Sedangkan *Summary panel* merupakan tempat keterangan atau ringkasan yang ada pada *Object Explorer*. Coba klik pada salah satu obyek di *Object Explorer*, maka isi *Summary panel* juga akan berubah sesuai dengan obyek yang kita pilih.

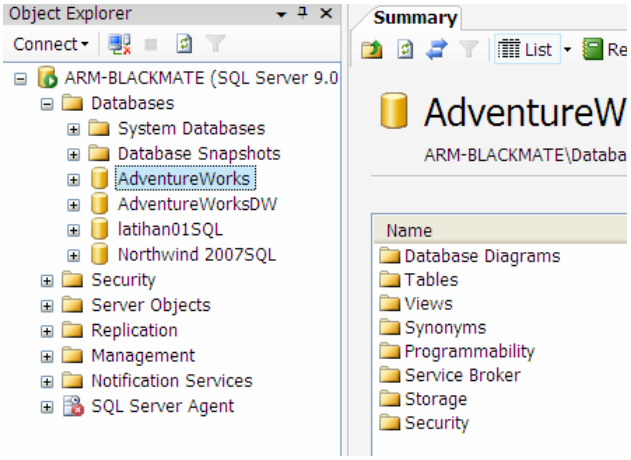


Gambar 12.6. Obyek Databases.

Kita akan fokus pada obyek *Databases*. Coba klik tanda + (atau tanda node) di depan obyek *Databases*, sehingga akan tampak tampilan seperti pada Gambar 12.6 di samping ini. Pada

Gambar tersebut tampak ada empat basis data yang sudah ada pada SQL Server, yaitu AdventureWorks, AdventureWorksDW, latihan01SQL dan Northwind2007SQL. Keempat basis data ini termasuk *user object* karena merupakan buatan pengguna. SQL Server juga membuat sendiri basis data ketika proses instalasi berlangsung. Basis data ini biasa disebut sebagai *system object*. Biasanya ada beberapa system object yang tersedia yaitu Master, Model, MSDB, Resource, TempDB, dan Distribution. Basis data ini tidak boleh dihapus karena akan membuat sistem menjadi tidak bekerja normal. Untuk memeriksanya, coba klik node di depan *System Databases*.

Sementara ini kita tidak akan berhubungan dengan System Databases. Kita akan melihat pada user object. Klik pada salah satu basis data maka tampilan pada *Summary Panel* akan berubah. Perhatikan Gambar 12.7. Pada gambar tersebut tampak bahwa basis data AdventureWorks berisi berbagai komponen antara lain *Database Diagrams, Tables, Views* dan yang lainnya. Setiap basis data yang kita buat di SQL server akan punya komponen-komponen ini secara default. Meskipun kita tidak membuat *Database Diagrams* misalnya, komponen *Database Diagrams* tetap disediakan.

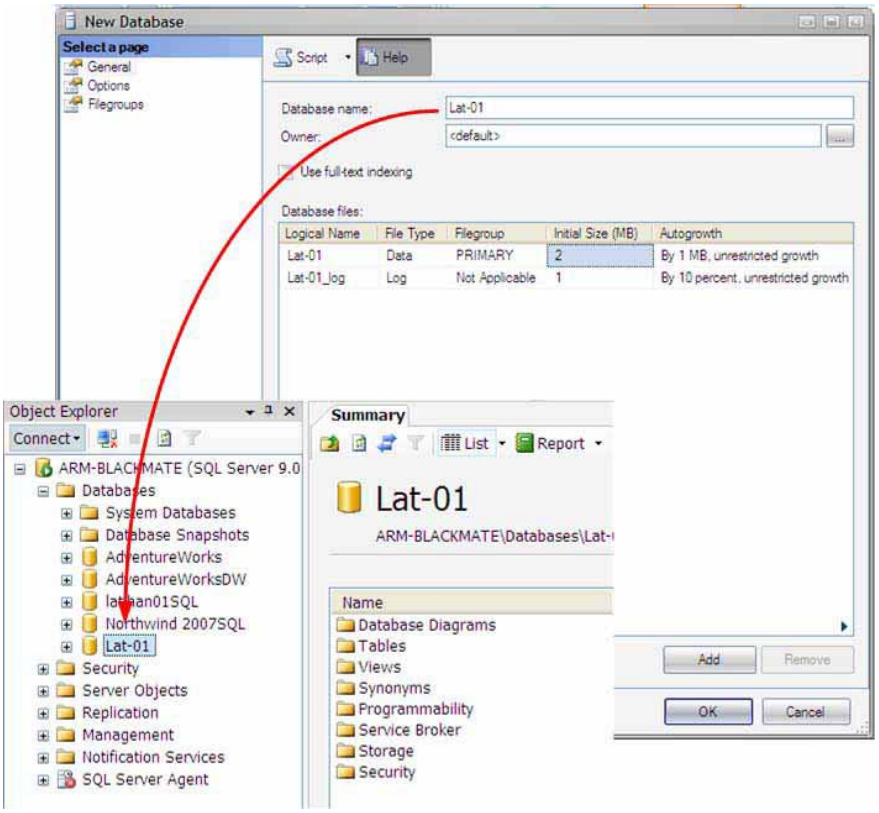


Gambar 12.7. Isi dari basis data pada SQL Server.

12.4. PEMBUATAN DAN PENGISIAN TABEL

12.4.1. Pembuatan Basis Data

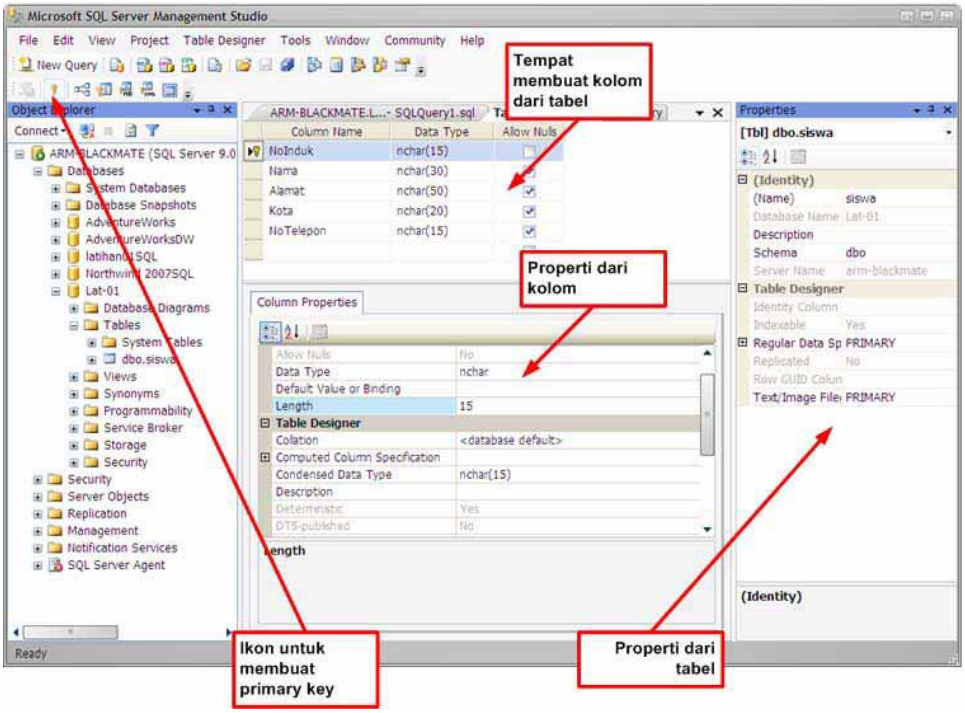
Sebelum kita membuat tabel, kita harus membuat lebih dahulu basis datanya. Untuk membuat basis data baru, klik kanan pada *Databases* dan pilih *New Database*. Jendela untuk membuat basis data baru akan terbuka (Gambar 12.8). Pada *textbox* di depan *Database name*, ketikkan nama basis data yang akan dibuat. Pada contoh ini namanya adalah Lat-01. Kemudian klik OK. Perhatikan pada *Object Explorer*, basis data baru sudah terbentuk dan memiliki komponen-komponen yang sama dengan basis data lainnya.



Gambar 12.8. Mendefinisikan basis data baru.

12.4.2. Pembuatan Tabel

Setelah basis data terbentuk kita baru bisa membuat tabel. Klik kanan pada komponen *Tables* di basis data Lat-01 dan pilih *New Table...* Bagian *Summary panel* akan berubah menjadi seperti pada Gambar 12.9. Kalian bisa mulai memasukkan atribut atau kolom dari tabel yang akan dibuat. Pembuatan tabel ini sama persis dengan apa yang telah kalian lakukan dengan Microsoft Access. Penentuan tipe data, lebar data dan *primary key* juga tidak jauh berbeda. Yang agak berbeda adalah SQL Server menyediakan tipe data yang lebih banyak daripada Microsoft Access. Untuk menentukan *primary key*, pilih atribut kemudian tekan tanda kunci pada *toolbar*.



Gambar 12.9. Pembuatan tabel.

Pada gambar di atas, kita membuat tabel dengan 5 kolom yaitu NoInduk, Nama, Alamat, Kota, NoTelepon. Masing-masing bertipe *nchar* dengan lebar data yang bervariasi. Untuk menentukan lebar data, lihatlah pada bagian property dari kolom. Setelah selesai simpan tabel dengan nama tertentu. Pada contoh di atas tabel disimpan dengan nama siswa. SQL server akan memberikan awalan nama (*prefix*) dbo secara default. Tetapi ini bisa kita rubah. Jadi nama tabel yang kita buat di atas akan menjadi dbo.siswa.

Hapuslah tabel dbo.siswa dengan cara klik kanan lalu pilih *Delete*. Sekarang buatlah tabel-tabel berikut ini dengan cara seperti di atas.

Tabel 12.1. Tabel, kolom, tipe data yang akan dibuat.

Nama Tabel	Kolom	Tipe dan lebar Data	Keterangan
Siswa	- NoInduk - NamaSiswa - Alamat - Kota - Telepon - IdProgram	Nchar (10) Nchar (20) Nchar (30) Nchar (20) Nchar (15) Smallint	
Guru	- NIP	Nchar (15)	NIP sebagai

	- NamaGuru - Alamat - Kota - Telepon	Nchar (20) Nchar (30) Nchar (20) Nchar (15)	primary key
Program	- IdProgram - NamaProgram - Deskripsi	Smallint Nchar (20) Nchar (100)	IdProgam sebagai primary key
Bidang	- IdBidang - NamaBidang - Deskrips	Smallint Nchar (20)	IdBidang sebagai primary key
Guru_Program	- NIP - IdProgram	Nchar (15) Smallint	Kedua atribut ini bersama-sama sebagai primary key
Guru_Bidang	- NIP - IdBidang	Nchar (15) Smallint	Kedua atribut ini bersama-sama sebagai primary key

12.4.3.Pengisian Data pada Tabel

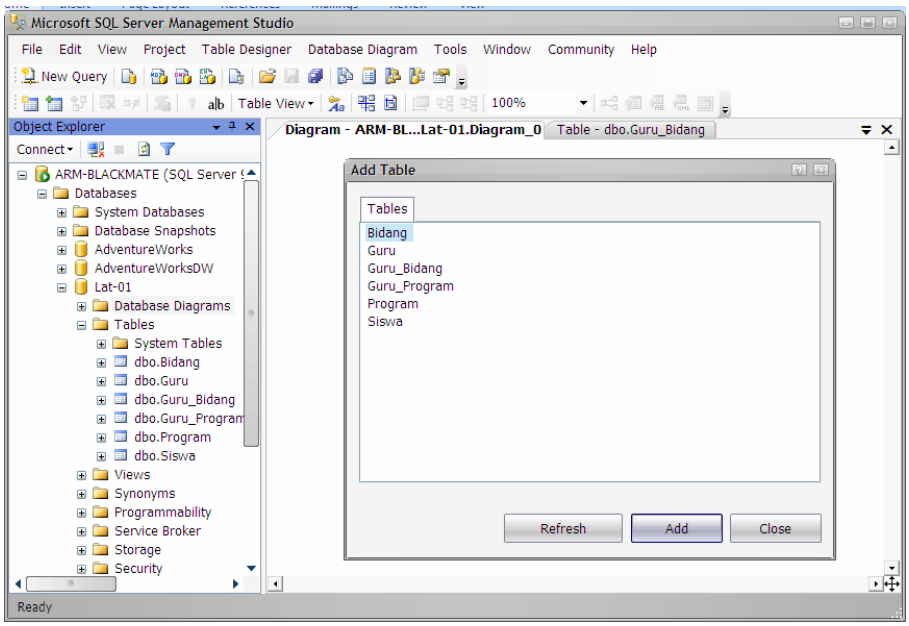
Setelah semua tabel terbentuk, kita dapat mengisi tabel dengan data-data yang kita inginkan. Dengan menggunakan GUI *SQL Server Management Studio* kita dapat mengisi data seperti halnya pada Access. Klik kanan pada nama tabel kemudian pilih *Open Table*. Jendela baru pada *Summary panel* akan terbuka dan siap untuk diisi datanya. Gambar 12.10 menunjukkan bagaimana kita mengisi data pada sebuah tabel. Setiap kali kalian mengisi untuk satu baris harus diikuti dengan menekan *Enter*. Jika tidak maka data tidak tersimpan. Cobalah isi tabel-tabel yang lain.

Gambar 12.10. Pengisian tabel.

12.5. OPERASI PADA TABEL DAN VIEW

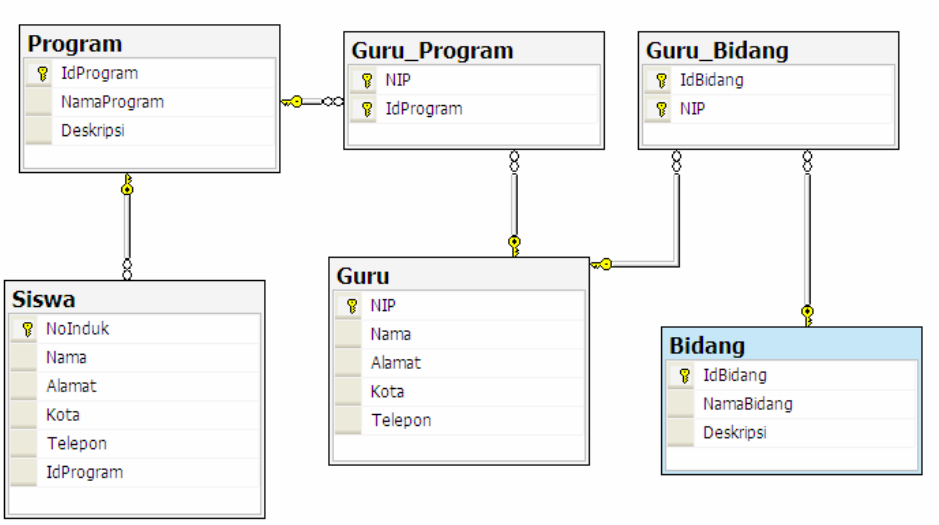
12.5.1.Relasi Antar Tabel

Tabel-tabel yang telah kalian buat dan isi di atas masih merupakan tabel yang berdiri sendiri tidak ada hubungan antar tabel. Padahal kalau dilihat dari tabel-tabel yang ada kita bisa membuat model hubungannya (lihat bab 10 dan 11 tentang relasi antar tabel). Tabel siswa berhubungan dengan tabel program, tabel guru berhubungan dengan tabel bidang dan tabel program. Untuk membuat hubungan antar tabel ini kita bisa menggunakan komponen Database Diagrams. Klik kanan pada Database Diagrams kemudian pilih New Database Diagram. Jendela baru akan terbuka seperti tampak pada Gambar 12.11.



Gambar 12.11. Jendela untuk menambah tabel yang berhubungan.

Pada jendela *Add Table* klik *Add* untuk memilih tabel yang akan kita relasikan. Pada contoh ini kita akan pilih semua tabel karena semua saling terkait. Klik *Add* beberapa kali sampai tidak ada nama tabel di jendela *Add Table*. Setelah semua tabel terpilih buat relasi seperti tampak pada Gambar 12.12. Cara menghubungkan tabel satu dengan tabel yang lain sama persis dengan yang kita lakukan di Microsoft Access. Simpan diagram ini.

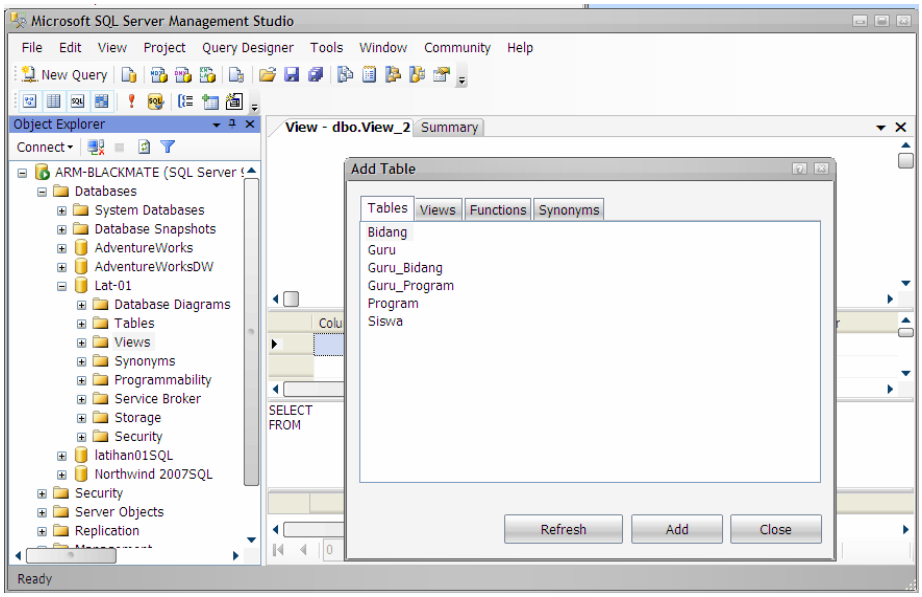


Gambar 12.12. Relasi antar tabel.

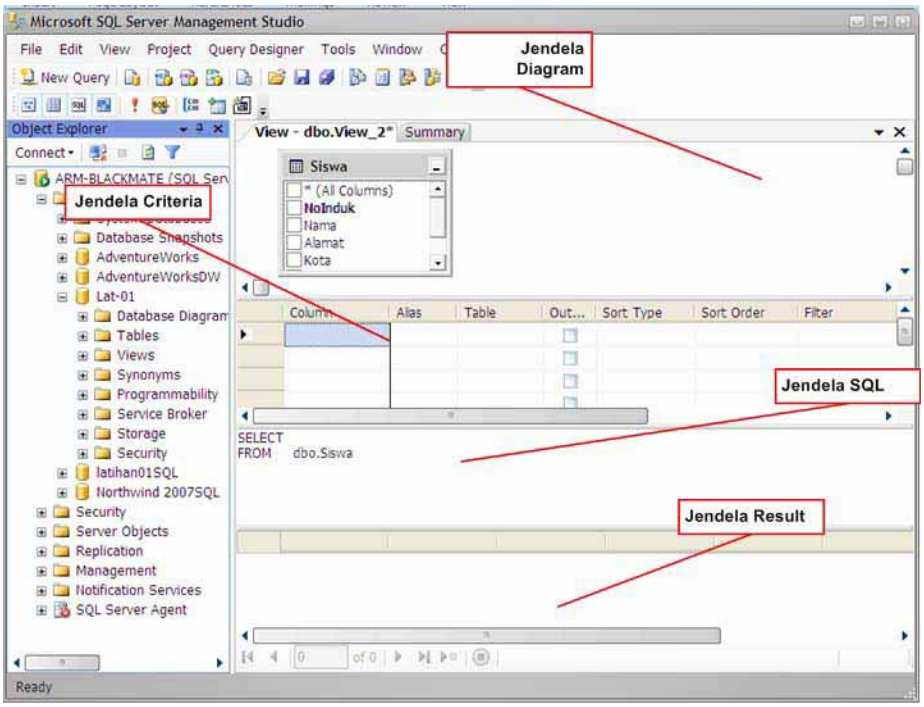
12.5.2. Membuat View

View sama dengan *Query* pada Microsoft Access. SQL Server juga menyediakan fitur GUI untuk membuat View seperti halnya GUI *Query* pada Microsoft Access. Untuk membuat *View*, klik kanan pada *View* kemudian pilih New *View* sehingga jendela seperti Gambar 12.13 terbuka. Pilih Add Table untuk menambahkan satu atau lebih tabel yang akan kita buat *View*-nya. Misalnya kita pilih tabel siswa. Klik Close untuk menutup jendela tersebut. Sehingga akan muncul tampilan seperti tampak pada Gambar 12.14.

Ada empat bagian penting pada Gambar 12.14 yaitu jendela Diagram, jendela Criteria, jendela SQL dan jendela Result. Jendela Diagram digunakan sebagai tempat tabel-tabel sumber yang akan kita buat *View*-nya. Jendela Criteria adalah tempat kita menentukan kriteria-kriteria pada *View*. Jendela SQL adalah tempat menuliskan perintah SQL dari *View* yang kita buat. Jendela Result adalah tempat untuk menampilkan hasil dari *View*.

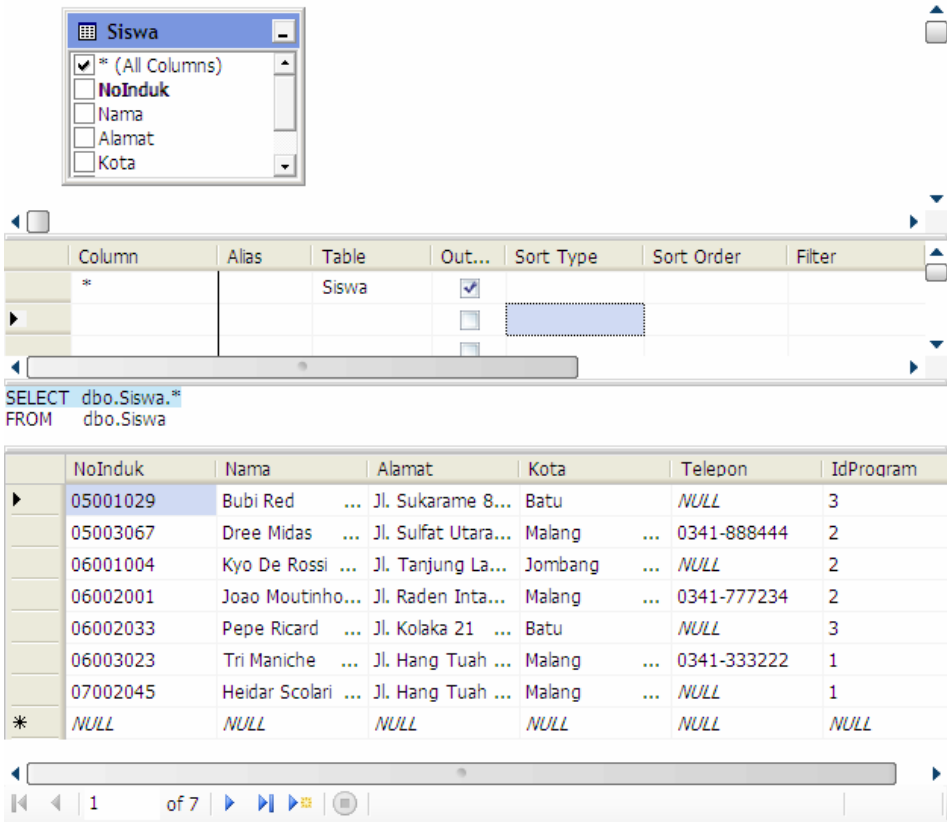


Gambar 12.13. Jendela untuk menentukan tabel yang akan dibuat *View*.



Gambar 12.14. Jendela untuk membuat *View*.

Pada Gambar 12.14, kita telah memilih tabel Siswa sebagai sumber *View* dan muncul pada jendela Diagrams. Kita akan membuat *View* sederhana berdasarkan tabel ini. Klik *(All Columns) di bawah nama tabel Siswa. Isi jendela SQL otomatis akan berubah. Isi jendela SQL akan berubah sesuai dengan apa yang kalian lakukan pada jendela Diagram dan jendela Criteria. Kemudian klik tanda seru (!) pada toolbar untuk mengeksekusi *View* tersebut atau klik kanan di salah satu bagian pada Gambar 12.14 (boleh di jendela *Diagram/Criteria/SQL/Result*) kemudian pilih Execute SQL. Kalian akan memperoleh tampilan seperti pada Gambar 12.15. Simpan *View* ini dengan cara klik pada ikon Disket pada Toolbar kemudian masukkan nama *View*.

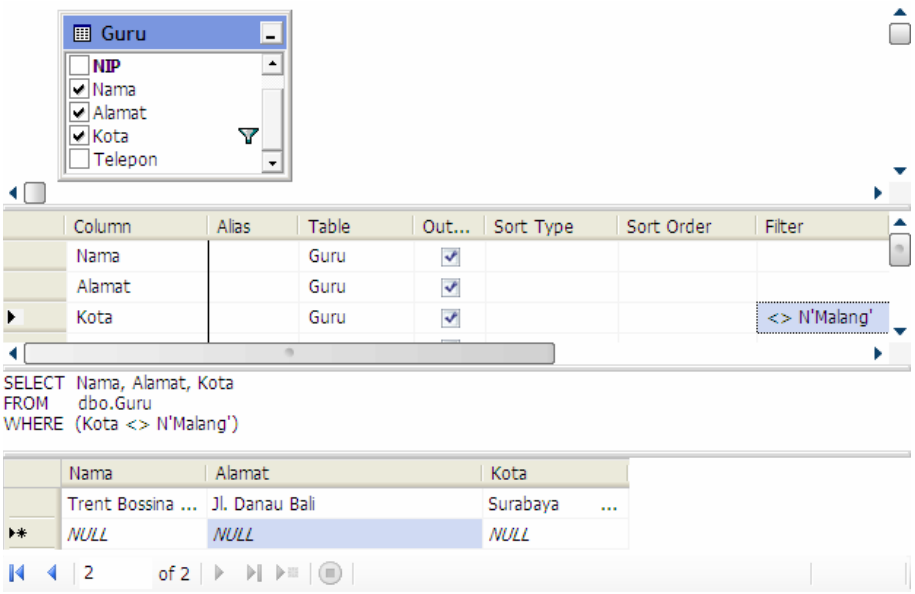


Gambar 12.15. Hasil eksekusi *View*.

Kalau kalian perhatikan, langkah-langkah membuat *View* di atas sama persis dengan apa yang kalian lakukan ketika membuat *Query* pada Microsoft Access. Kita akan buat beberapa contoh yang lebih kompleks. Perhatikan contoh-contoh berikut.

Contoh 12.1. *View* untuk menampilkan nama Guru yang alamat rumahnya di luar kota Malang.

Pada contoh ini kita menggunakan tabel Guru sebagai sumber *View*. Buat *View* baru kemudian tampilkan jendela seperti pada Gambar 12.13. Pilih tabel Guru kemudian *Close*. Klik Nama, Alamat dan Kota pada kotak tabel Guru. Kemudian pada jendela *Criteria*, pilih baris Kota dan pada kolom *Filter* ketikkan pernyataan <> 'Malang'. Jalankan *View* ini, kalian akan mendapatkan hasil seperti pada Gambar 12.16.

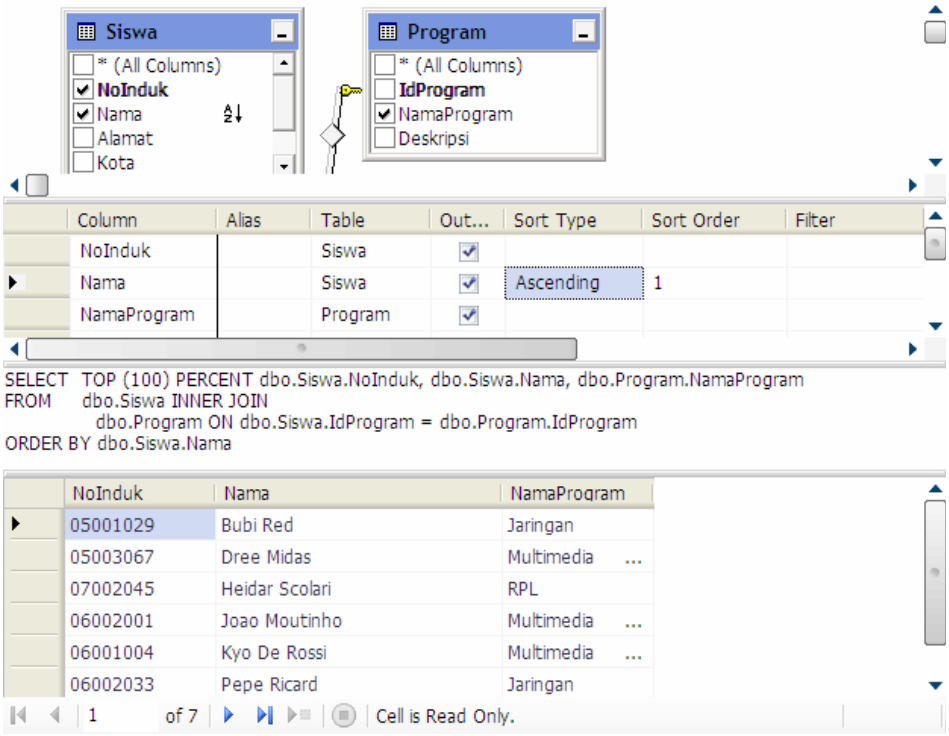


Gambar 12.16. Hasil eksekusi *View* contoh 12.1.

Kolom *Filter* pada jendela *Criteria* digunakan untuk memilih baris yang sesuai dengan keinginan kita. Pada contoh di atas, digunakan untuk memilih baris yang isi kolom Kotanya bukan 'Malang'. Kita menggunakan tanda <> untuk menyatakan ketidaksamaan.

Contoh 12.2. *View* untuk menampilkan nama, nomor induk, dan program keahlian Siswa secara urut abjad nama.

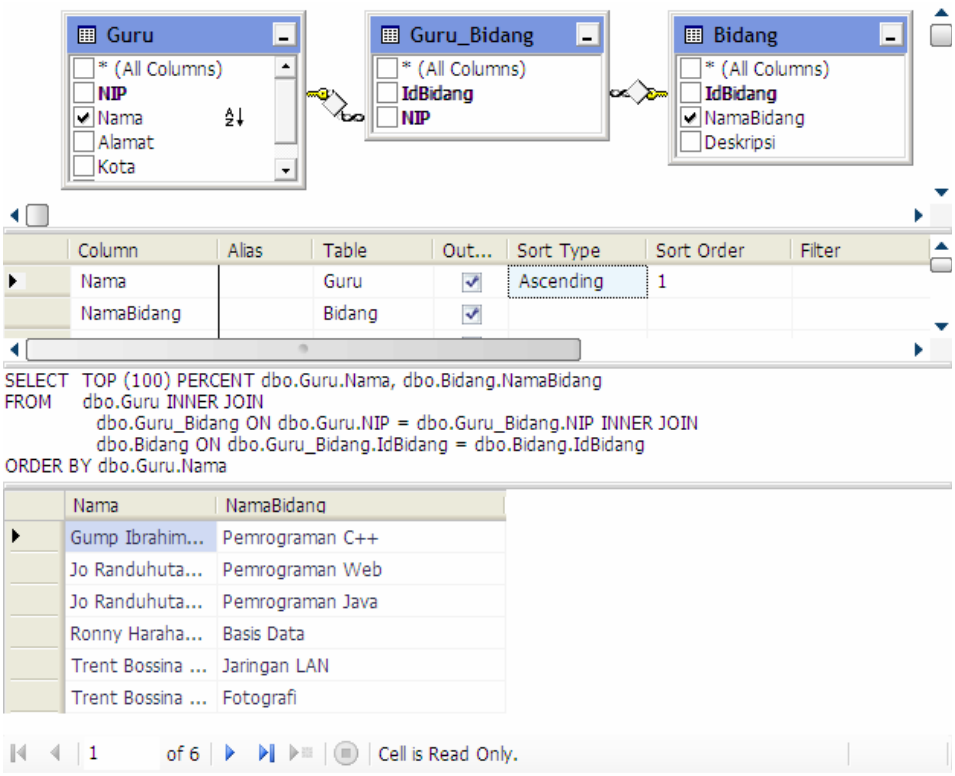
Contoh ini membutuhkan minimal 2 tabel, yaitu tabel Siswa dan tabel Program. Perhatikan relasi antar tabel pada Gambar 12.12. Tabel Siswa berhubungan langsung dengan tabel Program. Dengan cara yang sama seperti Contoh 12.13, pilih tabel Siswa dan tabel Program. Pada kotak Siswa di jendela Diagram pilih Nama dan NoInduk sedangkan pada kotak Program pilih NamaProgram. Pada jendela Criteria, pilih baris Nama dan klik pada kolom Sort Type kemudian pilih Ascending. Kemudian jalankan *View* tersebut. Kalian akan mendapat hasil seperti pada Gambar 12.17. Perhatikan pada jendela *Result*, kolom Nama ditampilkan berurutan sesuai dengan abjad.



Gambar 12.17. Hasil eksekusi *View* contoh 12.2.

Contoh 12.3. *View* untuk menampilkan nama Guru dan bidang keahliannya secara urut abjad nama.

Berdasarkan relasi tabel pada Gambar 12.12, maka *View* pada contoh ini membutuhkan tiga buah tabel yaitu tabel Guru, Guru_Bidang dan Bidang. Kolom Nama ada di tabel Guru sedangkan kolom NamaBidang ada pada tabel Bidang. Pilih tiga tabel tersebut. Pada kotak Guru di jendela Diagram pilih Nama kemudian pada kotak Bidang pilih NamaBidang. Pada jendela Criteria, pilih baris Nama dan klik pada kolom Sort Type kemudian pilih Ascending. Jalankan *View* ini. Perhatikan hasil yang diperoleh.



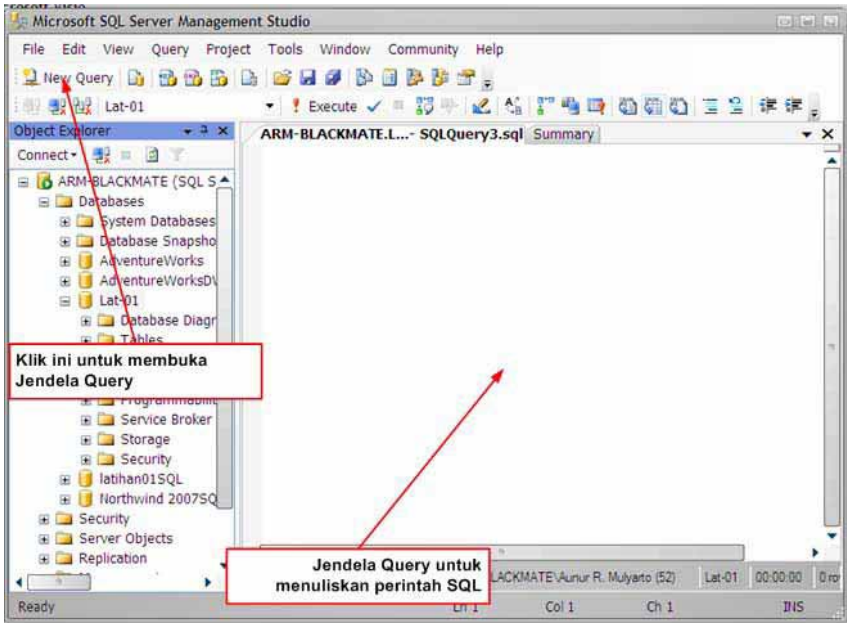
Gambar 12.18. Hasil eksekusi *View* contoh 12.3.

12.6. MENGGUNAKAN T-SQL

Transact-SQL (disingkat T-SQL) adalah varian dari bahasa basis data SQL yang dikeluarkan oleh perusahaan Microsoft dan Sybase. Microsoft SQL Server hanya mengenali bahasa SQL tipe T-SQL ini. Oleh karena itu jika kalian ingin menggunakan Microsoft SQL Server, maka kalian harus memahami bahasa ini.

Secara umum T-SQL tidak berbeda jauh dengan SQL standard. Hal ini karena T-SQL sebenarnya adalah bahasa SQL standar yang diberi tambahan perintah-perintah khusus untuk membantu kerja dalam SQL Server. Perintah SELECT, FROM dan WHERE yang telah kita singgung di awal bab, tetap menjadi perintah utama di T-SQL.

Untuk membuat perintah-perintah T-SQL ini SQL Server mempunyai jendela Query yang dapat kita buka dengan cara klik pada tombol *New Query* di bawah *Menu Bar* (lihat Gambar 12.19). Sebelumnya pilih terlebih dahulu basis data yang ingin kita gunakan dengan cara klik pada nama basis data di *Object Explorer*. Setelah jendela *Query* terbuka kalian dapat mulai mengetikkan perintah-perintah SQL.



Gambar 12.19. Membuka jendela *query*.

12.6.1. Definisi Tabel dengan T-SQL

Hampir semua aktivitas di dalam SQL Server dapat dilakukan dengan menggunakan perintah-perintah T-SQL. Termasuk membuat tabel. Pada sub bab sebelumya kalian telah membuat tabel-tabel dengan menggunakan fasilitas GUI. Kalian juga bisa membuat tabel-tabel tersebut dengan perintah-perintah SQL. Perintah-perintah yang berhubungan dengan definisi tabel termasuk dalam kategori DDL. Perintah untuk pendefinisian atau pembuatan tabel baru adalah CREATE TABLE. Sedangkan untuk menghapus kita menggunakan perintah DROP TABLE

Buat basis data baru dengan nama Lat-01_SQL. Kemudian pilih basis data tersebut. Buka jendela *Query* seperti pada Gambar 12.18. Kiata akan membuat tabel-tabel yang sama seperti pada Lat-01 tetapi dengan perintah SQL. Tabel pertama yang kita buat adalah tabel Siswa (lihat kembali struktur tabel ini pada Tabel 12.1 di atas). Ketikkan perintah berikut ini, kemudian jalankan dengan klik tanda seru (!).

```
CREATE TABLE [dbo].[Bidang](
      [IdBidang] [smallint] NOT NULL,
      [NamaBidang] [nchar](20)NULL,
      [Deskripsi] [nchar](100)NULL,
 CONSTRAINT [PK_Bidang] PRIMARY KEY CLUSTERED
(
      [IdBidang] ASC
)WITH (IGNORE_DUP_KEY = OFF) ON [PRIMARY]
     ) ON [PRIMARY]
```

Perintah CREATE TABLE diikuti dengan nama tabel yang akan kita buat ([dbo].[Bidang]) kemudian diikuti dengan daftar kolom yang ada pada tabel tersebut. Pada daftar kolom ini, tipe data, lebar data dan kondisi lainnya (misalnya NOT NULL atau NULL) harus dicantumkan. Setelah itu baru bagian CONSTRAINT dari tabel tersebut dituliskan. Bagian CONSTRAINT ini biasanya berisi pendefinisian Primary Key dari tabel tersebut. Perhatikan cara penulisan perintah-perintah di atas.

Setelah kalian jalankan, periksalah pada bagian node Tables apakah tabel kalian sudah terbentuk atau belum. Klik kanan pada Tables basis data kalian di Object Explorer kemudian pilih Refresh. Tabel baru yang kalian buat akan muncul di bawah Object Tables. Buatlah tabel-tabel lainnya dengan cara yang sama. Berikut ini perintah-perintah pembuatan masing-masing tabel.

Tabel Program

```
CREATE TABLE [dbo].[Program](
      [IdProgram] [smallint] NOT NULL,
      [NamaProgram] [nchar](20) NULL,
      [Deskripsi] [nchar](100) NULL,
 CONSTRAINT [PK_Program] PRIMARY KEY CLUSTERED
(
      [IdProgram] ASC
)WITH (IGNORE_DUP_KEY = OFF) ON [PRIMARY]
     ) ON [PRIMARY]
```

Tabel Guru

```
CREATE TABLE [dbo].[Guru](
      [NIP] [nchar](15) NOT NULL,
      [Nama] [nchar](20) NULL,
      [Alamat] [nchar](30) NULL,
      [Kota] [nchar](20) NULL,
      [Telepon] [nchar](15) NULL,
 CONSTRAINT [PK_Guru] PRIMARY KEY CLUSTERED
(
      [NIP] ASC
)WITH (IGNORE_DUP_KEY = OFF) ON [PRIMARY]
     ) ON [PRIMARY]
```

Tabel Siswa

```
CREATE TABLE [dbo].[Siswa](
      [NoInduk] [nchar](10) NOT NULL,
      [Nama] [nchar](20) NULL,
      [Alamat] [nchar](30) NULL,
      [Kota] [nchar](20) NULL,
      [Telepon] [nchar](15) NULL,
      [IdProgram] [smallint] NULL,
 CONSTRAINT [PK_Siswa] PRIMARY KEY CLUSTERED
(
      [NoInduk] ASC
)WITH (IGNORE_DUP_KEY = OFF) ON [PRIMARY]
```

```
     ) ON [PRIMARY]
```

Tabel Guru_Program

```
CREATE TABLE [dbo].[Guru_Program](
      [NIP] [nchar](15) NOT NULL,
      [IdProgram] [smallint] NOT NULL,
 CONSTRAINT [PK_Guru_Program] PRIMARY KEY CLUSTERED
(
      [NIP] ASC,
      [IdProgram] ASC
)WITH (IGNORE_DUP_KEY = OFF) ON [PRIMARY]
     ) ON [PRIMARY]
```

Tabel Guru_Bidang

```
CREATE TABLE [dbo].[Guru_Bidang](
      [IdBidang] [smallint] NOT NULL,
      [NIP] [nchar](15) NOT NULL,
 CONSTRAINT [PK_Guru_Bidang] PRIMARY KEY CLUSTERED
(
      [IdBidang] ASC,
      [NIP] ASC
)WITH (IGNORE_DUP_KEY = OFF) ON [PRIMARY]
     ) ON [PRIMARY]
```

Pada perintah-perintah pembuatan tabel di atas, kita belum membuat relasi antar tabel. Relasi antar tabel pada SQL biasanya dinyatakan dalam hubungan FOREIGN KEY (lihat kembali pengertian FOREIGN KEY pada bab 10 dan 11). Kita perlu memodifikasi tabel dengan menambahkan CONSTRAINT agar tabel dapat mengerti relasi antar tabel. Buka kembali jendela *Query* kemudian ketikkan perintah-perintah berikut ini.

```
ALTER TABLE [dbo].[Siswa]  WITH CHECK ADD  CONSTRAINT
[FK_Siswa_Program] FOREIGN KEY([IdProgram])
     REFERENCES [dbo].[Program] ([IdProgram])

ALTER TABLE [dbo].[Guru_Program]  WITH CHECK ADD
CONSTRAINT [FK_Guru_Program_Guru] FOREIGN KEY([NIP])
     REFERENCES [dbo].[Guru] ([NIP])

ALTER TABLE [dbo].[Guru_Bidang]  WITH CHECK ADD  CONSTRAINT
[FK_Guru_Bidang_Bidang] FOREIGN KEY([IdBidang])
     REFERENCES [dbo].[Bidang] ([IdBidang])

ALTER TABLE [dbo].[Guru_Bidang]  WITH CHECK ADD  CONSTRAINT
[FK_Guru_Bidang_Guru] FOREIGN KEY([NIP])
     REFERENCES [dbo].[Guru] ([NIP])
```

Perintah untuk memodifikasi atau merubah struktur tabel adalah ALTER TABEL kemudia diikuti dengan nama tabel yang ingin dirubah. Perhatikan pada baris **`ALTER TABLE [dbo].[Siswa] WITH CHECK ADD CONSTRAINT [FK_Siswa_Program] FOREIGN KEY([IdProgram])`**. Tabel yang ingin

dimodifikasi adalah dbo.Siswa dan tabel ini berhubungan dengan tabel Program sehingga dituliskan FK_Siswa_Program. Kolom IdProgram pada tabel Siswa merupakan FOREIGN KEY sehingga dituliskan sebagai **FOREIGN KEY([IdProgram])**. Kolom IdProgram ini berasal dari tabel program sehingga pada bagian akhir perintah harus dituliskan referensi tabelnya (ditulis dengan **REFERENCES [dbo].[Program] ([IdProgram])**). Dengan cara yang sama relasi-relasi antar tabel di atas dibuat.

Perintah DROP TABEL sangat mudah dilakukan yaitu tinggal menambahkan nama tabel didepan perintah tersebut. Misalnya **DROP TABLE [dbo].[Bidang]**.

12.6.2. Pengisian, Perubahan dan Penghapusan Isi Tabel dengan SQL

Setelah semua tabel selesai dibuat, maka kita dapat mengisi data pada masing-masing tabel dengan menggunakan perintah INSERT. Perintah ini termasuk dalam kelompok DML. Kita akan mencoba menggunakan perintah ini pada tabel Bidang. Kita periksa dulu isi tabel Bidang dengan cara klik kanan pada nama tabel kemudian pilih *Open Table*. Isi tabel akan muncul pada jendela Summary (Gambar 12.20). Tutup kembali jendela Open Table tersebut.

	IdBidang	NamaBidang	Deskripsi
	1	Pemrograman Web	*NULL*
	2	Pemrograman C++	*NULL*
	3	Pemrograman Java	*NULL*
	4	Jaringan LAN	*NULL*
	5	Fotografi	*NULL*
	6	Animasi Komputer	*NULL*
▶	7	Basis Data	*NULL*
*	*NULL*	*NULL*	*NULL*

Gambar 12.20. Isi tabel Bidang.

Buka jendela *Query* kemudian ketikkan perintah berikut.

```
INSERT INTO [Lat-01].[dbo].[Bidang]
           ([IdBidang]
           ,[NamaBidang]
           ,[Deskripsi])
     VALUES
                (8, 'Jaringan Wireless', NULL)
```

Perintah INSERT diikuti dengan nama basis data dan nama tabel (**[Lat-01].[dbo].[Bidang]**)kemudian diikuti dengan nama kolom yang ada pada tabel tersebut. Kata kunci VALUES digunakan untuk memberi petunjuk bahwa bagian setelah kata kunci ini adalah data yang akan dimasukkan. Perhatikan,

karena ada 3 kolom di tabel Bidang, maka kita juga memasukkan 3 buah data. Data pertama (8) harus terisi karena pada definisi tabel Bidang, kolom IdBidang dinyatakan NOT NULL. Data ketiga yaitu untuk kolom Deskripsi kita isi NULL karena kita ingin mengosongkan isi kolom ini.

Jalankan perintah di atas. Kemudian klik kanan pada tabel Bidang kemudian pilih *Refresh*. Kemudian klik kanan lagi pada tabel Bidang dan pilih *Open Table*. Data yang kalian masukkan sudah berada pada tabel Bidang (Gambar 12.21).

	IdBidang	NamaBidang	Deskripsi
▶	1	Pemrograman Web	*NULL*
	2	Pemrograman C++	*NULL*
	3	Pemrograman Java	*NULL*
	4	Jaringan LAN	*NULL*
	5	Fotografi	*NULL*
	6	Animasi Komputer	*NULL*
	7	Basis Data	*NULL*
	8	Jaringan Wireless	*NULL*
*	*NULL*	*NULL*	*NULL*

Gambar 12.21. Isi tabel Bidang setelah INSERT data.

Kalau kalian perhatikan, perintah INSERT di atas terlalu panjang. Coba hapus bagian daftar nama kolom pada perintah tersebut. Kemudian ganti data di bawah VALUES menjadi (9, 'Digital Animation', NULL). Kemudian jalankan dan periksalah hasilnya. Apa yang terjadi? Setelah *Refresh*, kalian akan menjumpai bahwa data kalian juga dapat dimasukkan tanpa harus menyebut daftar nama kolom pada perintah INSERT.

Seringkali dalam pengisian data pada suatu tabel, kita melakukan kesalahan. Kesalahan dapat berupa kesalahan ketik atau kesalahan pembacaan data. Sehingga ketika diperiksa, kita menginginkan untuk merubah/memperbaiki data tersebut. Proses ini biasa disebut sebagai update data. SQL menyediakan perintah UPDATE untuk melakukan proses ini. Perintah ini masih termasuk dalam kelompok DML.

Misalnya kita ingin merubah isi kolom NamaBidang pada IdBidang yang ke-5 (lihat Gambar 12.21). Dari 'Fotografi' kita rubah menjadi 'Fotografi Digital' maka kita ketikkan perintah seperti berikut.

```
UPDATE [Lat-01].[dbo].[Bidang]
   SET [NamaBidang] = 'Fotografi Digital'
 WHERE [IdBidang] = 5
```

Periksa kembali isi tabel Bidang. Jangan lupa untuk me-*refresh* dulu tabel Bidang sebelum perintah *Open Table* dijalankan.

Penghapusan data dengan SQL dilakukan dengan perintah DELETE. Perlu diperhatikan bahwa perintah DELETE akan menghapus isi seluru baris. Kalau kalian hanya ingin mengosongkan isi satu bagian dari baris (atau satu sel) saja gunakan perintah UPDATE. Misalnya kita ingin kita menghapus baris yang IdBidangnya sama dengan 8 maka kita ketikkan dengan perintah seperti berikut.

```
DELETE FROM [Lat-01].[dbo].[Bidang]
      WHERE [IdBidang] = 8
```

12.6.3. Mencari dan Menampilkan Data (*View*) dengan SQL

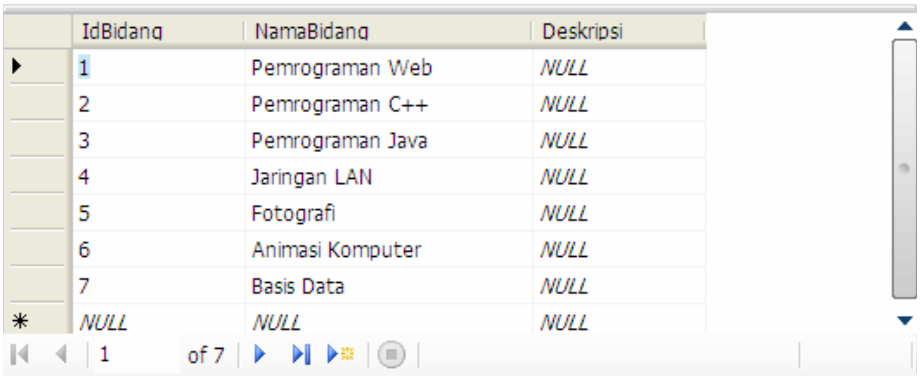
Seperti telah disinggung di awal bab, perintah utama dalam SQL adalah SELECT, FROM dan WHERE. Sebagian besar aktivitas kita menggunakan SQL berhubungan dengan perintah-perintah ini. Untuk menggunakan perintah-perintah ini, buka jendela Query kemudian ketikkan perintah yang kalian inginkan. Jalankan dengan menekan tombol tanda seru (!).

Perintah SELECT digunakan untuk memilih kolom mana saja yang akan ditampilkan. Jika kita ingin menampilkan semua kolom, kita cukup menggunakan tanda *. Perintah FROM digunakan untuk menentukan asal kolom yang ingin kita tampilkan datanya. Perintah WHERE digunakan untuk membatasi baris-baris yang akan kita tampilkan agar sesuai dengan kriteria yang kita inginkan. WHERE bisa dari tabel atau hasil dari *View* yang lain. Perhatikan contoh-contoh penggunaan SQL berikut ini.

Contoh 12.4. Menampilkan semua data pada tabel Bidang.

Contoh ini kita membutuhkan tabel Bidang. Sedangkan kolom yang kita ingin tampilkan adalah semua kolom. Kita dapat menggunakan tanda * atau mendaftar semua kolom yang ada pada tabel Bidang. Berikut adalah perintah SQL dan hasil eksekusinya.

```
SELECT  IdBidang, NamaBidang, Deskripsi
FROM    dbo.Bidang
```



IdBidang	NamaBidang	Deskripsi
1	Pemrograman Web	NULL
2	Pemrograman C++	NULL
3	Pemrograman Java	NULL
4	Jaringan LAN	NULL
5	Fotografi	NULL
6	Animasi Komputer	NULL
7	Basis Data	NULL
NULL	NULL	NULL

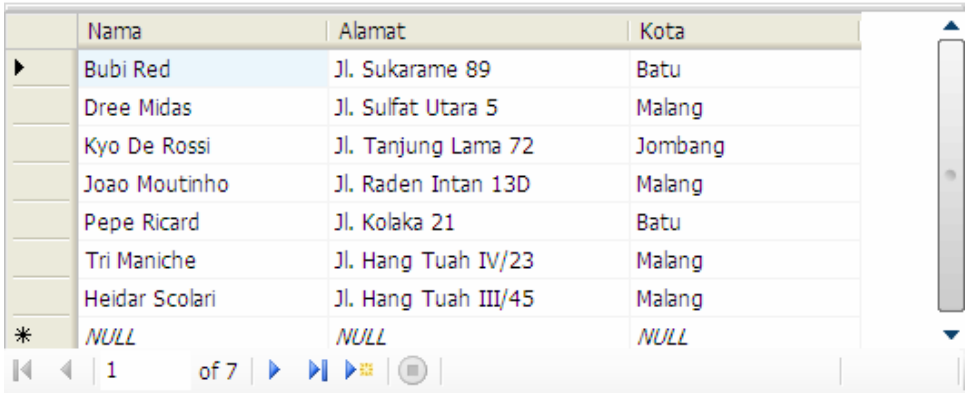
1 of 7

Jika kalian ingin menyimpan dalam bentuk View perintah di atas, tambahkan sebelum pernyataan SELECT perintah CREATE VIEW <nama_view> AS. Ganti <nama_view> dengan nama yang kalian inginkan.

Contoh 12.5. Menampilkan data nama dan alamat siswa.

Pada contoh ini kita membutuhkan tabel Siswa. Sedangkan kolom yang kita inginkan adalah kolom nama, alamat, dan kota. Kita tidak bisa menggunakan tanda *, tetapi harus mendaftarkan nama kolom nama dan alamat setelah perintah SELECT. Perhatikan perintah SQL dan outpunya berikut ini.

```
SELECT  Nama, Alamat, Kota
FROM    dbo.Siswa
```



	Nama	Alamat	Kota
▶	Bubi Red	Jl. Sukarame 89	Batu
	Dree Midas	Jl. Sulfat Utara 5	Malang
	Kyo De Rossi	Jl. Tanjung Lama 72	Jombang
	Joao Moutinho	Jl. Raden Intan 13D	Malang
	Pepe Ricard	Jl. Kolaka 21	Batu
	Tri Maniche	Jl. Hang Tuah IV/23	Malang
	Heidar Scolari	Jl. Hang Tuah III/45	Malang
*	*NULL*	*NULL*	*NULL*

1 of 7

Contoh 12.6. Menampilkan data nama siswa yang rumahnya di Batu.

Pada contoh ini kita membutuhkan pernyataan WHERE untuk membatasi baris yang ingin kita tampilkan. Tabel yang dipakai adalah tabel Siswa dan kolom yang ingin ditampilkan adalah kolom Nama. Kalau kalian perhatikan hasil eksekusi contoh 12.2 di atas, yang alamat rumahnya di Batu ada dua orang yaitu Pepe Ricard dan Bubi Red. Pernyataan WHERE diikuti dengan kondisi yang harus dipenuhi. Pada kasus ini adalah Kota='Batu'. Berikut ini perintah SQL secara lengkap dan outputnya. Perhatikan cara penulisannya.

```
SELECT  Nama
FROM    dbo.Siswa
WHERE  (Kota = 'Batu')
```

	Nama
▶	Bubi Red
	Pepe Ricard
*	*NULL*

Contoh 12.7. Menampilkan data nama, alamat dan kota dari guru yang rumahnya tidak di Malang.

Pada bagian ini kita juga menggunakan perintah WHERE untuk membatasi data yang ingin kita tampilkan. Tabel yang kita gunakan adalah tabel guru dan kolom yang kita butuhkan adalah kolom nama, alamat dan kota. Pernyataan yang

rumahnya tidak di Malang merupakan kondisi yang harus dipenuhi. Sehinga secara lengkap pernyataan akan tampak seperti berikut.

```
SELECT  Nama, Alamat, Kota
FROM    dbo.Guru
WHERE  (Kota <> 'Malang')
```

	Nama	Alamat	Kota
▶	Trent Bossina	Jl. Danau Bali	Surabaya
*	*NULL*	*NULL*	*NULL*

Contoh 12.8. Menampilkan data nama siswa dan program keahlian yang diikuti.

Pada contoh ini kita melibatkan dua buah tabel yang saling berhubungan yaitu tabel Siswa dan tabel Program. Yang kita ingin tampilkan adalah kolom Nama pada tabel Siswa dan kolom NamaProgram pada tabel Program. Untuk kasus seperti ini pada perintah SELECT pemanggilan nama kolom harus didahului dengan nama tabelnya. Sedangkan pada FROM kita menggunakan perintah INNER JOIN untuk menggabung dua tabel. Perlu kalian cermati (lihat Tabel 12.1 di atas), pada tabel Siswa terdapat Foreign Key yaitu IdProgram yang merupakan Primary Key pada tabel Program. IdProgram ini merupakan penghubung yang dapat kita gunakan untuk menggabung dua tabel. Perhatikan pernyataan SQL berikut ini dan cermati outputnya.

```
SELECT  dbo.Siswa.Nama, dbo.Program.NamaProgram
FROM    dbo.Siswa INNER JOIN
          dbo.Program ON dbo.Siswa.IdProgram = dbo.Program.IdProgram
```

	Nama	NamaProgram
▶	Bubi Red	Jaringan
	Dree Midas	Multimedia
	Kyo De Rossi	Multimedia
	Joao Moutinho	Multimedia
	Pepe Ricard	Jaringan
	Tri Maniche	RPL
	Heidar Scolari	RPL

1 of 7 Cell is Read Only.

Contoh 12.9. Menampilkan data nama siswa yang mengikuti program keahlian RPL.

Contoh ini merupakan kelanjutan dari contoh 12.4. Kita menambahkan pernyataan WHERE untuk menampilkan nama yang ikut program keahlian RPL. Selain itu kita hanya ingin menampilkan kolom Nama dari tabel Siswa. Perhatikan pernyataan SQL berikut dan hasil eksekusinya.

```
SELECT  dbo.Siswa.Nama
FROM    dbo.Siswa INNER JOIN
          dbo.Program ON dbo.Siswa.IdProgram = dbo.Program.IdProgram
WHERE  (dbo.Program.NamaProgram = 'RPL')
```

	Nama
▶	Tri Maniche
	Heidar Scolari

1 of 2 Cell is Read Only.

Contoh 12.10. Menampilkan data nama guru dan bidang keahliannya.

Contoh ini lebih kompleks dari contoh-contoh di atas. Kita melibatkan tiga buah tabel, yaitu tabel Guru, tabel Guru_Bidang dan tabel Bidang. Perhatikan relasi antar tabel tersebut pada Gambar 12.11. Ada dua INNER JOIN yang kita gunakan, yaitu INNER JOIN tabel Guru dengan tabel Guru_Bidang dan INNER JOIN Guru_Bidang dengan tabel Bidang. Perhatikan bagaimana membuat pernyataan INNER JOIN.

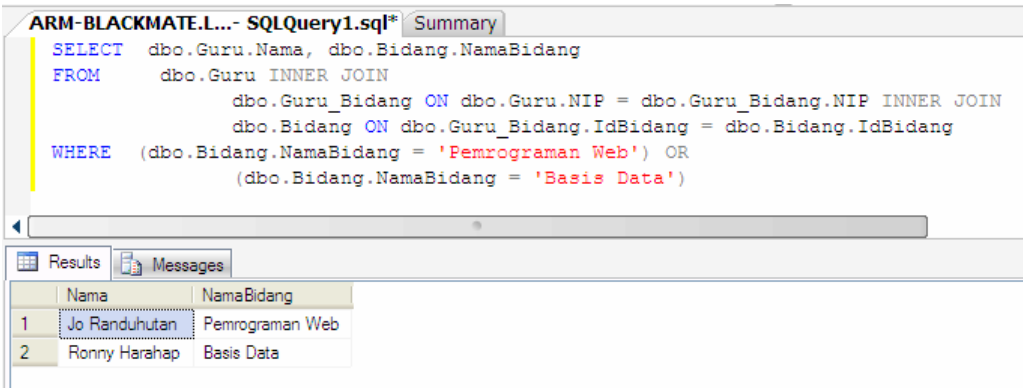
```
SELECT  dbo.Guru.Nama, dbo.Bidang.NamaBidang
FROM    dbo.Guru_Bidang INNER JOIN
          dbo.Bidang ON dbo.Guru_Bidang.IdBidang = dbo.Bidang.IdBidang INNER JOIN
          dbo.Guru ON dbo.Guru_Bidang.NIP = dbo.Guru.NIP
```

	Nama	NamaBidang
▶	Jo Randuhutan	Pemrograman Web
	Gump Ibrahimovic	Pemrograman C++
	Jo Randuhutan	Pemrograman Java
	Trent Bossina	Jaringan LAN
	Trent Bossina	Fotografi
	Ronny Harahap	Basis Data

1 of 6 Cell is Read Only.

Contoh 12.11. Menampilkan data nama guru yang bidang keahliannya Pemrograman Web atau Basis Data.

Contoh ini merupakan pengembangan dari contoh 12.10. Kita ingin menampilkan guru yang bidang keahliannya Pemrograman Web atau Basis Data. Kita perlu menambahkan perintah WHERE untuk membatasi baris. Pada WHERE kita juga perlu operator OR. Perhatikan perintah SQL dan hasil eksekusinya berikut ini.



Bandingkan dengan hasil eksekusi contoh 12.10. Cobalah ganti OR dengan AND dan jalankan kembali perintah SQL tersebut. Bagaimanakah hasilnya?

Contoh-contoh di atas dapat dikembangkan lagi dengan banyak variasi. Dengan banyak mencoba dan berlatih maka kalian akan dapat memahami dengan baik penggunaan perintah-perintah SQL.

12.7. FUNGSI, PROCEDURE DAN TRIGGER

Microsoft SQL Server menyediakan fasilitas-fasilitas tingkat lanjut yang tidak dimiliki oleh Microsoft Access seperti kemampuan membuat fungsi, mendefinisikan store procedure dan trigger. Hal ini karena SQL Server diperuntukkan sebagai basis data server yang membutuhkan kinerja yang lebih kuat dari Microsoft Access.

12.7.1. FUNGSI

T-SQL menyediakan banyak fungsi yang digunakan untuk mempermudah tugas-tugas dalam pengelolaan basis data. Beberapa fungsi penting akan disampaikan di sini. Untuk lebih lengkapnya silahkan baca manual atau online-help dari SQL Server.

- Fungsi-fungsi yang berhubungan dengan *numeric*

  Fungsi-fungsi penting yang berhubungan dengan numeric (angka) adalah *isNumeric* dan *Round*. Fungsi isNumeric digunakan untuk memeriksa apakah isi suatu data berupa data angka atau tidak. Buka jendela Query kemudian ketikkan perintah seperti pada contoh 12.12 dan periksalah hasilnya. Tabel Siswa mempunyai kolom Telepon yang isi datanya sekilas berupa angka. Kita dapat memeriksa dengan menggunakan fungsi isNumerik. Pada hasil eksekusi di bawah ini terlihat nilai hasil pemeriksaan menghasilkan angka 0 untuk seluruh data. Angka 0 berarti false. Berarti, semua data pada kolom Telepon bukan data numeric.

  Contoh 12.12. Menggunakan fungsi isNumeric.

```
SELECT NoInduk, Nama, Telepon, IsNumeric(Telepon) FROM dbo.Siswa
```

Results | Messages

	NoInduk	Nama	Telepon	(No column name)
1	05001029	Bubi Red	NULL	0
2	05003067	Dree Midas	0341-888444	0
3	06001004	Kyo De Rossi	NULL	0
4	06002001	Joao Moutinho	0341-777234	0
5	06002033	Pepe Ricard	NULL	0
6	06003023	Tri Maniche	0341-333222	0
7	07002045	Heidar Scolari	NULL	0

Fungsi ROUND digunakan untuk membulatkan bilangan pecahan ke bilangan bulat terdekat. Misalnya 13.58 akan menjadi 14.00. Perhatikan contoh berikut ini.

Contoh 12.13. Menggunakan fungsi Round.

```
SELECT OrderID, Freight, Round(Freight, 0)
FROM Orders
```

Results | Messages

	OrderID	Freight	(No column name)
1	10248	32.38	32.00
2	10249	11.61	12.00
3	10250	65.83	66.00
4	10251	41.34	41.00
5	10252	51.30	51.00
6	10253	58.17	58.00
7	10254	22.98	23.00
8	10255	148.33	148.00
9	10256	13.97	14.00

- Fungsi-fungsi yang berhubungan dengan *string*

  Fungsi-fungsi penting yang berhubungan dengan karakter (string) antara lain adalah **LEFT, RIGHT, LEN, LOWER, UPPER, LTRIM, dan RTRIM.**

  LEFT digunakan untuk memilih sejumlah karakter tertentu dari sebelah kiri sedangkan RIGHT dari sebelah kanan. Hasil dari RIGHT tergantung dari lebar data yang kalian tetapkan pada pembuatan table. LEN digunakan mengetahui panjang karakter pada data. Perhatikan contoh berikut.

Contoh 12.14. Menggunakan LEFT, RIGHT dan LEN.

```
SELECT NoInduk, LEFT(Nama,4), RIGHT(Nama,10), LEN(Telepon)
FROM dbo.Siswa
```

Results | Messages

	NoInduk	(No column name)	(No column name)	(No column name)
1	05001029	Bubi		NULL
2	05003067	Dree		11
3	06001004	Kyo	si	NULL
4	06002001	Joao	nho	11
5	06002033	Pepe	d	NULL
6	06003023	Tri	e	11
7	07002045	Heid	lari	NULL

LOWER digunakan untuk merubah karakter menjadi huruf kecil, sedangkan UPPER sebaliknya. LTRIM digunakan untuk menghilangkan space di sebelah kiri data string, sedangkan RTRIM di sebelah kanan. Lihat contoh berikut.

Contoh 12.15. Menggunakan fungsi UPPER dan LOWER.

```
SELECT NoInduk, UPPER(Nama), LOWER(Nama)
FROM dbo.Siswa
```

Results | Messages

	NoInduk	(No column name)	(No column name)
1	05001029	BUBI RED	bubi red
2	05003067	DREE MIDAS	dree midas
3	06001004	KYO DE ROSSI	kyo de rossi
4	06002001	JOAO MOUTINHO	joao moutinho
5	06002033	PEPE RICARD	pepe ricard
6	06003023	TRI MANICHE	tri maniche
7	07002045	HEIDAR SCOLARI	heidar scolari

- Fungsi-fungsi yang berhubungan dengan waktu

  Fungsi-fungsi yang berhubungan dengan waktu yang penting antara lain: GETDATE, MONTH, DAY, YEAR, DATENAME, DATEADD, and DATEDIFF. GETDATE digunakan untuk mendapatkan tanggal sekarang dari system computer. MONTH digunakan untuk mengambil bagian bulan dari data tanggal. DAY digunakan untuk mengambil bagian tanggal dari data tanggal. YEAR digunakan untuk mengambil bagian tahun dari data tanggal. DATENAME digunakan untuk mendapatkan nama hari dari suatu tanggal. DATEADD digunakan untuk menambah atau mengurangi data tanggal. DATEDIFF digunakan untuk melihat selisih antara dua buah data tanggal. Perhatikan contoh berikut.

Contoh 12.16. Menggunakan fungs-fungsi waktu.

```
SELECT GETDATE(), DAY(GETDATE())AS Tanggal, MONTH(GETDATE()) AS Bulan,
YEAR(GETDATE()) AS Tahun, DATENAME(dw,GETDATE())AS Hari
```

Results | Messages

	(No column name)	Tanggal	Bulan	Tah...	Hari
1	2008-07-19 19:07:56.437	19	7	2008	Saturday

12.7.2. Procedure dan Stored Prosedure

Selain fungsi-fungsi yang tersedia di atas, SQL Server juga memperkenalkan *user-defined function*. Fungsi ini adalah fungsi yang dapat kita buat sendiri untuk mempercepat pengelolaan basis data. Kadang-kadang fungsi jenis ini disebut juga procedure. Ada dua tipe user defined function yaitu *scalar* dan *inline table-value*. Fungsi scalar menghasilkan satu nilai keluaran melalui kata kunci Return. Inline Table-Value menghasilkan suatu table baru ketika dieksekusi. Perhatikan contoh berikut.

Contoh 12.17. Membuat fungsi scalar.

Fungsi yang ada di samping ini diberi nama dbo.FullName. Parameter yang digunakan ada dua yaitu @FirstName dan @LastName. Perhatikan bagaimana mendefinisikan suatu fungsi dan parameternya.

```
CREATE FUNCTION dbo.FullName
    (@FirstName nVarChar(10),
    @LastName nVarChar(20))
RETURNS nVarChar(35)
BEGIN
    RETURN (@LastName + ', ' + @FirstName)
END

SELECT FirstName, LastName,
    dbo.FullName(FirstName, LastName)
    FROM Employees
```

Results | Messages

	FirstName	LastName	(No column name)
1	Nancy	Davolio	Davolio, Nancy
2	Andrew	Fuller	Fuller, Andrew
3	Janet	Leverling	Leverling, Janet
4	Margaret	Peacock	Peacock, Margaret
5	Steven	Buchanan	Buchanan, Steven
6	Michael	Suyama	Suyama, Michael
7	Robert	King	King, Robert
8	Laura	Callahan	Callahan, Laura

Output dari fungsi akan bertipe data *nVarChar* dengan lebar data 35. Fungsi ini akan memberikan output satu nilai yaitu gabungan dari @LastName dan @Firstname. Sehingga kita bisa nyatakan ini termasuk dalam fungsi scalar.

Pada kode yang di blok, terlihat bagaimana fungsi tersebut dipanggil pada suatu pernyataan query. Perhatikan output yang dihasilkan. Terbentuk kolom baru yang berisi gabungan kolom LastName dan FirstName.

Stored procedure adalah potongan kode program yang dapat menerima parameter input dan menghasilkan satu atau lebih parameter output. Stored procedure umumnya digunakan untuk operasi-operasi pada basis data. Biasanya suatu perintah SQL yang rumit, panjang dan kompleks disimpan sebagai stored

procedure. Jika kita ingin melakukan operasi tersebut kita tidak perlu mengetik ulang, cukup kita panggil nama stored procedure dan kita eksekusi langsung.

Perintah untuk membuat stored procedure adalah CREATE PROCEDURE kemudian diikuti dengan nama procedure-nya. Perhatikan contoh berikut ini.

Contoh 12.18. Membuat stored procedure.

Buka jendela Query, kemudian ketikkan kode berikut ini.

```
CREATE PROCEDURE hapusBaris
      @IdNumber smallint
AS
DELETE
FROM dbo.Bidang Where dbo.Bidang.IdBidang = @IdNumber
```

Procedure yang kita buat ini bernama hapusBaris dengan satu parameter yaitu @IdNumber dengan tipe data smallint. Pernyataan setelah AS adalah pernyataan SQL yang akan dikerjakan ketika stored procedure di atas dijalankan.

12.7.3.Trigger

Trigger adalah tipe khusus dari stored procedure yang akan dieksekusi ketika suatu kejadian muncul. Kejadian tersebut misalnya ketika ada penambahan data (INSERT), penghapusan data (DELETE) atau perubahan data (UPDATE). Trigger biasanya merupakan komponen dari suatu table.

Cara membuatnya adalah klik node di depan table yang anda pilih. Setelah muncul daftar komponen table tersebut klik kanan pada Trigger dan pilih New Trigger. SQL Server akan menampilkan jendela baru yang isinya adalah template dari Trigger. Hapus semua teks pada jendela tersebut kemudian ketikkan contoh berikut ini.

Contoh 12.19. Membuat trigger.

```
CREATE TRIGGER coba_trigger
ON dbo.Bidang FOR INSERT
AS
DECLARE @varNama Varchar(20)
SELECT @varNama = NamaBidang FROM INSERTED
PRINT 'Anda baru memasukan data : ' + @varNama
```

Trigger di atas berada pada table Bidang. Nama triggernya adalah coba_trigger dan akan dijalankan ketika event pemasukkan data (INSERT) pada table dbo.Bidang terjadi.

12.8. ADMINISTRASI SQL SERVER

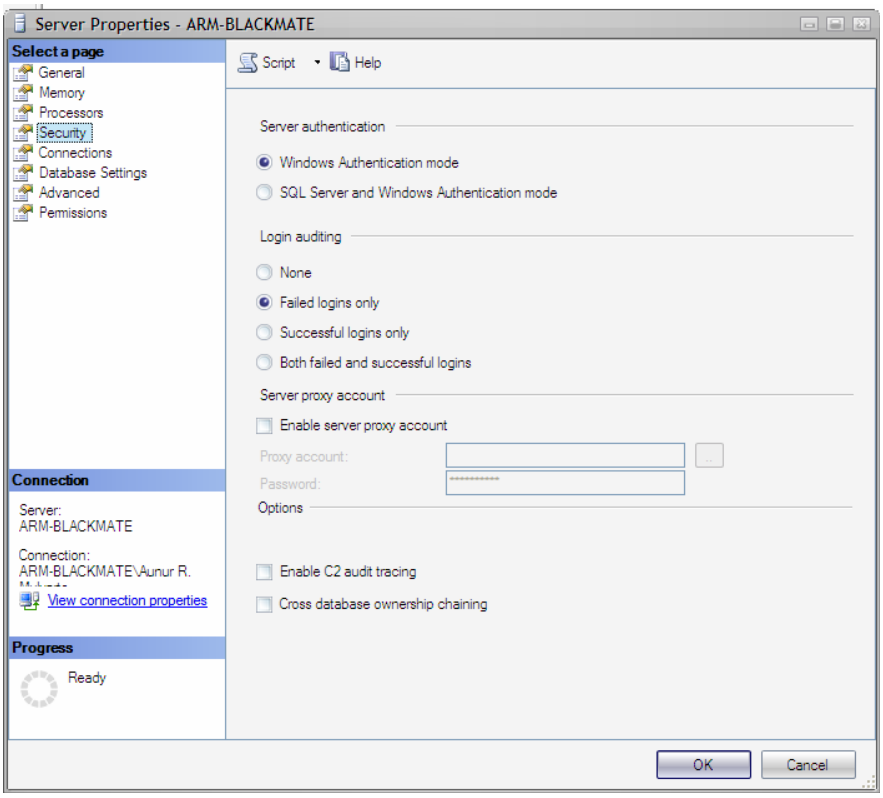
12.8.1.Security dan Authentication.

Security atau keamanan basis data merupakan komponen yang sangat penting. Karena sifatnya yang berperan sebagai server maka sebenarnya SQL Server secara teori dapat diakses oleh siapa saja. Hal ini tentu akan sangat merugikan jika ada pihak-pihak yang tidak bertanggung jawab melakukan perubahan, perusakan atau bahkan menghapus basis data yang kita buat dengan susah payah.

SQL Server menyediakan mekanisme *Authentication* untuk membatasi siapa yang berhak masuk ke dalam sistem SQL Server. Ada dua model *authentication* yaitu *Windows Authentication* dan *SQL Server Authentication*.

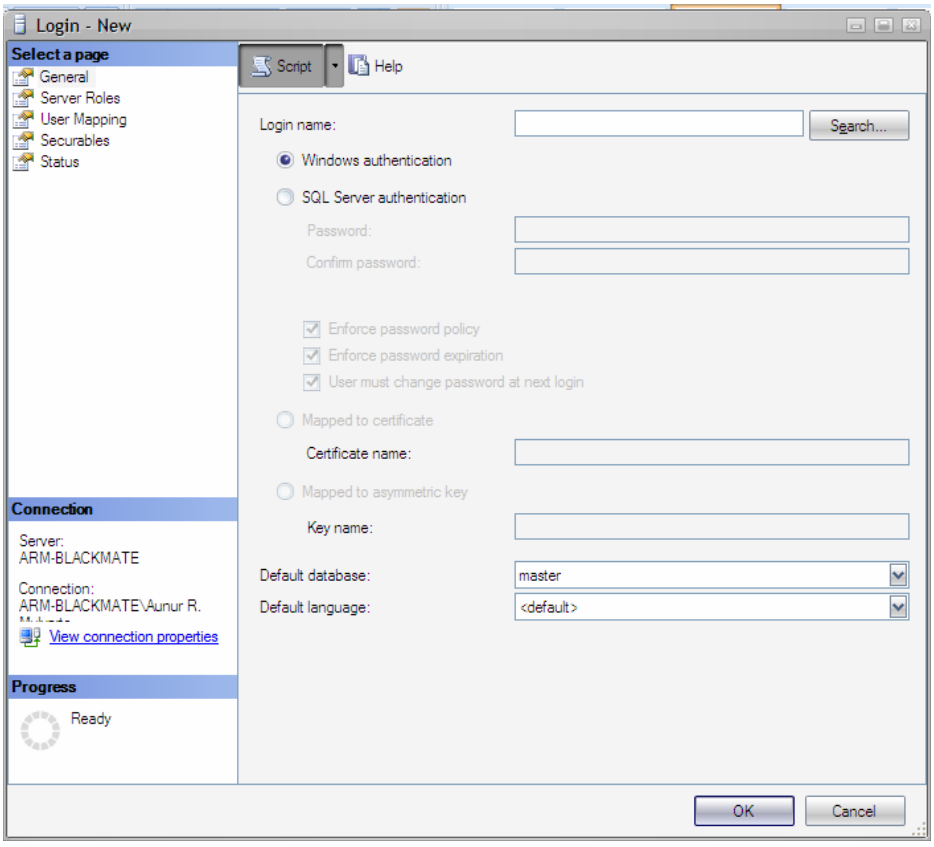
- *Windows Authentication*. Pada model ini SQL Server tidak melakukan pengecekan pada user name dan *password* yang diberikan namun dipercayakan pada system operasi Windows. Artinya sebenarnya hanya user yang terdaftar pada computer tersebut yang mungin bisa masuk pada SQL Server.
- SQL Server Authentication. Informasi user name dan password disimpan dalam SQL Server sehingga setiap kali ada user yang akan masuk koneksi ke SQL Server, SQL Server akan memeriksanya.

Untuk melihat model authentication apa yang ada di SQL Server, klik kanan pada nama server di jendela Object Explorer kemudian pilih Properties. Pada jendela Server Properties pilih bagian Security sehingga tampilan akan tampak seperti pada Gambar 12.22.



Gambar 12.22. Halaman security pada jendela Server Properties.

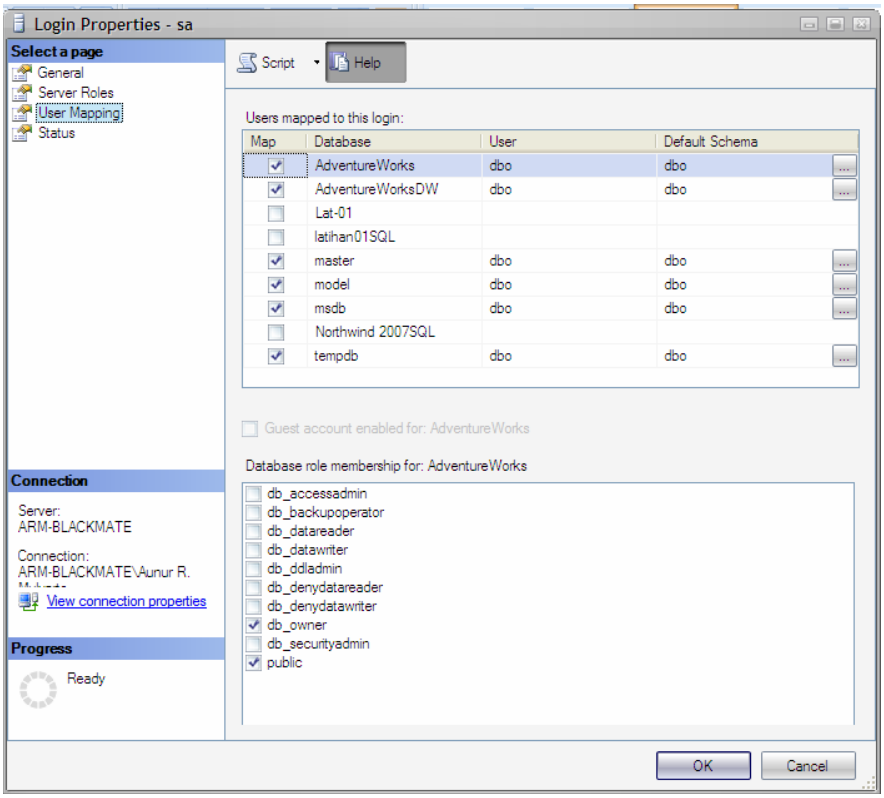
Untuk menambahkan user yang bisa masuk ke SQL Server, klik node pada Security di Object Explorer, kemudian klik kanan Logins. Pilih New Logins untuk membuka jendela Login seperti pada Gambar 12.23. Buat user baru yang kalian inginkan.



Gambar 12.23. Jendela untuk membuat user baru.

12.8.2.Permissions

Permissions berhubungan dengan hak akses seorang user pada suatu basis data. Seorang yang sudah terdaftar sebagai user pada SQL Server tidak secara otomatis bisa menggunakan basis data yang ada jika belum diberi hak. Untuk mengubah hak user pada basis data tertentu, klik node pada Logins. Klik kanan pada nama user yang terdaftar di bawah Logins kemudian pilih Properties. Pada jendela Login Properties, pilih bagian User Mapping sehingga tampilan akan tampak seperti pada Gambar 12.24.



Gambar 12.24. Hak akses basis data oleh user.

Gambar di atas menunjukka *user* dengan nama **sa** merupakan pemilik (*db_owner*) dari beberapa basis data (lihat basis data yang dicentang). Kita dapat mengatur peran (*role*) seorang member pada basis data dengan mencentang atau tidak pada bagian *Database role membership.*

12.9. RINGKASAN

- *Data Definition Language* (DDL) adalah satu paket bahasa DBMS yang berguna untuk melakukan spesifikasi terhadap skema basis data sedangkan *Data Manipulation Language* (DML) adalah satu paket DBMS yang memperbolehkan pemakai untuk mengakses atau memanipulasi data sebagaimana yang telah diorganisasikan sebelumnya dalam model data yang tepat.
- SQL lebih menekankan pada aspek pencarian dari dalam tabel hal ini karena pencarian adalah inti dari pengelolaan data.
- Pembuatan Tabel dan *View* pada SQL server dapat dilakukan dengan cara GUI atau dengan menggunakan perintah-perintah SQL.
- Perintah utama SQL adalah SELECT, FROM dan WHERE.

- Microsoft SQL Server menyediakan fasilitas fungsi *built-in* dan *user-defined function.*
- *Stored procedure* adalah potongan kode program yang dapat menerima parameter input dan menghasilkan satu atau lebih parameter output. *Trigger* adalah tipe khusus dari *stored procedure* yang akan dieksekusi ketika suatu kejadian muncul
- SQL Server menyediakan mekanisme *Authentication* untuk membatasi siapa yang berhak masuk ke dalam sistem SQL Server dengan dua model yaitu *Windows Authentication* dan *SQL Server Authentication*. Selain itu keamanan data juga diatur dengan cara pemberian *permissions.*

12.10. SOAL-SOAL LATIHAN

1. Buatlah tabel baru dengan menggunakan fasilitas GUI pada basis data Lat-01 dengan nama MataPelajaran dan kolom-kolomnya adalah: IdMatapelajaran, Nama, Deskripsi, Semester, dan JamPertemuan. Kolom Semester menunjukkan pada semester berapa mata pelajaran tersebut di ajarkan sedangkan JamPertemuan menunjukkan berapa jam pertemuan mata pelajaran tersebut di ajarkan (kedua kolom ini menggunakan tipe data numeric, kalian bisa memilih *smallint* atau *int* pada tipe datanya). Isilah tabel tersebut dengan mata pelajaran di sekolahmu.
2. Bagaimanakah perintah-perintah SQL untuk pembuatan tabel pada no 1.
3. Pada kondisi nyata ada hubungan antara mata pelajaran dan Guru. Yaitu Guru mengajarkan mata pelajaran. Buatlah hubungan antara tabel Guru dengan tabel MataPelajaran pada *Database Diagrams*. Jika perlu tambahkan kolom baru atau tabel baru yang bisa menjamin hubungan tersebut benar (Petunjuk: tentukan dulu kardinalitas hubungan apakan *one-to-many* ataukah *many-to-many*)
4. Buatlah *View* dengan perintah SQL untuk pernyataan-pernyataan berikut:
    a. Tampilkan semua data mata pelajaran.
    b. Tampilkan nama mata pelajaran yang diajarkan pada semester 3 ke atas (Petunjuk: gunakan tanda >, <, >= dan <= pada perintah WHERE)
    c. Tampilkan nama pelajaran yang diajarkan kurang dari 3 kali pertemuan.
    d. Tampilkan semua nama pelajaran dan nama guru yang mengasuhnya.
    e. Tampilkan nama pelajaran dan nama guru yang mengasuhnya untuk mata pelajaran yang diajarkan di semester 4.

BAB 13 DESAIN WEB STATIS DAN HTML

Jika kita sering berselancar di dunia internet kita akan sering menjumpai halaman-halaman web seperti pada Gambar 13.1. di samping ini. Halaman web ini ada yang bersifat statis tetapi juga ada yang dinamis. Tetapi apapun sifatnya, halaman-halaman tersebut hampir pasti melibatkan bahasa HTML.

Gambar 13.1. Halaman web.

Bab ini membahas dua standar kompetensi yaitu menerapkan dasar-dasar pembuatan web statis tingkat dasar dan membuat file HTML sesuai spesifikasi. Penggabungan dua kompetensi ini karena kedekatan isi kompetensi dasar. Standar kompetensi membuat file HTML sesuai spesifikasi terdiri dari empat kopetensi dasar yaitu menetapkan pemakaian struktur file, membuat struktur file HTML, memformat file dan menambahkan obyek. Standar kompetensi menerapkan dasar-dasar pembuatan web statis tingkat dasar juga terdiri dari empat kompetensi dasar yaitu menjelaskan konsep dasar dan teknologi web, mempersiapkan pembuatan web, melakukan pembuatan web, dan menampilkan web di *browser*.

Rangkuman diletakkan pada akhir bab dilanjutkan dengan soal-soal latihan yang disusun dari soal-soal yang mudah hingga soal-soal yang sulit. Latihan soal ini digunakan untuk mengukur kemampuan terhadap kompetensi dasar ini. Sebelum mempelajari kompetensi ini ingatlah kembali dasar sistem komputer, sistem operasi, dan algoritma pemrograman dasar.

TUJUAN

Setelah mempelajari bab ini diharapkan kalian akan mampu :

- Menjelaskan konsep dasar dan teknologi web
- Menetapkan pemakaian dan struktur dokumen
- Mempersiapkan pekerjaan pembuatan web
- Melakukan pembuatan dokumen web baru dan menampilkan dalam web *browser*
- Membuat struktur dokumen dengan bahasa HTML
- Memformat dokumen dan menambahkan obyek
- Membuat tabel
- Membuat link antar dokumen

13.1. KONSEP DASAR DAN TEKNOLOGI WEB

World Wide Web secara luas lebih dikenal dengan istilah *Web*. Web pertama kali diperkenalkan pada tahun 1992. Hal ini sebagai hasil usaha pengembangan yang dilakukan CERN di Swiss. Internet dan web adalah dua hal yang berbeda. Internet lebih merupakan perangkat keras, sedangkan web adalah perangkat lunak. Selain itu, protokol yang dipakai oleh keduanya juga berbeda. Internet menggunakan TCP/IP sebagai protokol operasionalnya, sedangkan web menggunakan HTTP (*Hyper Text Transfer Protocol*).

Web disusun dari halaman-halaman yang menggunakan teknologi web dan saling berkaitan satu sama lain. Suatu standar teknologi web saat ini sudah tersusun, meskipun penerapannya belum didukung oleh seluruh pengembang web. Standar ini disusun oleh suatu badan yaitu *World Wide Web Consortium* (W3C). Standar ini dibutuhkan karena semakin banyaknya variasi dalam teknologi web sehingga terkadang satu sama lain tidak kompatibel.

13.1.1 Standar Teknologi Web

Secara umum teknologi disain web terbagi menjadi beberapa layer (lapisan), yaitu structural layer, presentation layer dan behavioral layer.

- **Structural layer**

  Layer ini berhubungan dengan struktur dokumen dokumen web. Bagaimana sebuah dokumen tersusun, format apa yang dipakai, tanda atau mark up apa yang digunakan merupakan bagian dari layer ini. Standar teknologi yang direkomendasikan saat ini adalah *Extensible Hypertext Markup Language* (XHTML) dan *Extensible Markup Language* (XML). XHTML adalah HTML versi terakhir (4.01) yang ditulis ulang dengan dengan aturan-aturan yang lebih ketat mengacu pada XML. Sedangkan XML adalah sekumpulan aturan untuk menyusun bahasa *markup.*

- **Presentation layer**

  Layer ini berhubungan dengan bagaimana mengatur tampilan dokumen pada layar, suara yang keluar, atau bagaimana format pencetakan dokumen. Pada teknologi web lama bagian ini menyatu dengan *structural layer*. Tapi pada standar baru, layer ini disarankan untuk dipisah. Yang termasuk teknologi ini adalah *Cascading Style Sheets* (CSS).

- **Behavioral layer**

  Layer ini berhubungan dengan masalah penggunaan bahasa skrip dan pemrogramannya untuk tujuan meningkatkan sisi interaktif dan dinamis halaman *web*. Yang termasuk dalam layer ini adalah *Document Object Model* (DOM) dan JavaScript. DOM memungkinkan suatu dokumen atau skrip untuk mengakses atau meng-update isi, struktur, dan style dari dokumen. JavaScript merupakan teknologi yang cukup lama dan tetap digunakan untuk menambah dokumen menjadi lebih interaktif.

13.1.2. *Web* Statis dan *Web* Dinamis.

Halaman *web* dapat digolongkan menjadi *web* statis dan *web* dinamis. Pengertian *web* statis dan *web* dinamis seringkali mengundang perdebatan. Sebagian pengguna internet menyatakan jika pada halaman-halaman *web* dilengkapi dengan animasi yang bergerak maka disebut *web* dinamis sedangkan jika halaman-halaman *web* tersebut hanya berisi teks dan gambar yang tidak bergerak maka disebut *web* statis. Namun berdasarkan kesepakatan maka pengertian statis dan dinamis tidak ditentukan oleh ada atau tidaknya animasi bergerak pada halaman-halaman *web*, tetapi ditentukan oleh isi atau informasi yang ada pada halaman-halaman tersebut.

Data dan informasi yang ada pada *web* statis tidak berubah-ubah. Dokumen *web* yang dikirim kepada *client* akan sama isinya dengan apa yang ada di *web server*. Sedangkan *web* dinamis, memiliki data dan informasi yang berbeda-beda tergantung input apa yang disampaikan *client*. Dokumen yang sampai di *client* akan berbeda dengan dokumen yang ada di *web server*.

Contoh paling mudah untuk membedakan *web* statis dan *web* dinamis adalah bila kalian membuka situs Google. Halaman awal adalah statis karena kita tidak melihat perubahan isi atau informasi. Halaman ini baik di komputer *client* maupun di *web server* akan sama. Namun begitu kita memasukkan kata pada textbox yang tersedia kemudian menekan tombol search maka kita sedang berinteraksi dengan *web server* Google. *Web server* akan mengirimkan halaman *web* sesuai yang diminta oleh *client*. Tampilan di sisi *client* akan berupa daftar alamat dan keterangannya. Sedangkan di sisi *server* isi dokumennya adalah serangkain kode-kode untuk mencari apa yang diinputkan *client*.

Bab ini secara khusus akan mempelajari tentang pembuatan *web* statis sedangkan *web* dinamis akan kita bahas di bab 14.

13.2. PERSIAPAN PEMBUATAN *WEB*

Pembuatan halaman *web* membutuhkan persiapan tidak saja pengetahuan tentang bagaimana disain halaman *web*, namun juga perlu dukungan persiapan perangkat keras, perangkat lunak, dan yang lainnya.

13.2.2. Perangkat Keras

Perangkat keras yang dibutuhkan untuk pembuatan halaman *web* tidak berbeda jauh dengan kebutuhan komputasi biasa. Seperangkat komputer lengkap dengan CPU, monitor, keyboard, mouse, printer dan beberapa perangkat tambahan lain sudah dapat digunakan untuk membuat halaman *web*. spesifikasi tergantung dari perangkat lunak yang akan diinstal pada perangkat komputer tersebut. Jika kita menginstal *web server*, pengolah gambar untuk disain halaman *web*, HTML editor yang komplet, tentu kita membutuhkan spesifikasi yang lebih tinggi.

13.2.3. Perangkat Lunak

- **Sistem operasi**

  Sistem operasi memegang peranan penting dalam pembuatan *web* karena pada sistem operasi itu akan ditanamkan (diinstal) *web server*, *web* editor, sistem manajemen basis data dan bahasa pemrograman. Artinya pilihan pada sistem operasi tertentu akan menentukan pula pilihan *web server*, perangkat pengembang dan bahasa pemrograman yang akan digunakan. Hal ini dikarenakan adanya masalah kompatibilitas antar perangkat lunak. Sebagai contoh, apabila kita memilih menggunakan sistem operasi Linux maka kita tidak dapat menginstal IIS sebagai *web server*.

  Selain masalah kompatibilitas, hal lain yang juga perlu dipertimbangkan dalam penentuan sistem operasi yang akan kita gunakan pada *server* adalah masalah keamanan, stabilitas, kemudahan konfigurasi. Keamanan berhubungan dengan kemampuan sistem operasi untuk melindungi diri dari serangan virus, spam, atau kode-kode jahat yang sengaja disusupkan. Kemampuan ini sangat penting diperhatikan karena lalu lintas data dalam internet sangat rentan terhadap gangguan virus, spam, dan pengganggu lainnya. Stabilitas berhubungan dengan kemampuan sistem operasi untuk bekerja terus-menerus untuk merespon permintaan *client*. Kemudahan konfigurasi berhubungan mudah tidaknya konfigurasi dilakukan terhadap sistem operasi dalam perannya sebagai *server*.

- ***Web Server***

  Web server adalah perangkat lunak yang bertindak melayani permintaan-permintaan *client* terhadap halaman-halaman *web* tertentu. Ada beberapa nama yang cukup populer dalam dunia *web server*. Diantaranya adalah Apache dan IIS (Internet Information Service). Sampai dengan Desember

2007, wikipedia mencatat Apache berada dalam posisi pertama sebagai *web server* yang paling banyak digunakan, disusul IIS.

Apache dapat digunakan baik untuk *web* statis maupun *web* dinamis dan mendukung banyak platform sistem operasi dan bahasa pemrograman *server*, antara lain Perl, Phyton, Java (JSP dengan menggunakan Tomcat) dan tentu saja PHP. Fungsi-fungsi keamanan *web* juga dikendalikan dengan sangat baik. Dukungan pada koneksi dengan berbagai basis data, seperti MySQL, SQL Lite, PosgreSQL, Oracle, DB2 dan lain-lain dapat dilakukan dengan baik. Gambar 13.2 menunjukkan bagaimana *web server* Apache (httpd) digunakan di Linux.

Gambar 13.2. Menjalankan service Apache (httpd) pada Linux.

IIS adalah *web server* keluaran Microsoft. Sebutan *web server* bagi IIS mungkin tidak terlalu tepat, karena selain *web server*, IIS juga memberikan fasilitas *file server, email server* dan layanan lain berbasis internet. Oleh karena itu istilah yang tepat mungkin adalah *internet based-service*. Perangkat lunak ini dibundel dalam sistem operasi Microsoft Windows. Namun secara default tidak langsung diinstall. Sehingga kalau kita mau menggunakannya kita harus menginstall lebih dulu. Gambar 13.3 menunjukkan bagaimana IIS diinstall.

Gambar 13.3. Memeriksa dan menginstal IIS.

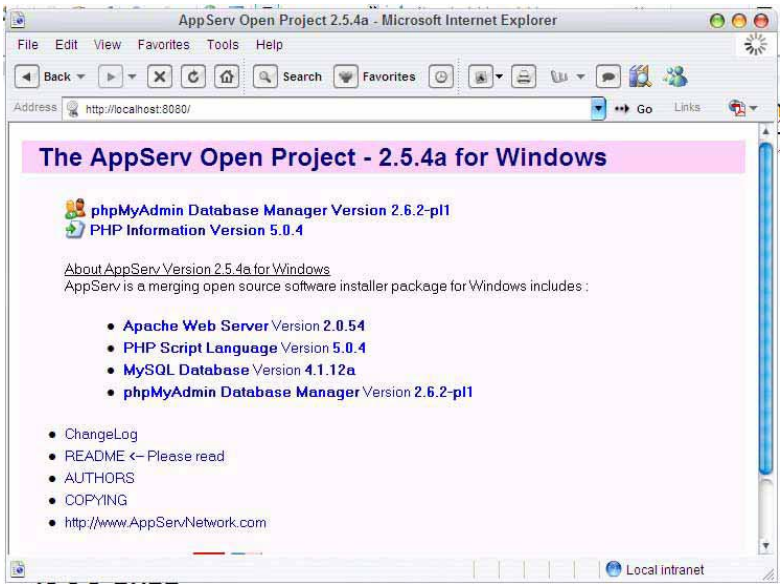
- ***Web* / HTML Editor**

  Web / HTML Editor adalah perangkat lunak yang digunakan untuk membuat halaman-halaman *web*, baik yang bersifat statis maupun dinamis. Di pasar perangkat lunak, saat ini tersedia banyak sekali jenis perangkat pengembang *web*, mulai dari yang sederhana sampai yang canggih dan kompleks. Namun sebenarnya untuk membuat halaman *web* baik statis maupun dinamis kita dapat menggunakan teks editor biasa seperti Notepad atau Vi. Hanya saja teks editor tidak menyediakan fasilitas-fasilitas yang memudahkan kita dalam membuat halaman *web*. Pada perangkat pengembang *web* yang lebih kompleks seperti Adobe

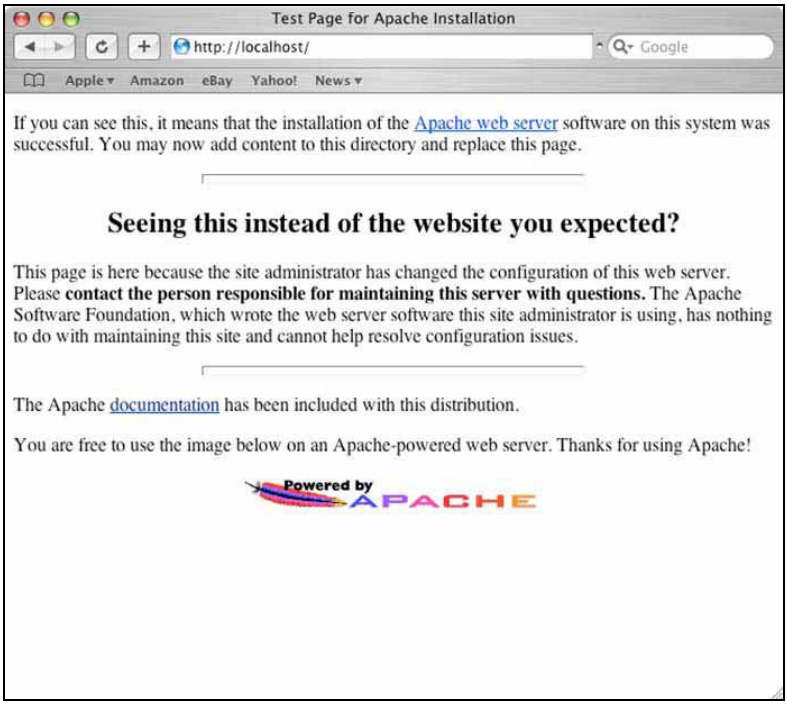
Dreamweaver (dulu Macromedia Dreamweaver), Microsoft Visual Studio .Net, dan beberapa yang lainnya, kita akan mendapati fasilitas yang sangat membantu mempercepat pembuatan halaman *web*, antara lain: tampilan berbasis GUI, *automatic code completion* (melengkapi kode secara otomatis), WYSIWYG (*What You See Is What You Get*) HTML Editor, koneksi ke basis data yang lebih mudah, dan banyak lagi fasilitas. Tentu saja perangkat pengembang ini berharga relative mahal. Pada bagian lain dari bab ini kalian akan diajak untuk mengenali sedikit beberapa perangkat lunak ini.

- ***Web Browser***

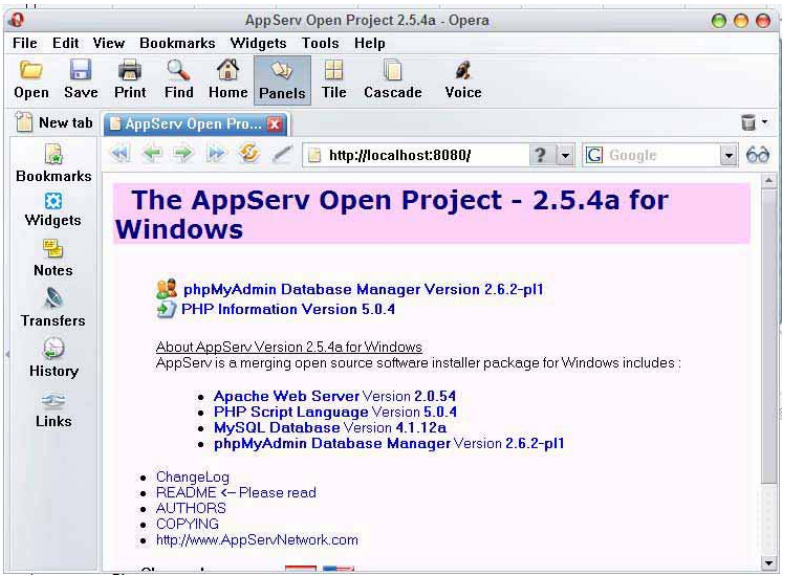
  Web browser berfungsi menerjemahkan kode-kode HTML menjadi tampilan yang kita kehendaki. Ada banyak *Web Browser* tersedia di internet. Hampir semuanya dapat kita *download* secara bebas. Beberapa nama yang cukup terkenal antara lain Microsoft Internet Explorer, Firefox, Opera atau Safari. Microsoft Internet Explorer adalah default *web browser* pada sistem operasi Microsoft Windows (lihat Gambar 13.4). Firefox adalah default *web browser* pada sebagian besar sistem operasi Linux. Safari adalah default *web browser* pada sistem operasi Mac OS X (lihat Gambar 13.5). Sedangkan Opera adalah *web browser* keluaran Opera Software yang dapat berjalan pada berbagai *platform* sistem operasi (Gambar 13.6).



Gambar 13.4. Microsoft Internet Explorer.



Gambar 13.5. Safari.



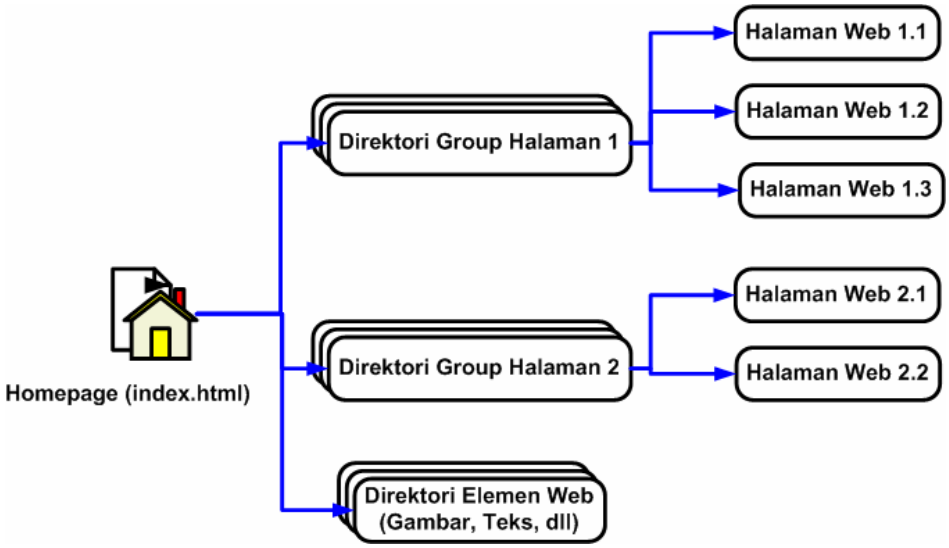
Gambar 13.6. Opera.

13.3. MEMBUAT DAN MENGUJI HALAMAN *WEB*

Ada dua model dalam pembuatan halaman *web* statis. Yang pertama adalah membuat halaman-halaman tersebut pada komputer lokal, kemudian setelah berhasil dipindahkan ke lokasi di *web server*. Model kedua adalah langsung membuat halaman-halaman *web* di lokasi *web server*. Lokasi *web server* dapat berada di satu komputer yang sama dengan tempat pembuatan halaman *web* atau di komputer lain yang berperan sebagai *server*. Cara pertama lebih mudah dilakukan karena tidak membutuhkan kerja *web server* di tahap disain. Pada bagian ini kita akan mencoba cara ini.

Buatlah direktori pada komputer kalian. Jika kalian menggunakan sistem operasi Windows, kalian tinggal buka Windows Explorer kemudian klik kanan pada area Windows eksplorer dan pilih New kemudian pilih Folder. Beri nama misalkan Latihan*Web*. Jika kalian menggunakan Linux, buka file manager yang tersedia misalnya Konqueror, Nautilus, atau Thunar. Lakukan cara yang sama seperti pada Windows Explorer.

Umumnya suatu situs *web* tidak hanya berisi satu halaman *web* tetapi kumpulan dari beberapa halaman *web* yang saling berkait. Biasanya pengembang *web* membuat struktur direktori untuk mempermudah pengaturan halaman. Perhatikan struktur direktori sebuah situs *web* pada Gambar 13.7.



Gambar 13.7. Contoh Struktur direktori situs *web*.

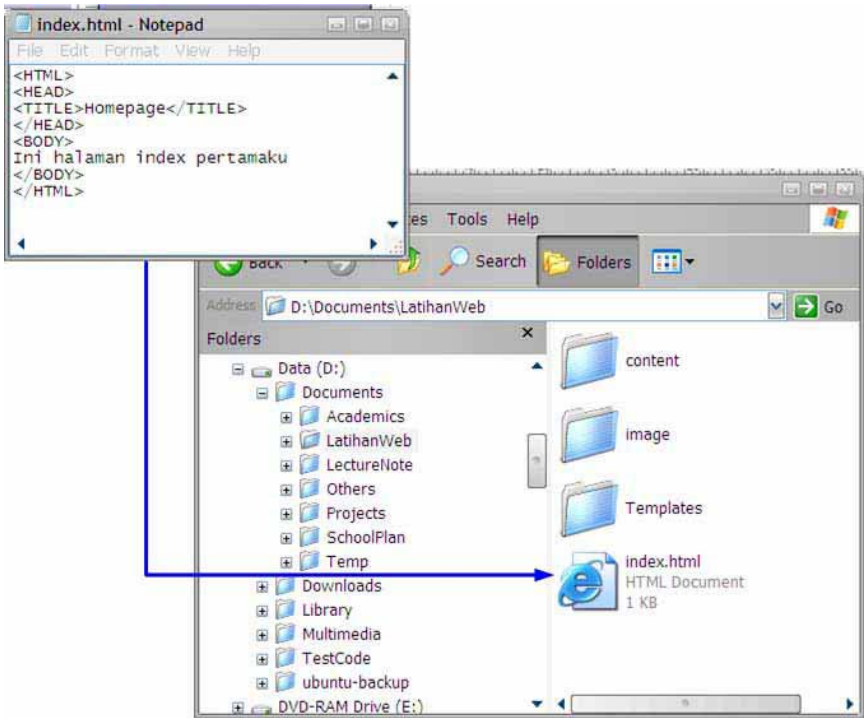
Halaman awal suatu situs *web* biasanya berupa halaman *web* yang diberi nama homepage. Biasanya filenya diberi nama index.html (atau bisa juga index.php, index.jsp, index.asp jika menggunakan bahasa skrip *server*). Di dalam direktori yang sama dengan index.html ini biasanya ada direktori-direktori lain yang berisi halaman-halaman yang dikelompokkan berdasarkan kedekatan

tema atau berdasarkan pengelompokkan lain. Selain itu ada direktori lain yang digunakan untuk menyimpan elemen-elemen yang digunakan dalam halaman *web*. biasanya berisi file-file gambar, teks, audio, video atau elemen-elemen yang lain.

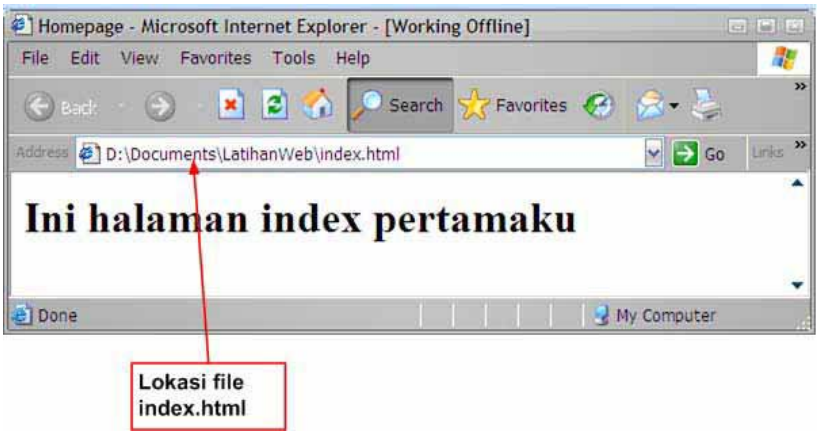
Buatlah struktur direktori seperti di atas dengan cara menambahkan sub direktori di bawah direktori Latihan*Web* yang telah kalian buat. Beri nama sesuai keinginan kalian, tetapi usahakan nama direktori menunjukkan apa isi dari direktori tersebut. Buka teks editor, misalkan Notepad kemudian ketikkan kode berikut ini.

```
<HTML>
<HEAD>
<TITLE>Homepage</TITLE>
</HEAD>
<BODY>
Ini halaman index pertamaku
</BODY>
</HTML>
```

Simpan dengan nama index.html dan letakkan di direktori Latihan*Web* (Gambar 13.8). Untuk menguji file tersebut, klik ganda pada file index.html. kalian akan mendapatkan tampilan seperti pada Gambar 13.9.



Gambar 13.8. File index.html dan lokasi penyimpanannya.



Gambar 13.9. Hasil pengujian file index.html.

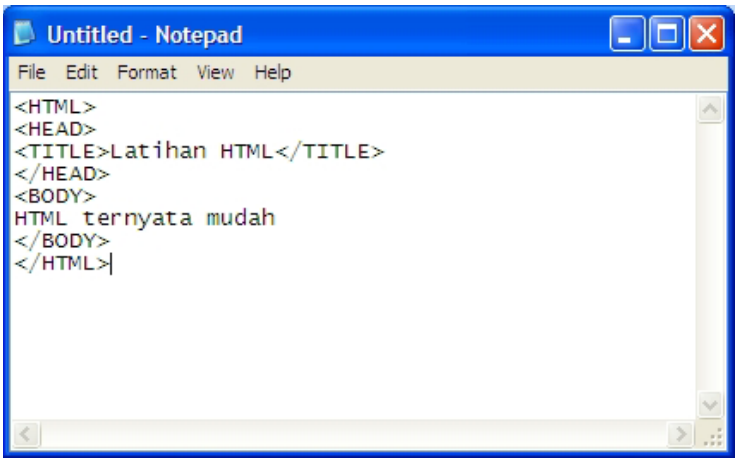
13.4. HTML

13.4.1.Pengertian HTML

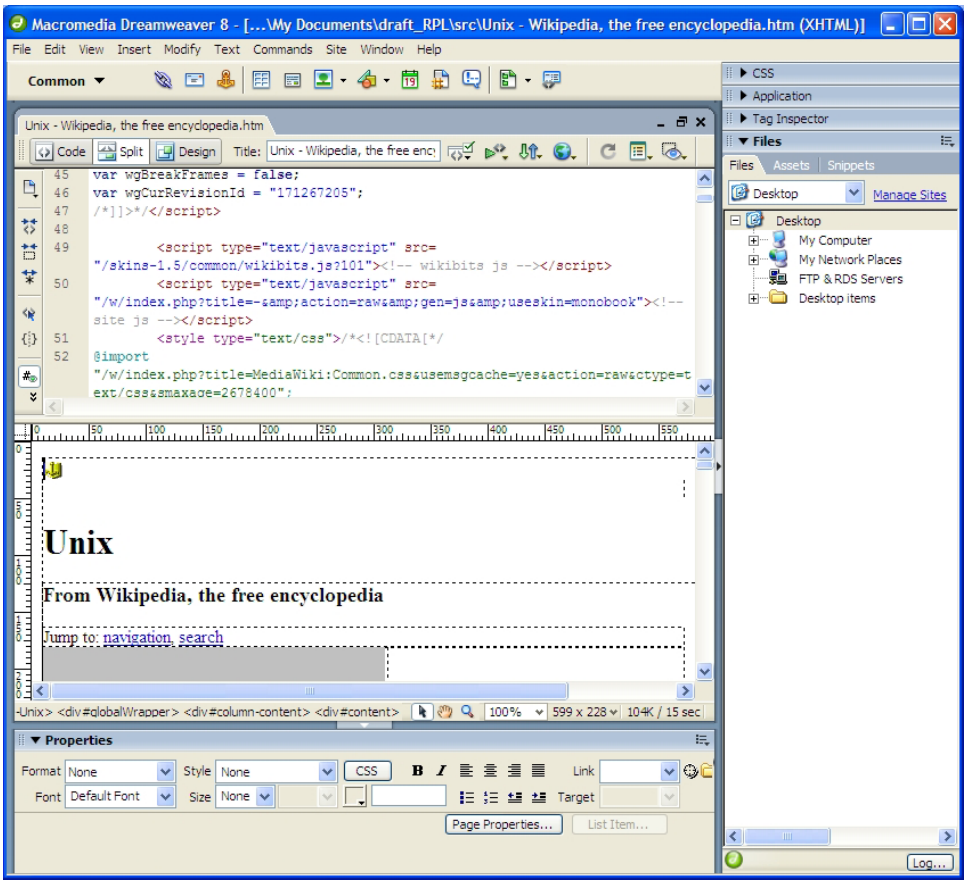
Gambar 13.8 dan 13.9 menunjukkan pada kalian bagaimana membuat halaman *web* sederhana. Halaman *web* yang kalian buat ini menggunakan bahasa yang disebut HTML *(Hypertext Markup Language)*. HTML merupakan pengembangan dari standar pemformatan dokumen teks yaitu *Standard Generalized Markup Language (SGML*). HTML sebenarnya adalah dokumen ASCII atau teks biasa yang dirancang untuk tidak tergantung pada suatu sistem operasi tertentu.

HTML dibuat oleh Tim Barners-Lee ketika masih bekerja untuk CERN. HTML dipopulerkan pertama kali oleh *browser Mosaic*. Selama awal tahun 90'an, HTML mengalami perkembangan yang sangat pesat. Setiap pengembangan HTML pasti akan menambahkan kemampuan dan fasilitas yang lebih baik daripada versi sebelumnya. Perkembangan yang pesat tersebut tidak sampai merubah cara kerja HTML.

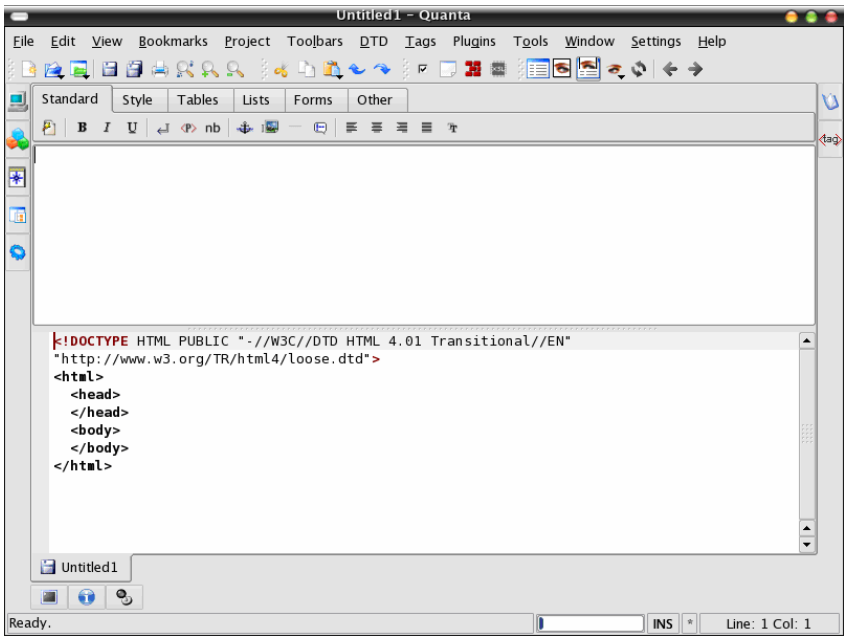
Sebuah dokumen atau file HTML agar dapat dibaca langsung oleh *browser* disimpan dalam ekstensi .htm atau .html. Untuk menulis HTML tidak dibutuhkan perangkat lunak yang spesifik, cukup dengan text editor sederhana seperti *Notepad* (pada Microsoft Windows) atau beragam text editor yang ada di platform Linux dan Apple Mac OS, seperti *vi, nano, joe, gedit, leafpad* dan lain-lain. Beberapa editor menyediakan fitur-fitur tambahan seperti *syntax coloring* (memberi warna pada kode-kode HTML) dan *code completion* (melengkapi secara otomatis kode yang akan dituliskan). Saat ini telah banyak perangkat lunak berbasis GUI yang sangat membantu dalam pembuatan halaman-halaman HTML. *Macromedia Dreamweaver* dan *Microsoft Frontpage* merupakan dua nama yang cukup populer di platform Microsoft Windows. Sedangkan di Linux tersedia *Quanta+*, Bluefish dan *Nvu*.



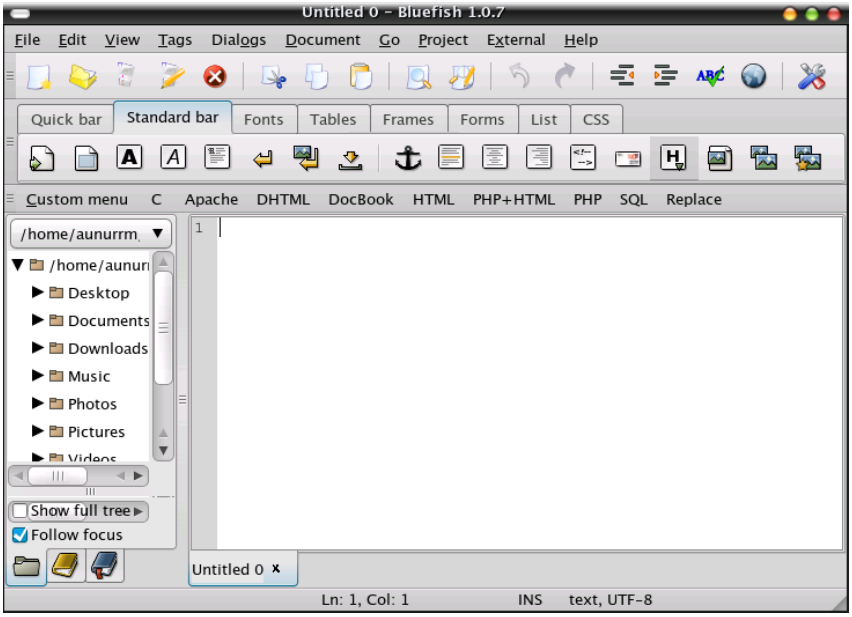
Gambar 13.10. Teks editor Notepad.



Gambar 13.11. Macromedia Dreamweaver.



Gambar 13.12. Quanta pada sistem operasi Linux



Gambar 13.13. Bluefish pada sistem operasi Linux

13.4.2. Struktur Umum File dengan Bahasa HTML

HTML adalah bahasa yang disisipkan (*embedded language*) pada dokumen dengan memberi tanda tertentu yang disebut **tag**. Tag merupakan aturan penulisan kode yang ditulis dengan diawali tanda lebih kecil dan di akhiri dengan tanda lebih besar (**<tag>**). *Browser* akan menentukan tampilan teks atau dokumen berdasarkan tag yang digunakan.

Sintaks penulisan tag mengikuti aturan-aturan umum berikut ini:

a) Setiap tag mempunyai nama yang spesifik. Kadang-kadang diikuti opsi-opsi yang disebut atribut. Baik nama maupun opsi harus berada dalam tanda <..>.

Contoh:

```
<a href="/wiki/PHP" title="PHP">PHP</a>
```

Pada contoh ini tagnya memiliki nama **<a>** sedangkan atribut untuk tag **<a>** adalah **href** dan **title**. Sehingga baik nama tag dan atributnya harus berada di dalam tanda **<…>** seperti pada contoh.

b) Sebagian besar tag berpasangan. Penulisan untuk tag yang berpasangan adalah sebagai berikut : **<namatag>….</namatag>**

Contoh:

```
<TITLE>Paragraf</TITLE>
<strong>Cetak Tebal</strong>
```

Pada tag yang berpasangan seperti pada contoh ini, **<TITLE>** adalah tag awal dan **</TITLE>** adalah tag akhir. Perhatikan tanda / pada tag akhir.

c) Nama tag dan atribut-nya tidak bersifat case sensitive. Penulisan **Cetak Tebal** memberikan hasil yang sama dengan **<STRONG>Cetak Tebal</STRONG>**.

d) Penulisan atribut suatu tag diletakkan setelah nama tag. Jika ada lebih dari satu atribut maka digunakan spasi untuk memisahkan. Urutan atribut tidak penting.

Contoh:

```
<FONT SIZE=3>Teks Baru</FONT>
<FONT SIZE=5 FACE="verdana">Teks Baru </FONT>
```

e) Nilai dari atribut ditulis setelah tanda sama dengan (=). Pada contoh sebelumnya (lihat bagian d) terlihat bahwa atribut **SIZE** dari tag **FONT** memiliki nilai **5** sedangkan atribut **FACE** memiliki nilai **"verdana".**

f) Jika nilai dari atribut hanya tunggal, maka kita langsung menuliskan setelah tanda =. Jika lebih dari satu maka dapat digunakan tanda '... ' atau "...". Pada contoh bagian d, tampak bahwa penulisan nilai untuk atribut **SIZE** tanpa menggunakan tanda **".."**, sedangkan pada atribut **FACE** tanda **".."** untuk menandai kata **verdana**.

Dokumen HTML secara umum akan terdiri dari dua bagian yaitu Header dan Body (Gambar 13.14)

```
<html>
<head>
<meta http-equiv="Content-Type"
content="text/html; charset=iso-8859-1" /
>
<title>Untitled Document</title>
</head>

<body>
<strong>Cetak Tebal</strong>
</body>
</html>
```

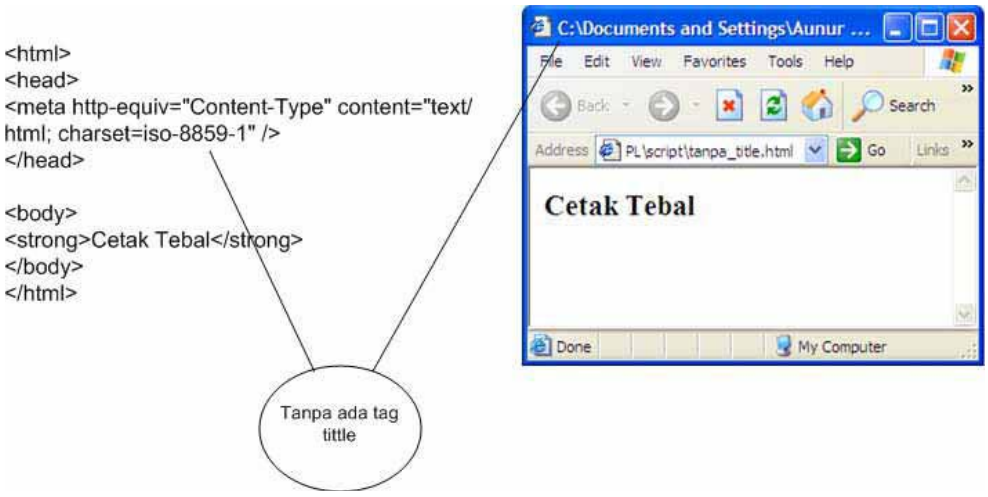
Header

Body

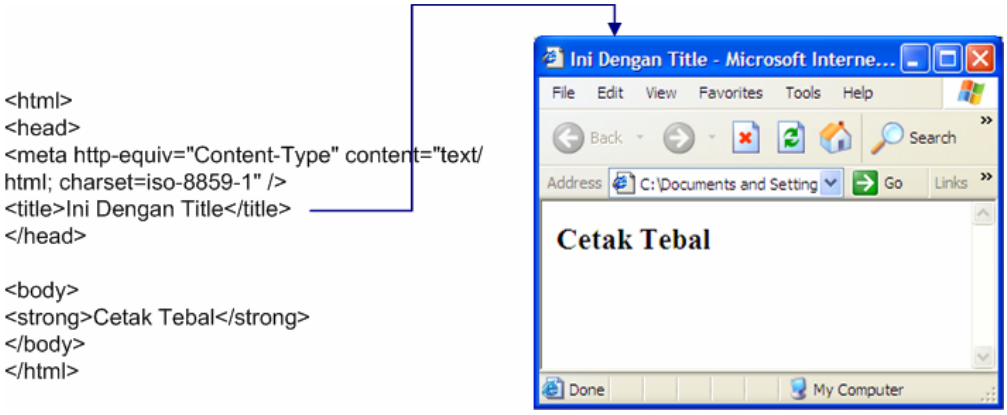
Gambar 13.14. Struktur umum dokumen HTML.

- **Header**

Bagian ini biasanya berisi berbagai macam keterangan tentang dokumen termasuk title (judul dokumen), posisinya dalam sekumpulan halaman *web* dan hubungannya dengan dokumen lain. Bagian ini ditandai dengan tag **<head> …. </head>**. Tag ini tidak mempunyai atribut. Di dalam tag ini kita dapat meletakkan beberapa tag lain seperti tag title dan tag link. Lihat Gambar 13.15 dan 13.16.



Gambar 13.15. Header dokumen HTML tanpa tag title.



Gambar 13.16. Header dokumen HTML dengan tag title.

- **Body**

Body adalah bagian dari dokumen HTML tempat dimana kita meletakkan isi dari dokumen. Bagian ini ditandai dengan tag **<body>** dan diakhiri dengan **</body>**. Apapun yang berada diantara dua tanda ini disebut sebagai *body content*. Dokumen HTML yang paling sederhana mungkin hanya berisi sebaris atau dua baris teks saja tanpa format apapun (Gambar 13.17). Pada dokumen yang lebih kompleks, *body content* bisa berisi teks yang terformat, gambar, tabel atau bahkan animasi yang rumit (Gambar 13.18).

Gambar 13.17. Dokumen HTML dengan *body content* sederhana.

```
<BODY>
<p class="style1">HTML ternyata mudah</p>
<p>Bisa menampilkan gambar</p>
<p class="style1"><img src="../picture/150px-
NewTux_Layer 1.jpg" alt="tux" width="150"
height="180"> </p>
</BODY>
```

Gambar 13.18. Dokumen HTML dengan *body content* yang lebih kompleks.

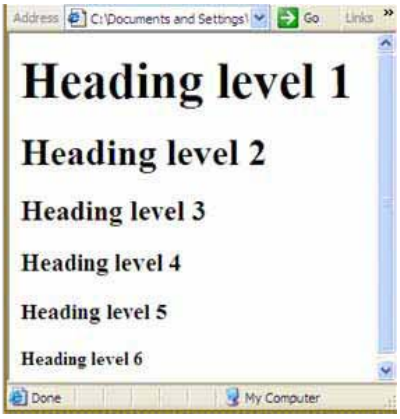
13.4.3.Format Dokumen

Ada banyak sekali tag HTML yang tersedia, baik itu berhubungan dengan deskripsi dokumen atau yang berhubungan dengan format tampilan dokumen. Tidak semuanya akan dibahas dalam buku ini. Pada bagian berikut ini akan dibahas beberapa tag yang biasa dipakai dalam format tampilan dokumen.

- **Heading**

Heading adalah sekumpulan kata yang menjadi judul atau subjudul dalam sebuah dokumen HTML. *Heading* berbeda dengan tag **<TITLE>**. HTML menyediakan enam tingkatan *heading*, dimana *heading* level 1 (**<H1>**) adalah yang terbesar dan *heading* level 6 (**<H6>**) adalah yang terkecil. Gambar 13.19 menunjukkan bagaimana penggunaan *heading*.

```
<HTML>
<HEAD>
        <TITLE>Heading</TITLE>
</HEAD>
<BODY>
        <H1>Heading level 1</H1>
        <H2>Heading level 2</H2>
        <H3>Heading level 3</H3>
        <H4>Heading level 4</H4>
        <H5>Heading level 5</H5>
        <H6>Heading level 6</H6>
</BODY>
</HTML>
```

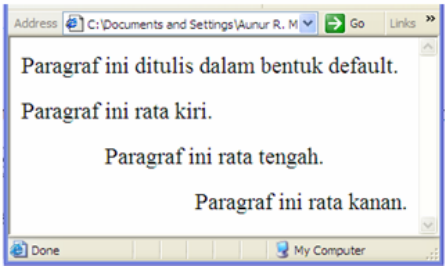


Gambar 13.19. Penggunaan heading.

- **Paragraph**

Paragraf dalam HTML dibuat dengan tag **<P>**. Tag ini akan membuat baris baru dengan menyisipi satu baris kosong. Penulisan isi paragraf diapit oleh **<P>** dan **</P>**. Pengaturan posisi paragraf dapat dilakukan dengan atribut **ALIGN** yang diikuti dengan posisi yang diinginkan, yaitu **left** untuk rata kiri, **center** untuk rata tengah horisontal dan **right** untuk rata kanan (Lihat Gambar 13.20).

```
<HTML>
<HEAD>
<TITLE>Paragraf</TITLE>
</HEAD>
<BODY>
<P>Paragraf ini ditulis dalam bentuk default.</P>
<P ALIGN=left>Paragraf ini rata kiri.</P>
<P ALIGN=center>Paragraf ini rata tengah.</P>
<P ALIGN=right>Paragraf ini rata kanan.</P>
</BODY>
</HTML>
```

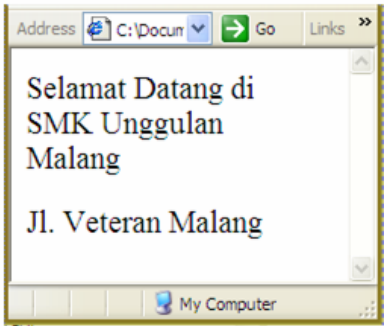


Gambar 13.20. Penggunaan paragraph.

- **Line Break**

Line break digunakan untuk menuliskan teks pada baris berikutnya. Line break dibuat dengan tag tunggal **
. Tag ini akan membuat baris baru tanpa memberi baris kosong sebagaimana pada tag **<P> (Lihat Gambar 13.21).

```
<HTML>
<HEAD>
        <TITLE>Line Break</TITLE>
</HEAD>
<BODY>
Selamat Datang di
<BR>SMK Unggulan
<BR>Malang
<P>Jl. Veteran Malang
</BODY>
</HTML>
```



Gambar 13.21. Tag **
** dan **<P>.**

- **List**

  HTML menyediakan 3 cara untuk membuat daftar atau list, yaitu

  o *Ordered list*

  Ordered list digunakan untuk membuat daftar dimana setiap bagian memiliki nomor secara berurutan. *Ordered list* dimulai dengan tag **<OL>** dan diakhiri dengan tag **</OL>**, sedangkan setiap bagiannya digunakan tag **<LI>** tanpa tag penutup. Tag ini menggunakan angka sebagai urutan secara default, tapi bisa dirubah dengan merubah nilai atribut **TYPE**. Nilai-nilai atribut yang diijinkan dapat dilihat pada Tabel 13.1.

Tabel 13.1. Daftar atribut TYPE untuk *Ordered list* dan *Unordered list*.

Jenis List	Atribut	Fungsi
Ordered list	**TYPE=1**	Daftar berurutan dengan angka arab (1,2,3,...)
	TYPE=I	Daftar berurutan dengan angka romawi besar (I,II,III,...)
	TYPE=i	Daftar berurutan dengan angka romawi kecil (i,ii,iii,...)
	TYPE=A	Daftar berurutan dengan abjad besar (A,B,C,...)
	TYPE=a	Daftar berurutan dengan abjad kecil (a,b,c,...)
Unordered List	**TYPE=circle**	Daftar dengan tanda lingkaran
	TYPE=square	Daftar dengan tanda tanda kotak
	TYPE=disk	Daftar dengan tanda cakram

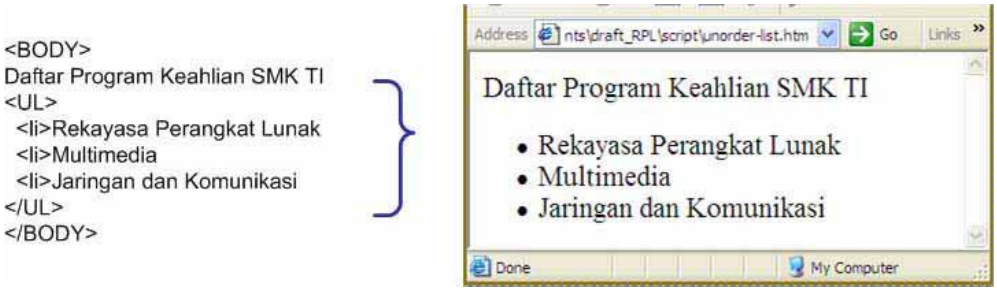
Contoh penggunaan *Ordered list* dapat dilihat pada Gambar 13.22.

```
<BODY>
Daftar Program Keahlian SMK TI
<OL>
  <li>Rekayasa Perangkat Lunak
  <li>Multimedia
  <li>Jaringan dan Komunikasi
</OL>

</BODY>
```

Gambar 13.22. Penggunaan *Ordered List.*

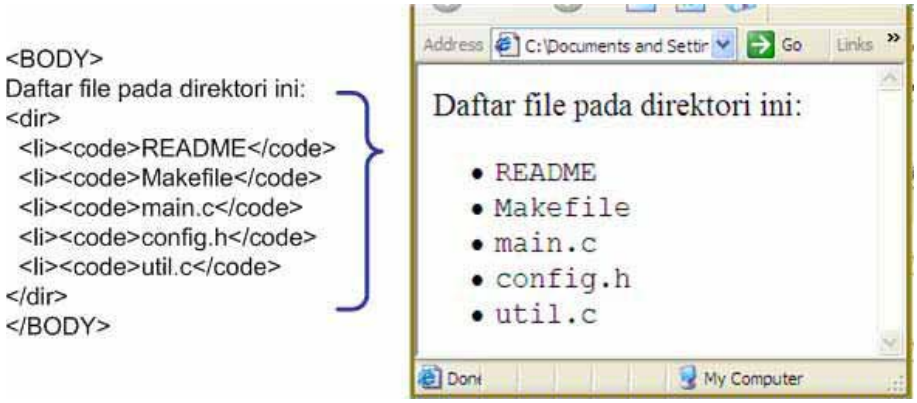
o *Unordered list*

Unordered list digunakan untuk membuat daftar yang disajikan tanpa nomor urut, melainkan dengan secara default dengan tanda bulatan utuh (bullet). Tanda ini bisa dirubah dengan merubah nilai atribut **TYPE** untuk unordered list. Nilai-nilai atribut yang diijinkan dapat dilihat pada tabel 13.1. *Unordered list* dimulai dengan tag **<UL>** dan diakhiri dengan tag **</UL>**, sedangkan setiap bagiannya digunakan tag **<LI>** tanpa tag penutup (lihat contoh pada Gambar 13.23).

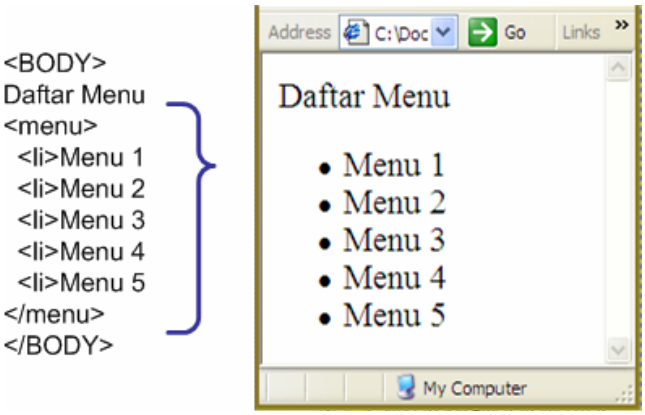


Gambar 13.23. Penggunaan *Unordered List.*

Tipe lain dari list yang termasuk Unordered list adalah *Directory list* dan *Menu list*. *Directory list* merupakan daftar tak bernomor yang digunakan untuk menangani direktori. Tag yang digunakan adalah **<DIR> ... </DIR>**. *Menu list* umumnya digunakan untuk menu pilihan. Tag yang digunakan adalah **<MENU> ... </MENU>**. Baik *Direktori list* maupun *Menu list* menghasilkan tampilan yang sama dengan *Unordered list* (Gambar 13.24 dan Gambar 13.24).



Gambar 13.24. Penggunaan *Direktori List*



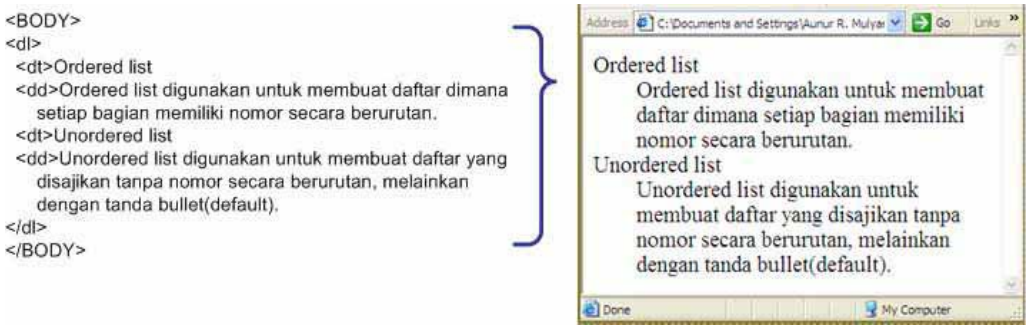
Gambar 13.25. Penggunaan *Menu List*

o *Definition list*

Definition list membuat daftar definisi mirip seperti tampilan pada kamus, dengan definisi suatu istilah agak menjorok ke kanan. Tiga buah pasang tag yang terkait dengan definition list adalah:

- **<DL> ... </DL>** untuk menyatakan tempat bagi daftar definisi.
- **<DT> ... </DT>** untuk menyatakan tempat bagi istilah yang akan didefinisikan.
- **<DD> ... </DD>** untuk menyatakan tempat bagi definisi dari istilah.

Contoh penggunaan *Definition list* dapat dilihat pada Gambar 13.26.



Gambar 13.26. Penggunaan *Definition List.*

- **Font**

HTML menyediakan fasilitas pengaturan huruf yang akan ditampilkan dalam dokumen. Pengaturan ini dilakukan dengan tag berpasangan **<FONT>** dan **</FONT>**. Tag ini memiliki beberapa atribut untuk mengatur ukuran, jenis dan warna huruf yang digunakan.

o Atribut **SIZE** untuk mengatur ukuran huruf, dimana nilai 1 untuk huruf terkecil dan nilai 7 untuk huruf terbesar.

o Atribut **FACE** untuk mengatur jenis huruf yang diinginkan, dimana nilainya berupa *string* nama font seperti Arial, Tahoma dan sebagainya.

o Atribut **COLOR** untuk mengatur warna teks yang dikehendaki, dimana nilainya dapat diisi dengan dua cara dengan menyebut nama warna dalam bahasa Inggris seperti *red, blue* dan *green* atau dengan menggunakan nilai RGB (*Red Green Blue*) seperti **FF000** untuk merah.

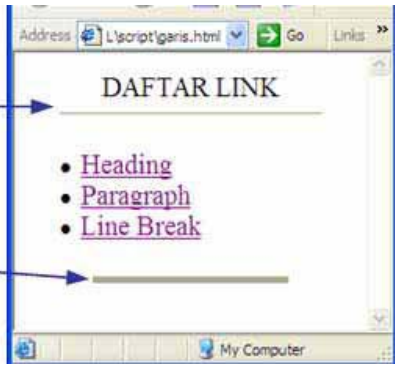
Gambar 13.27. Penggunaan tag Font.

13.4.4. Penambahan Obyek

- **Horizontal Line**

Untuk mempercantik tampilan halaman HTML, kita dapat menambahkan garis horizontal dengan tag **<HR>**. Tag **<HR>** mempunyai atribut **SIZE** untuk menentukan ketebalan garis, atribut **WIDTH** untuk menentukan lebar garis, Atribut **ALIGN** untuk menentukan letak teks dalam garis, dan atribut **NOSHADE** untuk mengatur agar garis tidak disertai bayangan. Gambar 13.28 menunjukkan bagaimana tag **<HR>** digunakan.

```
<BODY>
<div align="center">DAFTAR LINK</div>
<hr align="center" width="80%" size="2">
<menu>
<li><a href="heading.htm" title="Heading">Heading</a>
<li><a href="paragraf.htm" title="Paragraph">Paragraph</a>
<li><a href="linebreak.htm" title="Line Break">Line Break</a>
</menu>
<hr align="center" width="60%" size="5" noshade>
</BODY>
```



Gambar 13.28. Penggunaan garis.

- **Image**

Dokumen HTML dapat diperindah dengan menyertakan gambar pada halaman *web* yang dibuat. Tag **<IMG>** dapat digunakan untuk memanggil dan menampilkan gambar pada halaman *web*. Sintaks penulisan tag **<IMG>** adalah:

<IMG SRC="file_gambar" ALT="nama_alternatif">

Atribut **SRC** digunakan untuk menentukan sumber file gambar yang akan ditampilkan. Atribut **ALT** berfungsi untuk memberi tulisan pengganti, apabila gambar tidak ditampilkan.

```
<BODY>
<div align="center">DAFTAR GAMBAR</div>
<hr align="center" width="80%" size="2">
<ul>
<li>Logo Linux <img src="../picture/150px-NewTux_Layer 1.jpg"
alt="linux logo" width="99" height="125">
<li>Logo Windows <img src="../picture/225px-
Windows_logo_Layer 1.jpg" alt="windows logo">
</ul>
<hr align="center" width="60%" size="5" noshade>
</BODY>
```

Gambar 13.29. Penggunaan tag image.

Untuk pengaturan gambar yang lebih baik, tag IMG menyediakan beberapa atribut, antara lain:

- Atribut **ALIGN** untuk mengatur penempatan teks pada gambar.
- Atribut **BORDER** untuk memberi bingkai pada gambar.
- Atribut **HEIGHT** dan **WIDTH** untuk mengatur tinggi dan lebar gambar.

Contoh berikut ini memperlihatkan penggunaan atribut-atribut tersebut.

```
<BODY>
<div align="center">DAFTAR
GAMBAR</div>
<hr align="center" width="80%"
size="2">
<ul>
<li>Logo Linux <img src="../picture/
150px-NewTux_Layer 1.jpg"
alt="linux logo" width="99"
height="125" align="middle">
<li>Logo Windows <img src="../
picture/225px-Windows_logo_Layer
1.jpg" alt="windows logo" border="2"
align="top">
</ul>
<hr align="center" width="60%"
size="5" noshade>
</BODY>
```

Gambar 13.30. Penggunaan atribut-atribut tag IMG.

Pada Gambar 13.30 terlihat perbedaan tampilan dibandingkan dengan Gambar 13.29. Tulisan Logo Linux terletak ditengah gambar karena kita menggunakan atribut **`align`** dengan nilai **`middle`**. Sedangkan tulisan logo Windows terletak di bagian atas karena kita menggunakan **`align`** dengan nilai **`top`**. Gambar logo windows diberi garis bingkai dengan menggunakan atribut **`border`** dengan nilai 2.

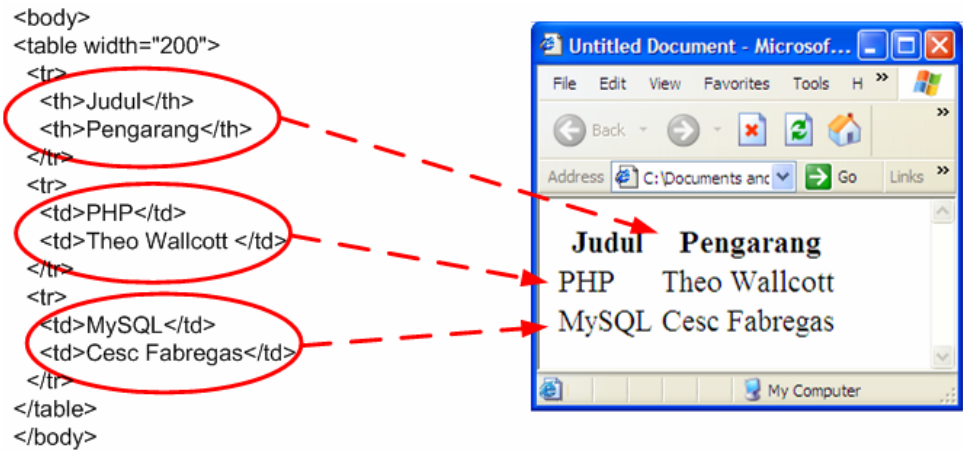
13.4.5.Tabel

Tabel dalam HTML dibuat dengan menggunakan tag awal **`<TABEL>`** dan tag penutup **`</TABLE>`**. Tag ini memiliki beberapa bagian penting, seperti dapat dilihat pada Tabel berikut ini.

Tabel 13.2. Bagian-bagian pada tag Table

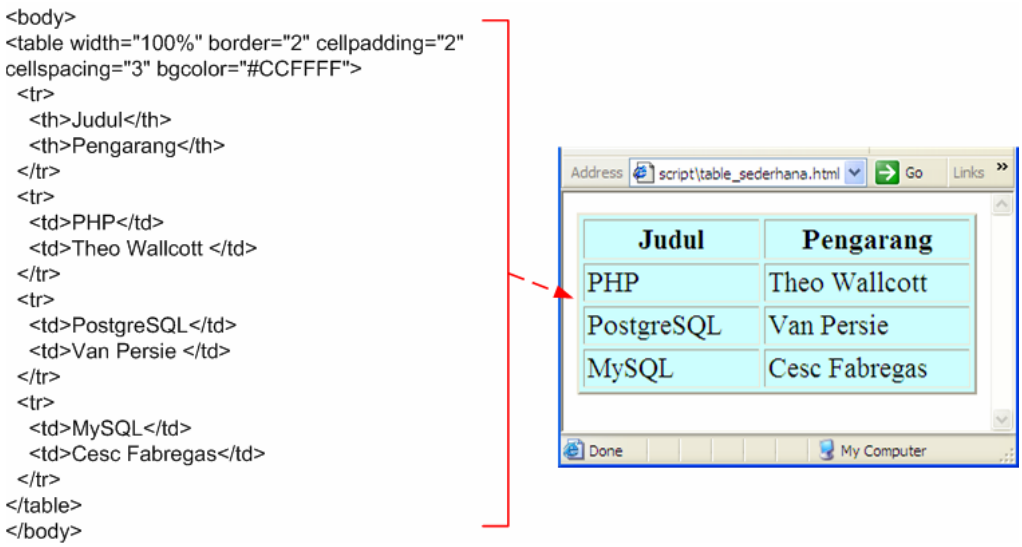
Tag	Fungsi
`<CAPTION>…</CAPTION>`	Membentuk judul tabel
`<TH>…</TH>`	Membuat judul kolom
`<TR>…</TR>`	Membentuk baris pada suatu tabel
`<TD>...</TD>`	Membuat sebuah sel data

Contoh-contoh penggunaan tabel adalah sebagai berikut:



Gambar 13.31. Tabel sederhana.

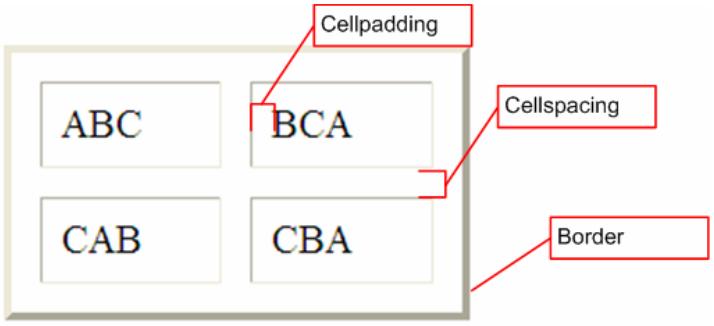
Pada Gambar 13.31, tabel yang kita buat adalah tabel sederhana dengan dua buah kolom dan 3 buah baris (perhatikan ada 3 buah pasangan tag **<TR> … </TR>.** Secara default tabel ditampilkan tanpa ada garis pada tabel tersebut. Kita dapat menambahkan garis dengan menggunakan atribut border pada tabel (lihat Gambar 13.32).



Gambar 13.32. Tabel dengan format yang lebih kompleks.

Pada Gambar 13.32, terlihat tabel yang tampilannya lebih baik daripada Gambar sebelumnya. Ada beberapa atribut yang kita tambahkan pada tabel yaitu :

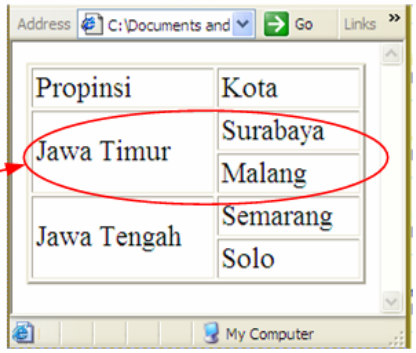
- Atribut **WIDTH** untuk mengatur lebar tabel pada halaman. Kita dapat menggunakan satuan persen (%) atau pixel (px).
- Atribut **BORDER** untuk memberikan garis pada tabel. Nilai untuk atribut ini dari dimulai dari 0 yang berarti tidak ada garis. Semakin besar angka semakin tebal garis.
- Atribut **BGCOLOR** untuk menambahkan warna latar belakang pada tabel.
- Atribut **CELLPADDING** untuk menentukan jarak antara teks dan tepi kiri sebuah sel (lihat gambar 13.33 untuk lebih jelasnya).
- Atribut **CELLSPACING** untuk menentukan jarak bagian sel terhadap tepi dalam bingkai tabel (lihat gambar 13.33 untuk lebih jelasnya).



Gambar 13.33. Cellpadding, cellspacing dan border.

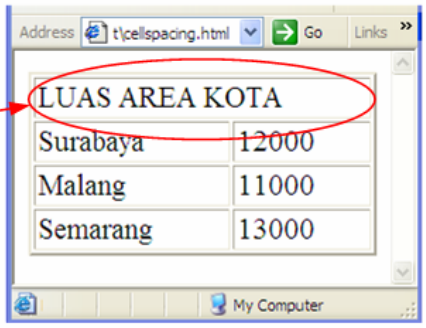
Seperti halnya perangkat lunak *word-processor*, pada HTML kita dapat menggabungkan dua atau lebih sel menjadi satu buah sel. Untuk menggabung baris dapat digunakan atribut **ROWSPAN** dan untuk menggabung kolom dapat digunakan atribut **COLSPAN**. Contoh penggunaannya dapat dilihat pada Gambar 13.34 dan 13.35.

```
<body>
<table width="100%" border="2"
cellspacing="2" cellpadding="2">
  <tr>
    <td>Propinsi</td>
    <td>Kota</td>
  </tr>
  <tr>
    <td rowspan="2">Jawa Timur </td>
    <td>Surabaya</td>
  </tr>
  <tr>
    <td>Malang</td>
  </tr>
  <tr>
    <td rowspan="2">Jawa Tengah </td>
    <td>Semarang</td>
  </tr>
  <tr>
    <td>Solo</td>
  </tr>
</table>
</body>
```



Gambar 13.34. Rowspan.

```
<body>
<table width="100%" border="2"
cellspacing="2" cellpadding="2">
  <tr>
    <td colspan="2">LUAS AREA KOTA </td>
  </tr>
  <tr>
    <td>Surabaya</td>
    <td>12000</td>
  </tr>
  <tr>
    <td>Malang</td>
    <td>11000</td>
  </tr>
  <tr>
    <td>Semarang</td>
    <td>13000</td>
  </tr>
</table>
</body>
```



Gambar 13.35. Colspan.

Sel pada tabel tidak selalu harus berisi teks namun dapat juga berisi gambar seperti terlihat pada Gambar 13.36.

```
<body>
<table width="100%" border="2" cellpadding="2"
cellspacing="3" bgcolor="#FFFFFF">
  <tr>
    <th>Nama Logo </th>
    <th>Gambar</th>
  </tr>
  <tr>
    <td>Logo Linux</td>
    <td><img src="../picture/150px-NewTux_Layer 1.jpg"
alt="linux logo" width="99" height="125" align="middle" /
></td>
  </tr>
  <tr>
    <td>Logo Windows</td>
    <td><img src="../picture/225px-Windows_logo_Layer
1.jpg" alt="windows logo" border="2" align="top" /></td>
  </tr>
</table>
</body>
```

Gambar 13.36. Tabel dengan sel berisi gambar.

13.4.6.Link antar Dokumen

Link merupakan pautan untuk membuka atau memanggil halaman *web* atau file tertentu. *Link* merupakan tag yang sangat penting dalam penggunaan HTML, karena disinilah letak perbedaan antara dokumen HTML dengan dokumen teks yang lain. *Link* dapat dibuat dengan memberi perintah tag *anchor* **<A>**. *Anchor* memiliki beberapa atribut, diantaranya **HREF** yang berfungsi untuk membuat link ke dokumen HTML tertentu dan **NAME** yang berfungsi untuk memberi tanda/nama titik tertentu pada dokumen HTML yang sama. Contoh penggunaan tag *anchor* dapat dilihat pada Gambar 13.37.

```
<BODY>
DAFTAR LINK
<menu>
  <li><a href="heading.htm" title="Heading">Heading</a>
  <li><a href="paragraf.htm" title="Paragraph">Paragraph</a>
  <li><a href="linebreak.htm" title="Line Break">Line Break</a>
  <li>Ordered List
  <li>Unordered List
</menu>
</BODY>
```

Gambar 13.37. Penggunaan tag anchor.

Atribut **HREF** dapat digunakan untuk memanggil halaman *web* pada sistem yang sama (pada satu komputer) seperti ditunjukkan pada Gambar 13.37. Pada kasus ini kita tinggal menuliskan lokasi dimana halaman yang akan kita

panggil berada. **HREF** dapat juga kita gunakan untuk memanggil halaman lain diluar sistem kita atau memanggil situs-situs lain di internet. Caranya dengan mengetikkan alamat URL situs yang akan kita panggil. Cobalah ketikkan kode HTML berikut ini dengan teks editor kemudian simpan file dengan ekstensi .htm.

```
DAFTAR ALAMAT MESIN PENCARI
<menu>
      <li><a href="http://www.google.com/">Google</a>
      <li><a href="http://www.yahoo.com/">Yahoo</a>
      <li><a
href="http://www.altavista.com/”>Altavista</a>
</menu>
```

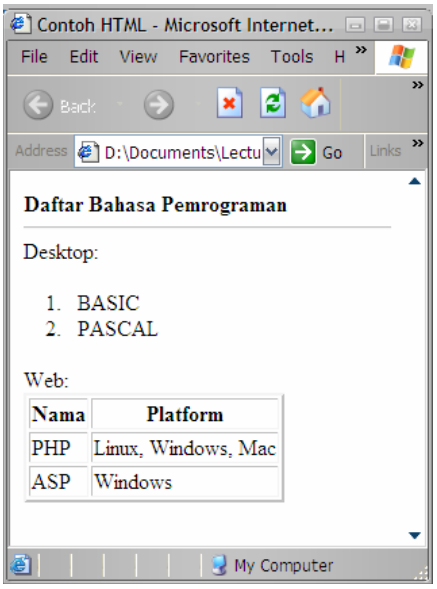
13.5. RINGKASAN

- Teknologi disain *web* terbagi menjadi beberapa layer (lapisan), yaitu structural layer, presentation layer dan behavioral layer.
- *Web* statis adalah halaman *web* yang isi data dan informasinya tidak berubah-ubah. Sedangkan *web* dinamis, memiliki isi data dan informasi yang berbeda-beda tergantung input apa yang disampaikan *client*.
- Pembuatan halaman *web* membutuhkan dukungan persiapan perangkat keras, perangkat lunak dan pemahaman teknologi pembuatan *web*.
- HTML *(Hypertext Markup Language)* merupakan bahasa yang digunakan untuk pembuatan halaman *web* dilakukan dengan cara disisipkan (*embedded language*) pada dokumen dengan memberi tanda tertentu yang disebut **tag.**
- Dokumen HTML secara umum akan terdiri dari dua bagian yaitu Header dan Body.
- HTML menyediakan tag-tag untuk pendeskripsian dokumen dan format dokumen yang lengkap.
- Pada halaman HTML dapat ditambahkan gambar, tabel, suara, atau file-file lain.

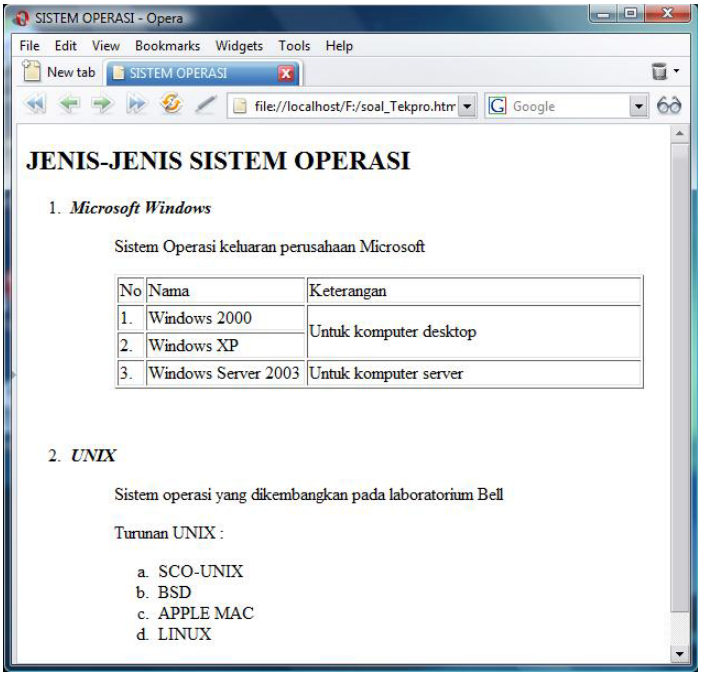
13.6. SOAL-SOAL LATIHAN

1. Apakah perbedaan antara *web* statis dan *web* dinamis?
2. Kunjungilah beberapa situs, kemudian cermatilah apakan halaman-halaman *web* yang tersedia termasuk halaman *web* statis ataukah dinamis?
3. Bukalah sebuah situs tertentu kemudian dari menu bar *web browser* kalian, pilih View kemudian pilih Source. Pada jendela yang muncul, cermatilah kode-kode HTML yang muncul. Cari bagian mana yang merupakan kode-kode untuk format dokumen, tabel, gambar, dan elemen-elemen HTML yang lain.

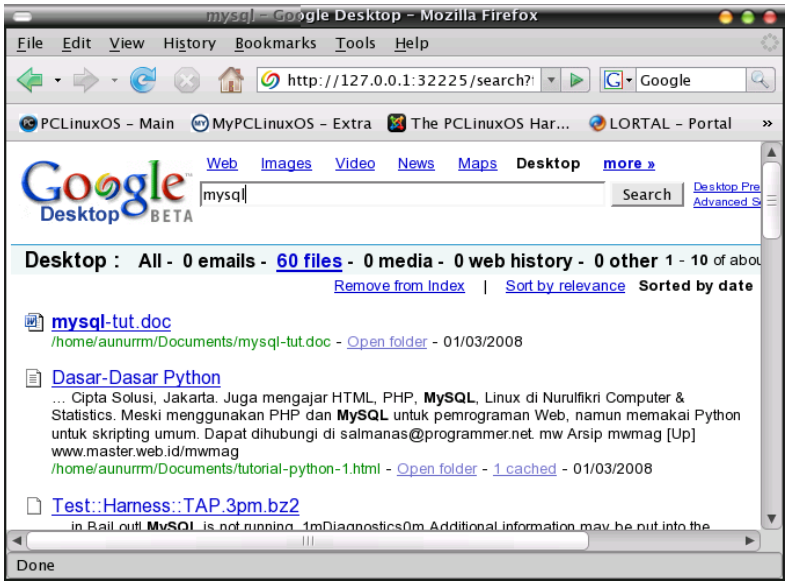
4. Buatlah halaman HTML seperti berikut.



5. Buat lagi halaman *web* seperti berikut.



BAB 14 DINAMIS BERBASIS JSP



Gambar 14.1. Halaman pencarian Google

.

Bagi kalian yang sering berselancar di internet gambar seperti di atas hampir pasti sering kalian jumpai. Gambar hasil pencarian dari Google di atas dapat digolongkan dalam halaman-halaman web dinamis. Hal ini karena apa yang akan ditampilkan berubah-ubah tergantung pada apa yang kita inputkan.

Bab ini membahas tiga standar kompetensi yaitu membuat halaman web dinamis dasar, membuat halaman dinamis tingkat lanjut dan membuat program aplikasi web berbasis JSP. Penggabungan tiga kompetensi ini karena kedekatan isi kompetensi dasar. Penyusunan sub bab tidak mengacu langsung pada kompetensi dasar, namun lebih mengacu pada urutan dan kedekatan pokok bahasan. Rangkuman diletakkan pada akhir bab dilanjutkan dengan soal-soal latihan yang disusun dari soal-soal yang mudah hingga soal-soal yang sulit. Latihan soal ini digunakan untuk mengukur kemampuan terhadap kompetensi dasar ini. Sebelum mempelajari kompetensi ini ingatlah kembali dasar sistem komputer, sistem operasi, algoritma pemrograman dasar, pemrogaman Java, web statis dan HTML.

TUJUAN

Setelah mempelajari bab ini diharapkan kalian akan mampu :
Mempersiapkan lingkungan teknis
Membuat halaman dinamis
Menambahkan fungsi-fungsi pada halaman dinamis
Menguji halaman dinamis
Memahami konsep pemrograman web
Mempersiapkan pembuatan aplikasi
Mengenal bahasa-bahasa skrip untuk pemrograman web
Mengenali isu-isu keamanan web
Menjelaskan kebutuhan perangkat lunak untuk aplikasi web berbasis JSP
Menjelaskan dasar-dasar JSP

14.1 DASAR WEB DINAMIS

Pengertian tentang web dinamis telah disinggung secara ringkas pada Bab 13. Pada bab ini kita akan lebih dalam mempelajari web dinamis.

Kebutuhan Lingkungan Teknis

Ada perbedaan yang penting dalam kebutuhan lingkungan pengembangan antara web statis dan web dinamis. Pada bab sebelumnya kalian telah mengetahui bahwa ada beberapa kebutuhan perangkat untuk membuat halaman web statis, antara lain perangkat komputer personal, sistem operasi, text editor atau perangkat lunak pembuat halaman web, dan *web browser*. Beberapa kebutuhan tersebut dapat digunakan untuk lingkungan web statis, namun beberapa perangkat tambahan juga diperlukan.

Perangkat keras

Pengembangan web dinamis membutuhkan perangkat keras yang lebih tinggi spesifikasinya dibanding web statis. Umumnya pengembang akan melakukan pembuatan web dinamis pada satu komputer yang berperan sekaligus sebagai *server* dan *client*. Tapi, seringkali juga dibutuhkan lebih dari satu komputer, dimana satu komputer berperan sebagai server dan yang lainnya sebagai *client*. Pada komputer yang berperan sebagai server maka spesifikasi teknisnya harus lebih tinggi dari komputer *client*. Kebutuhan prosesor yang lebih cepat dan memori utama yang lebih besar merupakan kebutuhan mutlak.

Perangkat lunak

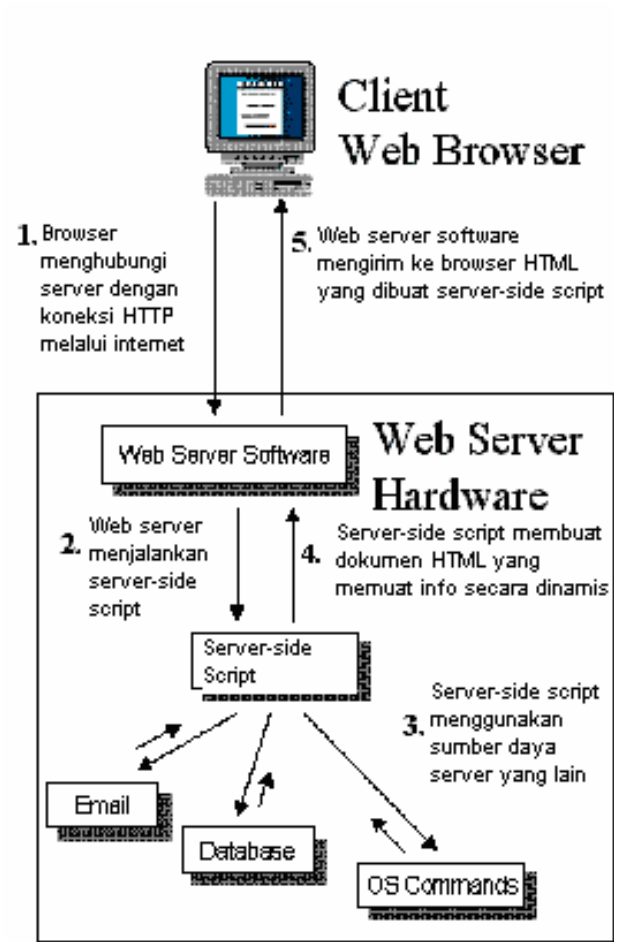
Sistem operasi yang digunakan pada pengembangan web dinamis lebih baik jika menggunakan versi yang mendukung. Biasanya pembuat sistem operasi akan menyediakan versi yang memang khusus diperuntukkan bagi server. Versi ini biasanya memiliki tingkat keamanan dan stabilitas yang lebih tinggi dari versi *desktop*-nya. Sebagai contoh, pada Linux ada distro Ubuntu versi server, SuSe menyediakan versi Enterprise Server, dan demikian juga distro-distro lainnya. Sedangkan pada Windows tersedia Windows NT, Windows Server 2000, Windows Server 2003 dan yang terbaru Windows Server 2008.

Kebutuhan perangkat lunak yang sangat membedakan antara web statis dan web dinamis adalah bahasa pemrograman sisi server. Bahasa pemrograman ini diinstall untuk digunakan web server menerjemahkan perintah-perintah tertentu

dalam bahasa tertentu. Sebagai contoh jika kita hanya menginstall *web server* Apache saja, maka kita tidak dapat menjalankan halaman web dinamis yang kita tulis dengan bahasa pemrograman PHP atau JSP. Agar dapat menjalankan halaman web dinamis tersebut kita perlu menginstall PHP atau Tomcat.
Perangkat pengembang web dinamis saat ini sudah banyak tersedia, meskipun beberapa memiliki harga yang relative mahal. Beberapa diantaranya adalah Microsoft Visual Studio, Borland Delphi Studio, Adobe Dreamweaver, dan lain-lain. Perangkat lunak ini selain dapat sebagai HTML Editor juga mendukung bahasa pemrograman server seperti ASP.Net, PHP, JSP, dan ColdFusion. Perangkat lunak pengembang yang gratis juga tersedia, antara lain NetBeans yang mendukung penuh JSP, Eclipse yang mendukung banyak bahasa pemrograman, Komodo Editor, dan lain-lain. Meskipun gratis, tetapi fasilitas yang disediakan tidak kalah dengan yang tidak gratis.
Perangkat lunak lain yang dibutuhkan adalah DBMS. Hal ini karena biasanya web dinamis menggunakan basis data sebagai tempat penyimpanan data. DBMS personal seperti Microsoft Access bukan pilihan yang baik untuk web dinamis, karena factor keamanan dan kinerjanya yang tidak sesuai dengan sifat-sifat web dinamis. Basis data berbasis SQL yang bersifat server seperti MySQL, Oracle, Microsoft SQL Server, dan lain-lain merupakan pilihan yang cocok untuk digunakan dalam web dinamis.

Pemrograman Web Dinamis

Pemrograman web merupakan usaha untuk membuat halaman web dengan menggunakan bahasa pemrograman web (*script*). Pemrograman web (*web programming*) dikenal juga dengan istilah pengembangan web (*web development*). Istilah lain yang mungkin juga cukup terkenal adalah *web design*. *Web design* lebih memfokuskan bagaimana merancang tampilan halaman-halaman web menjadi menarik bila dilihat. Atau boleh dikatakan *web design* lebih pada aspek visualnya sedangkan web programming lebih fokus pada aspek logika proses yang terjadi di dalam halaman-halaman web. Seorang *web programmer* mungkin juga seorang *web designer*, namun biasanya pekerjaan ini dipisahkan orangnya karena perbedaan fokus penekanannya.
Untuk membuat interaksi yang baik, dibutuhkan beberapa model pemrograman web. Model pemrograman yang umum digunakan adalah *client-side* dan *server-side*. *Client* dan *server* dalam kasus ini menunjukkan dua tempat yang berbeda. *Server* adalah komputer yang bertindak sebagai pihak yang melayani permintaan data atau informasi. Sedangkan *Client* adalah komputer pengguna yang hendak mengakses program ke *server* untuk meminta data atau informasi dengan menggunakan alamat yang unik. Secara umum interaksi dan pertukaran data antara *client* dan *server* dalam internet tampak pada gambar 14.2. Pada gambar tersebut, tampak bahwa segala proses dilakukan di *web-server* (*server-side*) sedangkan *client* hanya akan menerima hasil olahan dari *web-server* yaitu berupa halaman-halaman dalam format HTML.



Gambar 14.2. Pertukaran data antara *client* dan server.

Biasanya perangkat komputer yang bertindak sebagai *server* akan berjalan terus-menerus tanpa henti dan berperan sebagai tempat dimana file/program dari aplikasi web ditempatkan. Secara umum, untuk dapat mengakses layanan ke *server* dibutuhkan *browser* seperti Internet Explorer (IE), Opera, Mozilla dan Netscape.

Bahasa Skrip untuk Pemrograman Web

Seperti dijelaskan di atas ada dua model pemrograman web yaitu *client*-side dan server-side. Bahasa pemrograman untuk membuat web dinamis juga terbagi menjadi dua yaitu *client-side script* dan *server-side script*.

Bahasa pemrograman yang digunakan dalam membuat aplikasi pada sisi *client* biasa disebut sebagai *client-side script*. *Client-side script* yang umum digunakan adalah JavaScript dan VBScript. Keuntungan utama dari *client-side script* adalah waktu prosesnya yang jauh lebih cepat dibanding *server-side*. Hal ini karena seluruh permintaan pengguna akan diproses pada komputer pengguna sendiri. Namun masalah terbesar dari aplikasi dengan *client-side script* adalah keamanan

kode dan data. Hal karena pengguna dapat dengan mudah membuka dan melihat kode program.
Contoh penggunaan *client-side script* adalah :
aplikasi web untuk kalkulator, tanggal atau permainan.
pemeriksaan *event* pada *browser*, jika mouse diklik kanan maka akan muncul keterangan yang dibutuhkan.
validasi isi form yang diinputkan oleh pengguna, sebelum isi form tersebut dikirim ke server.
Bahasa pemrograman yang digunakan dalam membuat aplikasi pada sisi server biasa disebut sebagai *server-side script*.
Ada beberapa keuntungan pada penggunaan *server-side script*, yaitu :
Keamanan kode
Script yang kita buat tidak akan dapat dibaca oleh user karena seluruhnya akan disimpan dan dijalankan di *web server*. Yang akan dikirim ke *client* (pengguna) adalah hasil pengolahan yang berupa dokumen dalam format HTML saja.
Koneksi dengan basis data
Kemampuan koneksi dengan basis data merupakan keuntungan terbesar dari *server-side script*. Dengan kemampuan ini informasi yang dihasilkan oleh web server menjadi sangat dinamis bukan lagi halaman-halaman statis. Informasi yang disampaikan ke user tergantung dari apa yang diinginkan oleh user tersebut.
Dapat melakukan *tracking* (pelacakan) pengguna
Dengan *server-side script*, dapat diketahui siapakah *user* yang sedang akses ke aplikasi web dengan menggunakan fasilitas *session*.
Saat ini ada tiga teknologi utama yang digunakan dalam *server side script*, yaitu ASP (termasuk ASP.Net), PHP, dan JSP. JSP akan kita bahas pada bagian lain dari bab ini.

ASP

ASP sebenarnya bukan bahasa pemrograman karena ASP adalah mesin (*engine*) untuk *server side script* yang ditanamkan pada IIS. Bahasa pemrograman yang dipakai adalah VBScript. Namun orang lebih sering menyebut ASP saja. Secara default ini adalah *server side script* pada *web server* Microsoft Windows. Seperti halnya bahasa pemrograman web lainnya, untuk menandai bahwa suatu kode dalam halaman web merupakan VBScript maka digunakan tag sebagai penanda. Berikut ini suatu contoh VBScript:

```
<html>
<body>
<% Response.Write("Hello World!") %>
</body>
</html>
```

Atau sebagai berikut:

```
<html>
<body>
<%= "Hello World!" %>
</body>
</html>
```

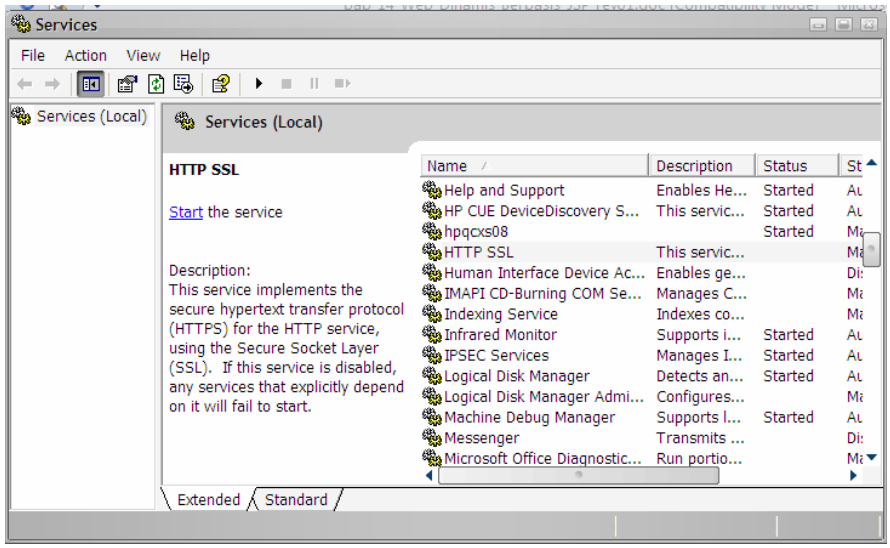
Pada kode di atas bagian yang ditandai dengan <% dan %> merupakan bagian yang memuat kode VBScript. VBScript merupakan bahasa yang sangat mirip dengan Visual Basic. Namun telah disesuaikan dengan lingkungan web. Dokumen berisi kode ini dapat disimpan dalam bentuk html atau dengan ekstensi .asp. Namun yang paling penting adalah dokumen tersebut harus diletakkan pada direktory yang dapat dibaca dan dieksekusi oleh web server IIS.
ASP.Net merupakan versi lanjutan dari ASP, namun dengan teknologi yang sangat berbeda. ASP.Net dibangun berdasarkan pada teknologi .Net yang dikembangkan oleh Microsoft. Bahasa yang digunakan tidak lagi VBScript namun telah menggunakan kemampuan penuh dari salah satu dari bahasa-bahasa dalam kelompok .Net yaitu Visual Basic.NET, C#, Visual J. Dengan teknologi .Net memungkinkan tag-tag HTML digantikan secara penuh oleh *script-script* yang ditulis dengan bahasa-bahasa .Net.

PHP

PHP adalah bahasa pemrograman yang didesain khusus untuk membuat halaman web. PHP adalah singkatan dari *PHP Hypertext Preprocessor*. Singkatan yang agak aneh. Awalnya, PHP adalah singkatan *dari Personal Home Page* yang pertama kali diciptakan oleh Rasmus Lerdorf. PHP diciptakan pertama kali untuk keperluan mencatat jumlah pengunjung *homepage*nya. Perkembangan php saat ini dapat dilihat pada www.php.net.
PHP adalah salah satu bahasa *server-side* yang paling populer. Kepopulerannya disebabkan kelebihan-kelebihannya dibanding bahasa sejenis, seperti Perl dan CGI. PHP mampu menutupi kekurangan pada bahasa pemrograman web pada umumnya. PHP mudah dibuat dan cepat dijalankan. PHP dapat berjalan dalam *web server* yang berbeda, seperti Apache, PWS, IIS dan sebagainya. PHP juga dapat berjalan dalam sistem operasi yang berbeda pula, seperti UNIX, Windows, Mac OS X dan Linux. PHP diterbitkan secara gratis. *Source code* PHP dapat di-*download* tanpa perlu megeluarkan uang. PHP juga termasuk bahasa yang *embedded* (bisa diletakkan di dalam tag HTML).

Persiapan Membuat Halaman Web Dinamis

Persiapan yang perlu dilakukan sebelum membuat halaman web dinamis antara lain adalah:
Memeriksa apakah web server sudah terinstal dengan benar dan berjalan sebagaimana mestinya.
Pada system operasi windows, kita dapat memeriksa apakah web server berjalan atau tidak melalui jendela *Service*. Buka *Control Panel*, kemudian klik ganda pada ikon *Administrative Tools*. Pilih ikon *Service* dan klik ganda pada ikon tersebut sampai terbuka jendela seperti pada Gambar 14.3.

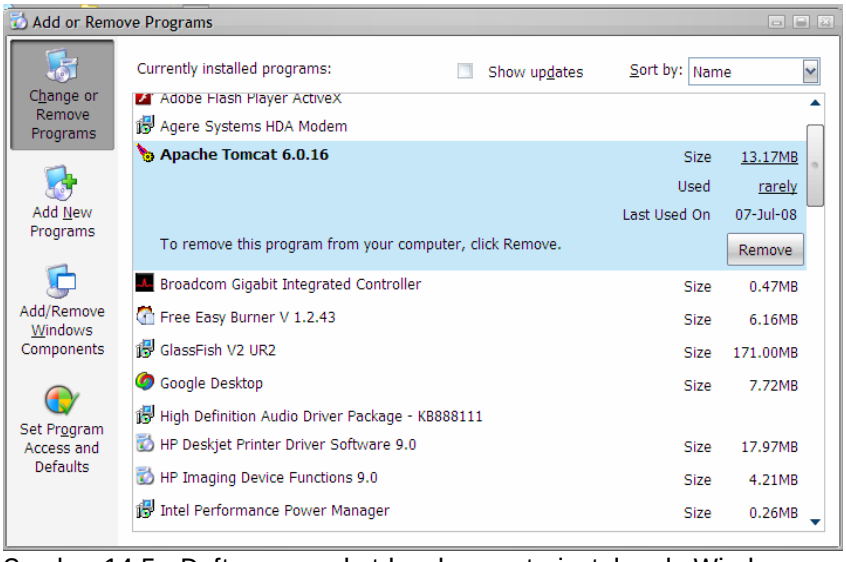


Gambar 14.3. Jendela Services.

Pada jendela Services tersebut cari nama servis *web server* kalian. Jika kalian menggunakan IIS cari nama *Internet Information Services* pada kolom nama. Jika kalian memakai *Apache*, cari juga nama *Apache*. Jika belum ada mungkin belum terinstal atau belum didaftarkan sebagai servis. Jika sudah ada maka periksalah apakah sudah dijalankan apa belum dengan cara memeriksa kolom status apakah sudah *started* atau belum. Jika belum, klik *start* untuk menjalankan servis tersebut. Setelah itu ujilah dengan menjalankan *web browser* kalian. Ketikkan pada bagian alamat : http://localhost/ kemudian tekan enter. Pada beberapa konfigurasi web server, pemanggilan alamat mungkin harus menyertakan nomor port seperti http://localhost:8080/. Jika kalian mendapati halaman muka web server kalian tampil (lihat Gambar 14.4) berarti web server telah dapat berjalan. Tetapi jika tidak muncul atau terjadi pesan kesalahan berarti *web server* belum dapat berjalan dengan baik.

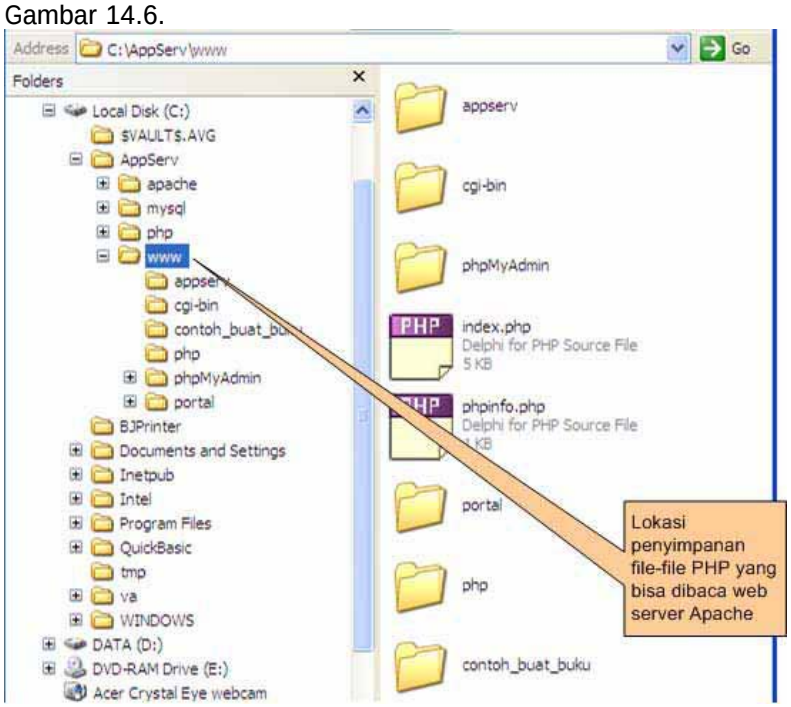
Gambar 14.4. Opera sedang memanggil alamat *server*.

Memeriksa apakah bahasa pemrograman *server* yang akan digunakan sudah terinstall dan dapat dipanggil oleh *web server*.
Pemeriksaan dapat dilakukan dengan cara memeriksa daftar perangkat lunak yang sudah diinstal di dalam sistem. Pada Windows dapat dilakukan dengan cara membuka jendela *Add and Remove Programs* seperti pada Gambar 14.5. Jika perangkat lunak bahasa pemrograman kalian sudah terinstal maka akan terdaftar pada jendela ini.



Gambar 14.5. Daftar perangkat lunak yang terinstal pada Windows.

Mempersiapkan lokasi penyimpanan file-file kode program.
File-file yang berisi kode-kode *server-side script* harus diletakkan dalam direktori yang bisa dibaca oleh *web server*. Sebagai contoh pada beberapa distribusi Linux file-file PHP disimpan pada direktori /usr/var/www/ dibeberapa distribusi yang lain disimpan di direktori /svr/www/. Pada Microsoft Windows demikian juga. Lokasi ini dapat diubah dengan cara mengedit file konfigurasi Apache Web Server (httpd.conf). Contoh lokasi penyimpanan file-file PHP dapat dilihat pada Gambar 14.6.



Gambar 14.6. Lokasi direktori yang bisa dibaca web server.

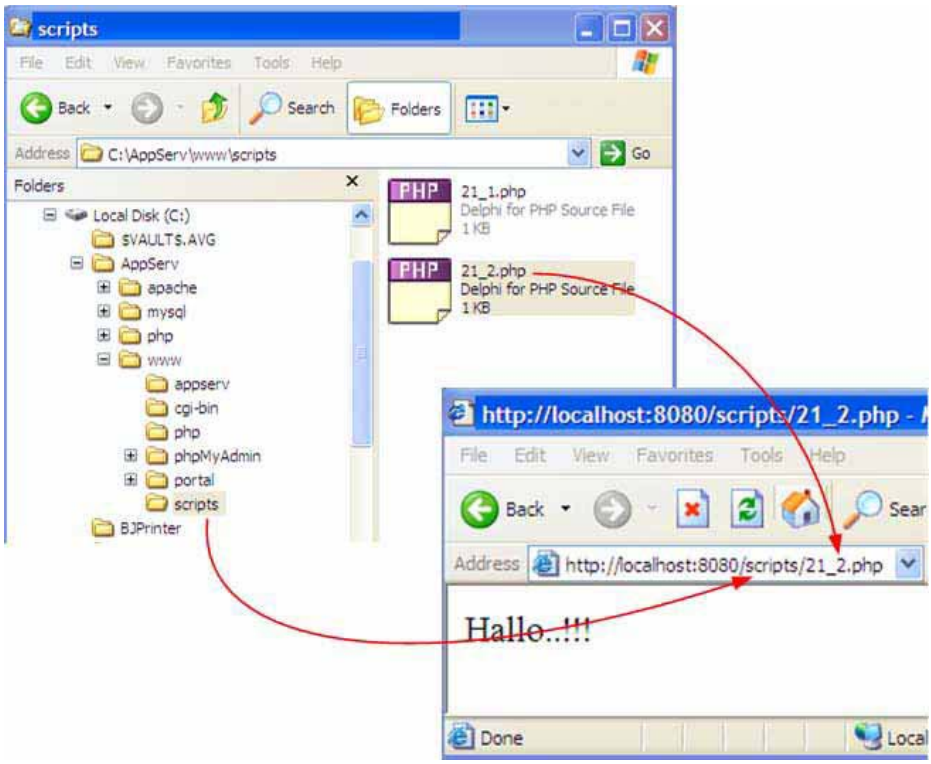
Pembuatan dan Pengujian Halaman Web Dinamis

Pada bagian ini kita akan mencoba membuat halaman dinamis sederhana dengan bahasa PHP pada *web server* Apache.
Buka teks editor kalian (bisa menggunakan Notepad) kemudian ketikkan kode berikut ini.

```
<?php
echo “Hallo, ini PHP.  Salam kenal ya..!!!”;
?>
```

Simpan file dengan nama 21_1.php dan simpan di direktori yang bisa diakses *web server* Apache (pada kasus ini disimpan di direktori c:\appserv\www\scripts\).
Bukalah *web browser* kalian kemudian ketikkan di halaman alamat: http://localhost:8080/scripts/21_2.php. *Localhost* adalah nama *server*-nya dan 8080 adalah *port* dimana Apache dijalankan. Perhatikan bagaimana menuliskan alamat ini dan bandingkan dengan lokasi asli dari file tersebut. Jika semuanya lancar maka kalian akan mendapati tampilan seperti pada Gambar 14.7.

Halaman web dinamis harus dipanggil dengan cara tersebut. Karena memang hanya bisa dijalankan oleh *web server*. Kalau dengan file halaman web statis, kalian bisa langsung klik ganda pada file tersebut dan web browser akan menampilkan hasilnya. Tetapi dengan halaman web dinamis tidak bisa dengan cara tersebut. Cobalah klik ganda pada halama web dinamis yang telah kalian buat di atas. Bagaimanakah outputnya?



Gambar 14.7. Pengujian halaman web dinamis.

SEKILAS TENTANG JSP

Seperti telah dijelaskan di atas, *Java Server Pages* (JSP) adalah bahasa scripting untuk web programming yang bersifat *server side* seperti halnya PHP dan ASP. JSP dapat berupa gabungan antara baris HTML dan fungsi-fungsi dari JSP itu sendiri.

JSP adalah suatu teknologi web berbasis bahasa pemrograman Java dan berjalan di Platform Java, serta merupakan bagian teknologi J2EE (*Java 2 Enterprise Edition*). Teknologi JSP menyediakan cara yang lebih mudah dan cepat untuk membuat halaman-halaman web yang menampilkan isi secara dinamik. Teknologi JSP didesain untuk mempermudah dan mempercepat pembuatan aplikasi berbasis web yang bekerja dengan berbagai macam *web server, application server, browser* dan *development tool*.

Java dan JSP

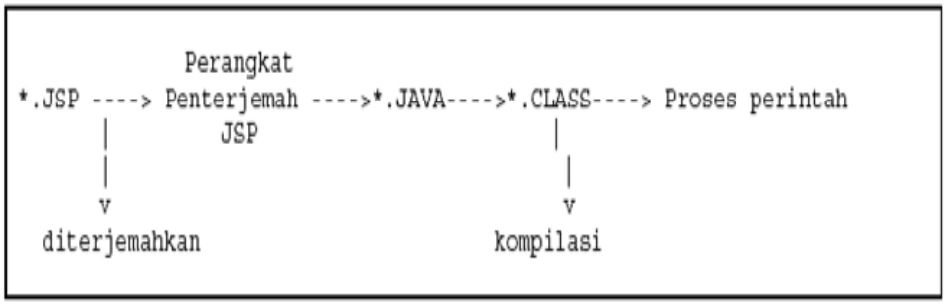
Ada tiga cara untuk menggunakan teknologi Java dalam lingkungan web, yaitu *applet, servlet* dan JSP. Applet merupakan program Java yang disisipkan pada halaman HTML dengan menggunakan tag <APPLET>. Kita dapat membuat sebuah program yang kompleks dengan menggunakan bahasa Java kemudian jika ingin ditampilkan dalam halaman web, kita menyimpan program tersebut dalam bentuk applet dan menyisipkannya pada halaman-halaman HTML. Applet ini akan dieksekusi oleh *Java Virtual Machine* (JVM) pada *browser*. Sayangnya seringkali komputer *client* tidak didukung oleh JVM, sehingga applet tersebut tidak dapat dijalankan.

Servlet adalah program yang ditulis dengan bahasa Java yang dijalankan pada *server* yang terkoneksi web. Pada *servlet*, komputer client tidak membutuhkan JVM, karena semua kode program akan dieksekusi di sisi server. Output yang akan disampaikan kepada browser client adalah murni HTML. Servlet ini akan dikompilasi dalam bentuk class. Servlet secara umum mampu meningkatkan sisi interaktif dan dinamis halaman web. Kelemahan dari servlet adalah teks-teks HTML yang berisi tampilan halaman harus dibuat dengan menggunakan bahasa pemrograman Java. Hal ini menyulitkan jika kita ingin mengubah tampilan bagian HTML-nya, karena berarti kita harus membuka kembali kode program java di servlet.

JSP merupakan solusi dari *servlet*. Kita tidak perlu mengkodekan teks-teks HTML pada program, tapi cukup menyisipkan kode JSP pada teks HTML. Artinya bagian static yang berupa tag-tag HTML akan terpisah dari kode JSP. Kita dapat membuat halaman web static dengan HTML / *Web editor*, kemudian kita sisipi dengan kode JSP untuk membuat halaman menjadi web dinamis.

Mekanisme Kerja Aplikasi Web Berbasis JSP

Secara umum aplikasi web berbasis JSP akan mengikuti mekanisme seperti Gambar 14.8. Halaman-halaman web yang mengandung kode JSP akan dikirim ke web server. Kemudian *web server* akan memanggil perangkat lunak penerjemah JSP. Oleh penerjemah ini kode JSP akan dirubah menjadi file sumber berekstensi .Java. file ini akan dikompilasi untuk menghasilkan *class*. Hasil dari kompilasi ini akan dijalankan kemudian hasilnya akan disampaikan kepada web server untuk diteruskan pada *browser client*.



Gambar 14.8. Mekanisme kerja aplikasi web dengan JSP.

Sekilas proses di atas terlihat bertele-tele. Pada awalnya, memang untuk menampilkan halaman yang mengandung JSP terasa lambat karena proses kompilasi harus dilakukan. Tetapi ketika memanggil untuk yang kedua kali maka

proses penampilan halaman akan sangat cepat. Hal ini karena proses kompilasi tidak perlu dilakukan lagi.

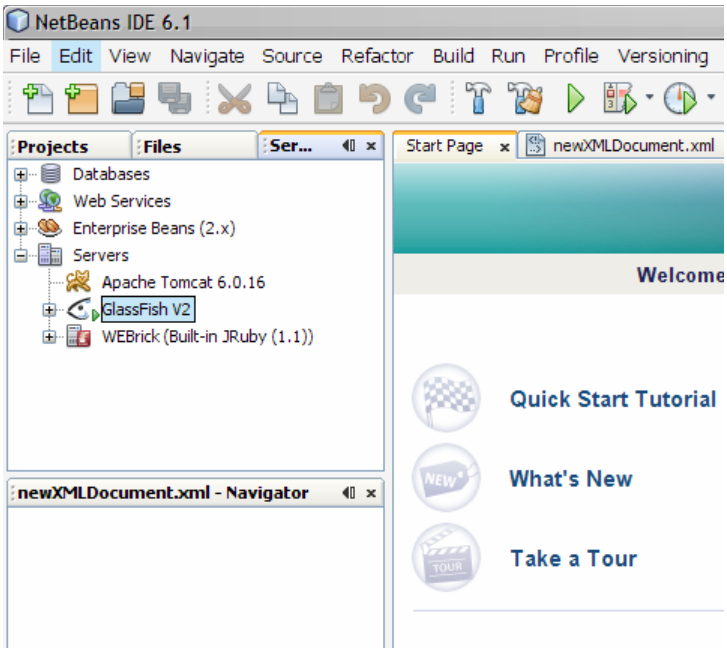
Kebutuhan Perangkat Lunak dan Konfigurasi

Untuk membangun aplikasi web berbasis JSP, diperlukan perangkat lunak yang hampir sama dengan web dinamis lainnya. Namun ada kekhususan dibanding dengan ASP atau PHP. Hal ini karena perbedaan mekanisme proses antara JSP dengan ASP atau PHP. Selain perangkat lunak yang telah disebutkan di bagian awal berikut ini adalah kebutuhan perangkat lunak tambahan untuk JSP.

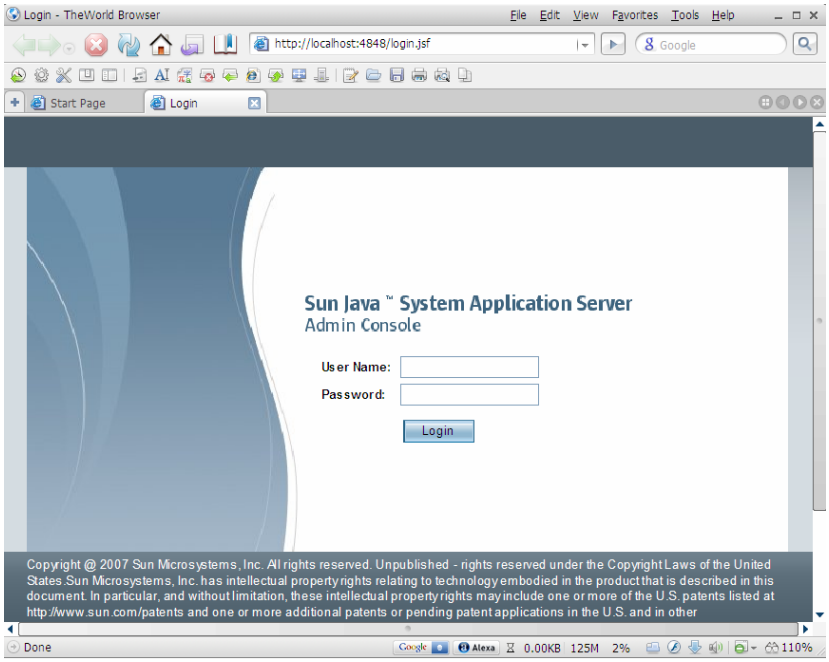
Java Development Kit (JDK). Karena JSP berdasarkan bahasa Java, maka JDK harus terinstal di dalam komputer. JDK telah dibahas secara singkat pada bab 8.

Tomcat. Tomcat adalah *servlet container* dan implementasi JSP. Dibutuhkan untuk mensimulasi komputer personal menjadi *web server*. Tomcat biasanya diinstalkan di atas *web server* Apache, sehingga orang sering menyebut sebagai Apache Tomcat. Perangkat lunak ini dapat di*download* gratis di situs http://jakarta.apache.org.

Aplikasi pengembang terpadu (IDE) seperti NetBeans atau Eclipse. Meskipun membuat halaman JSP bisa dengan menggunakan teks editor biasa namun disarankan untuk menggunakan aplikasi seperti NetBeans untuk mempermudah pembuatan halaman web. Bahkan pada versi 6.1. (dapat di*download* di http://www.netbeans.org) selain mendapatkan IDE kita juga mendapatkan *web server* dan *servlet container* (Apache Tomcat) terpadu (Gambar 14.9). Kita juga bisa menggunakan aplikasi server yang sangat kuat yaitu SUN Java Application Server (Gambar 14.10) yang juga terintegrasi pada paket penuh NetBeans. Pada buku ini kita akan banyak menggunakan fasilitas dari NetBeans baik untuk konfigurasi maupun untuk membuat halaman-halaman JSP.

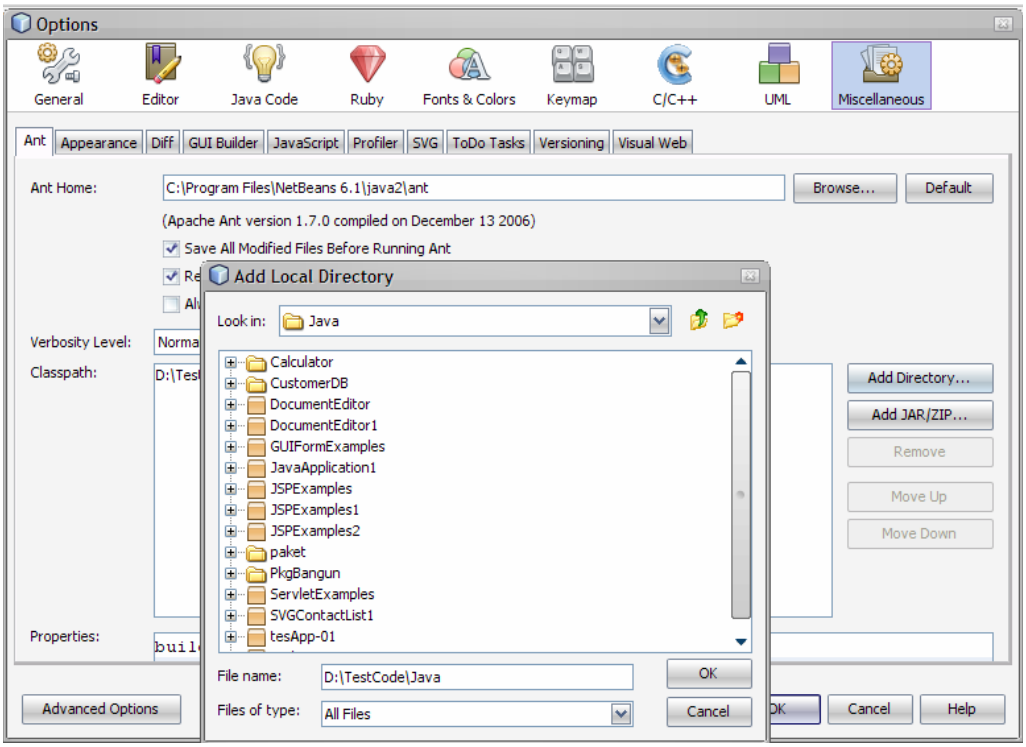


Gambar 14.9. NetBeans 6.1 dengan Apache Tomcat terintegrasi.



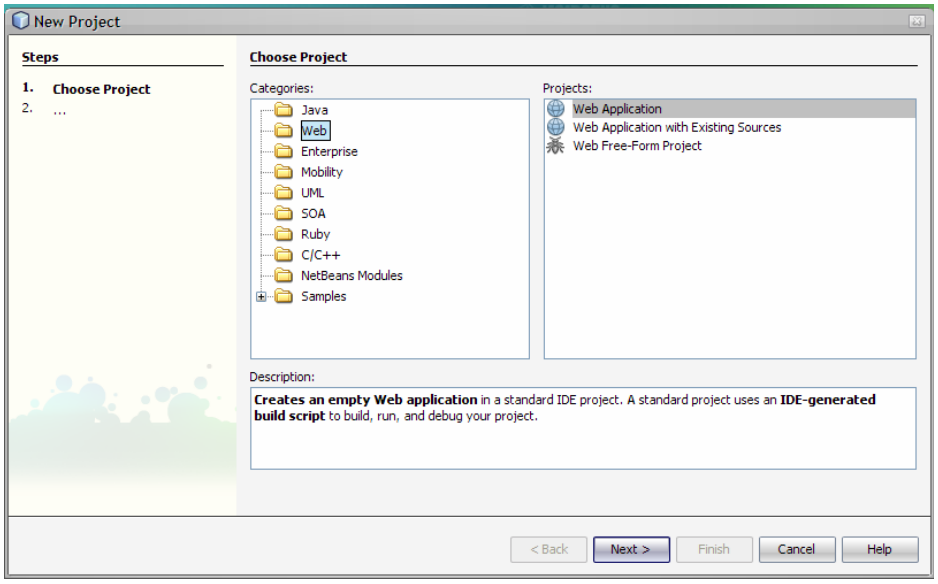
Gambar 14.10. *Admin Console Java Application Server*.

Setelah semua terinstal dengan benar, maka langkah yang kedua adalah konfigurasi, terutama lokasi penyimpanan file. Buka NetBeans, kemudian setelah terbuka pilih menu *Tools* kemudian pilih *Options*. Pada jendela Options (Gambar 14.11), pilih tab *Miscellaneous*. Klik Add Directory untuk untuk membuka jendela *Add Local Directory*. Pilih lokasi kemudian tekan OK. Pada klik Ok pada jendela Options untuk keluar.

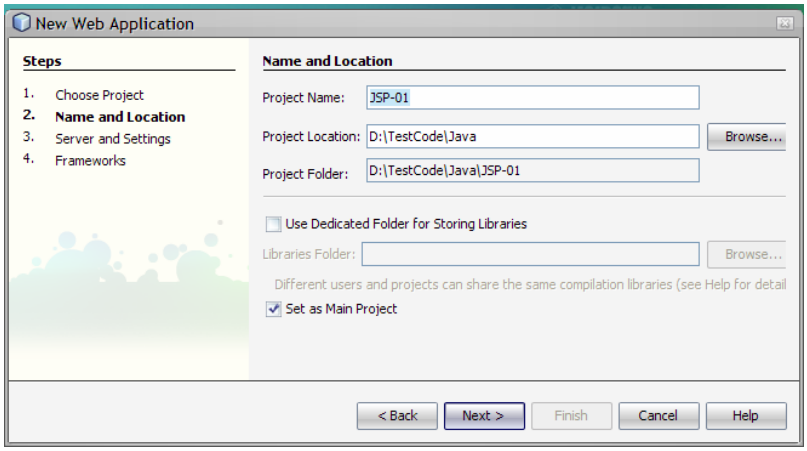


Gambar 14.11. Penentuan lokasi penyimpanan file.

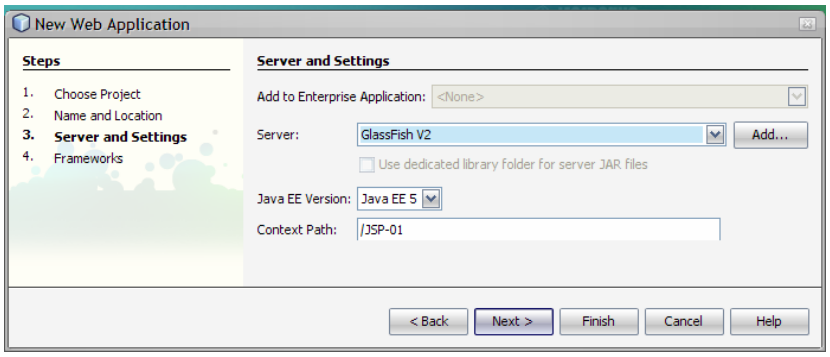
Untuk menguji apakah halaman JSP dapat dijalankan, pada halaman awal NetBeans, klik menu *File* dan pilih *New Project*. Jendela *New Project* seperti Gambar 14.12. akan terbuka. Pada bagian *Choose Project* pilih kategori *Web* dan pada *Projects* pilih *Web Application*. Klik Next untuk melanjutkan pada menentukan nama dan lokasi penyimpanan *project* (Gambar 14.13). klik Next untuk melanjutkan menentukan tipe dan setting server (Gambar 14.14). Pada bagian ini kita dapat memilih apakah menggunakan Apache Tomcat atau GlassFish (Java *Application Server*). Klik *Finish* untuk menyelesaikan konfigurasi aplikasi web.



Gambar 14.12. Penetuan tipe project.

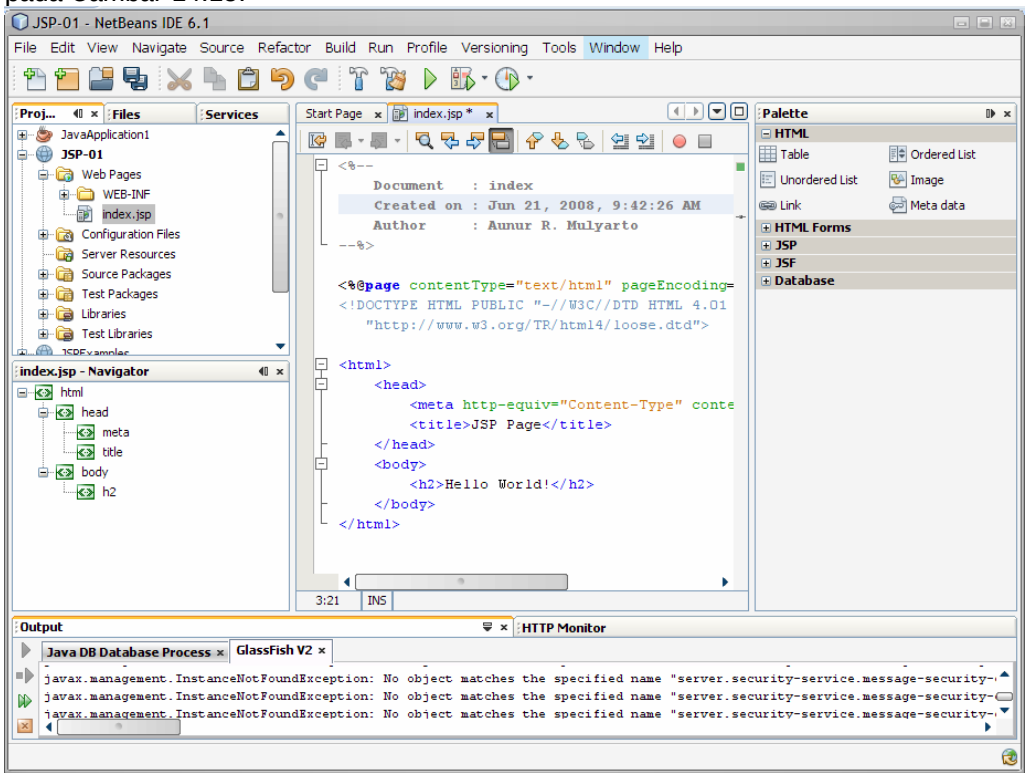


Gambar 14.13. Penentuan nama dan lokasi *project*.

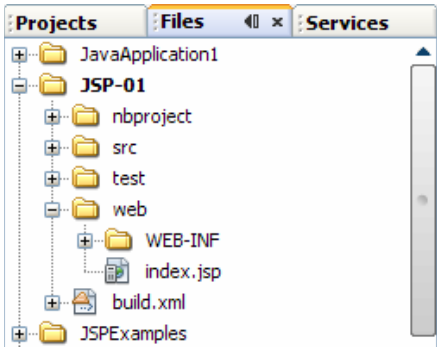


Gambar 14.14. Tipe dan pengaturan *server*.

Setelah kalian menekan *Finish* kalian akan dihadapkan pada tampilan seperti pada Gambar 14.15.



Gambar 14.15. Lingkungan kerja NetBeans.



Gambar 14.16. Struktur direktori aplikasi JSP

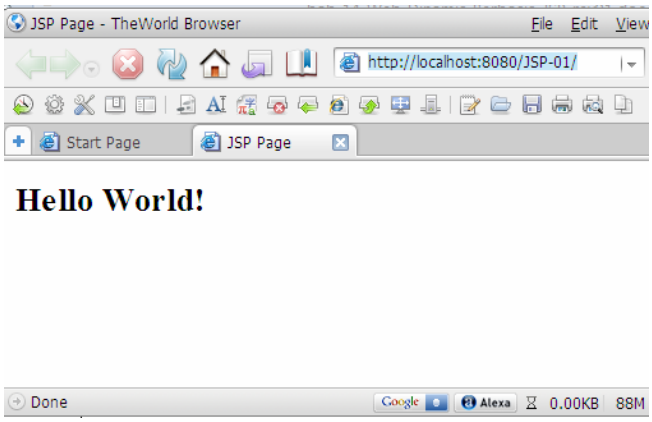
Perhatikan pada jendela Project klik Files kemudian cermati struktur direktori pada direktori JSP-01 (Gambar 14.16). Struktur direktori aplikasi JSP harus mengikuti aturan yang ketat. Di bawah direktori JSP-01 terdapat direktori dengan penting yaitu web. direktori ini tempat kita menyimpan file dengan ekstensi .jsp. Di bawah direktori web terdapat direktori WEB-INF. Direktori ini harus ditulis namanya seperti ini, jika tidak maka aplikasi tidak akan jalan. Direktori ini berisi file-file konfigurasi yang penting bagi jalannya aplikasi web dengan JSP.

Coba kalian jalankan halaman JSP yang telah otomatis dibuatkan oleh NetBeans dengan cara pilih menu *Run*, kemudian pilih *Run Main Project*. Kode program yang dibuatkan otomatis oleh NetBeans ini seperti berikut:

<%- -

```
    Document   : index
    Created on : Jun 21, 2008, 9:42:26 AM
    Author     : Aunur R. Mulyarto
--%>

<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01
Transitional//EN"
   "http://www.w3.org/TR/html4/loose.dtd">

<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html;
charset=UTF-8">
        <title>JSP Page</title>
    </head>
    <body>
        <h2>Hello World!</h2>
    </body>
</html>
```

Jika *web browser* kalian telah berhasil menampilkan halaman seperti pada Gambar 14.17, berarti kalian telah siap untuk membuat aplikasi dengan JSP. Perhatikan pada alamat URL-nya, halaman ini dipanggil dengan alamat http://localhost:8080/JSP-01/. Karena file yang kita panggil adalah file index.jsp, maka kita dibolehkan memanggil halaman tersebut tanpa menyebut nama file. Konfigurasi pada *server* telah membuat *server* mengenali bahwa halaman dengan nama index adalah halaman awal.



Gambar 14.17. Hasil eksekusi halaman web JSP.

DASAR-DASAR JSP

JSP menyediakan empat kategori tag, yaitu *directive*, elemen *scripting*, komentar, dan *action*. Pada bagian ini kita tidak akan mempelajari tentang deklarasi variabel, struktur kontrol program, *class* dan *method*. Hal ini karena

JSP didasarkan pada java, sehingga bagian-bagian tersebut sama dengan apa yang kalian pelajari pada bab tentang pemrograman Java (Bab 8).

Directive

Directive adalah sekumpulan tag yang menentukan bagaimana dokumen yang berisi direktif ini akan diproses. *Directive* digunakan JSP untuk mengirimkan "pesan" ke JSP *container*. *Directive* berguna untuk melakukan setting nilai global seperti deklarasi *class* atau *method*. Setting yang dilakukan oleh directive berlaku pada seluruh halaman (hanya halaman itu saja).

Secara umum sintaks *directive* adalah sebagai berikut :

```
<%@ nama_directive atribut1="nilai1" atribut2="nilai2" . . %>
```

Directive pada JSP terdiri atas tiga jenis tentu saja dengan fungsi yang berbeda-beda.

Page : digunakan untuk mendefinisikan atribut-atribut yang terdapat pada halaman JSP. Atribut-atribut ini misalnya atribut *language, import, info, errorpage* dan lain-lain. Cara penulisannya dengan menggunakan tanda @ setelah tag JSP (<%) diikuti kata page dan atributnya.

Contoh : `<%@ page language="java" %>`

Include : digunakan untuk menyisipkan suatu berkas atau mengimpor suatu kelas. Cara penulisannya dengan menggunakan tanda @ setelah tag JSP (<%) diikuti kata include dan atributnya.

Contoh : `<%@ include file="/header.html" %>`

Taglib : digunakan untuk mendefinisikan tag-tag yang dibuat sendiri oleh pemrogram. Cara penulisannya dengan menggunakan tanda @ setelah tag JSP (<%) diikuti kata taglib dan atributnya.

Contoh:

```
<%@ taglib uri =
"http://jakarta.apache.org/taglibs/application-1.0"
prefix="app" %>
```

Elemen Scripting

Elemen *scripting* digunakan untuk menggabungkan instruksi-instruksi pemrograman Java ke dalam halaman web. Instruksi tersebut akan dieksekusi setiap kali halaman diproses sebagai permintaan. Ada tiga jenis *scripting* yaitu deklarasi, *scriplet* dan ekspresi.

Deklarasi

JSP menyediakan tag yang secara khusus ditujukan untuk melakukan pendeklarasian variable yang berlevel halaman. Variabel seperti ini akan dikenali di sepanjang halaman. Tag yang dimaksud dinamakan tag deklarasi. Tag ini berbentuk sebagai berikut :

```
<%!.....%>
```

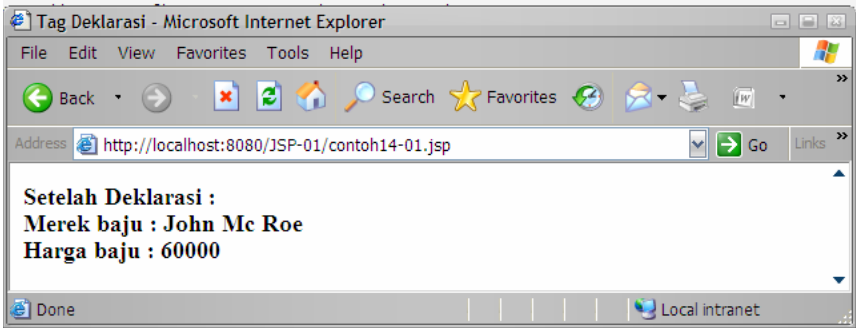
Contoh 14.1. Penggunaan tag deklarasi.

Ketikkan kode berikut dan simpan dengan nama contoh14-1.jsp.

```
<html>
    <head>
        <title>Tag Deklarasi</title>
    </head>
    <body>
```

```
<%!
    String baju;
    int harga = 60000;
%>

Setelah Deklarasi : <br>
<%
    baju = "John Mc Roe";
    out.println("Merek baju : " + baju + "<BR>");
    out.println("Harga baju : " + harga + "<BR>");
%>
    </body>
</html>
```

Jalankan program tersebut maka tampilan hasil akan tampak seperti pada Gambar 14.18. Pada contoh kode ini tag deklarasi digunakan untuk mendeklarasikan dua variabel yaitu baju dengan tipe data *string* dan harga dengan tipe data *int*. Perhatikan juga cara mengisi, memanggil dan menampilkan kembali isi dari variabel dengan pernyataan out.println.



Gambar 14.18. Hasil eksekusi penggunaan tag deklarasi.

Scriplet

Scriplet merupakan sekumpulan kode program Java uang dijalankan setiap kali halaman JSP dipanggil. Pada bagian ini kalian bisa memasukkan kode-kode program Java yang telah kalian pelajari pada Bab 8. Cara penulisannya adalah dengan memberikan tag <% *kode* %>.

Contoh 14.2. Penggunaan tag scriplet.

```
<html>
    <head>
        <title>Contoh Println</title>
    </head>
    <body>
    <%
      out.print("Ini keluar dari tag scriplet");
      out.print("Coba saja kalau tidak percaya”);
    %>
    </body>
</html>
```

Simpan file dengan nama yang diakhiri .jsp. Jalankan kode progam tersebut dan periksalah hasilnya.

Ekspresi

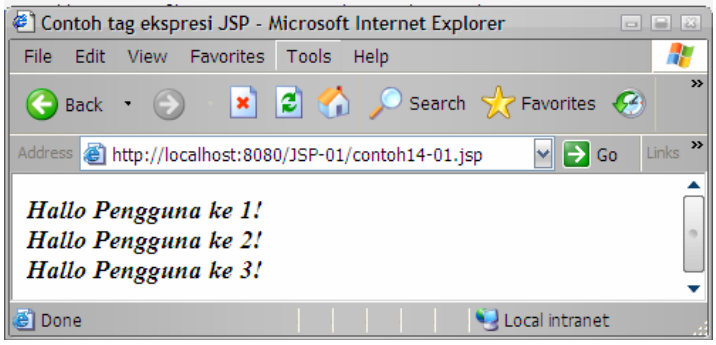
Ekspresi adalah satu baris perintah yang digunakan untuk mengeksekusi perintah sekaligus menampilkan dalam halaman web. Eskpresi ini mirip seperti ketika kita memanggil fungsi/*method* pada Java. Cara penulisannya adalah dengan menggunakan tag <%= *kode* %>

Contoh 14.3. Penggunaan tag ekspresi.

Ketikkan kode di bawah ini kemudian simpan sebagai file jsp.

```
<%! //Deklarasi variabel
    int jmlUser=3;
    //Deklarasi method
    public String Hello(String nama) {
        return "Hallo " + nama;
    }
%>
<HTML>
<HEAD>
    <TITLE>Contoh tag ekspresi JSP</TITLE>
</HEAD>
<BODY>
    <H3>
<% for (int i=1; i<=jmlUser; i++) {
       //Contoh scriplets
%>
       <i><%=Hello("User "+ i + "!")%> </i> <br>
   <%}%>
    </H3>
</BODY>
</HTML>
```

Jalankan kode program di atas. Tampilan output akan tampak seperti pada Gambar 14.19.



Gambar 14.19. Hasil eksekusi tag ekspresi.

Komentar

Komentar digunakan untuk memberikan keterangan pada kode-kode JSP. Ada tiga jenis komentar yang dapat disisipkan dalam halaman JSP, yaitu komentar *content*, komentar JSP dan komentar bahasa *script*.

Komentar *content* ditulis dengan tag <!-- *komentar* -->. Komentar model ini tidak menampilkan output ke browser. Tapi tertulis pada source HTML.

Komentar JSP merupakan tipe komentar yang hanya tampak pada kode program JSP. Ketika kode JSP dikompilasi, komentar tersebut akan dilewati dan tidak ikut dikompilasi. Komentar ini dituliskan dengan cara <%-- komentar --%>.

Komentar bahasa *script* sama dengan yang digunakan pada Java (lihat kembali bab 8). Komentar ini disisipkan pada *scriplet*. Pada contoh 14.3 kita telah menggunakan salah satu cara komentar bahasa *script*. Perhatikan pada baris yang diawali dengan tanda //. Cara yang lain adalah dengan tanda /* komentar */.

Action

Action mendukung beberapa aksi yang berbeda-beda. Hampir mirip dengan *scripting*, *action* akan dieksekusi setiap kali ada permintaan. Fungsi penting dari *action* adalah memungkinkan terjadinya transfer kontrol antar halaman, mendukung penggunaan *applet* Java dan memungkinkan JSP terintegrasi dengan komponen JavaBeans.

JSP mengenal tag *action standar* dan *custom tag*. Tag standar adalah tag yang didefinisikan dalam spesifikasi JSP, sedangkan custom tag adalah tag baru yang dapat didefinisikan sendiri. Pada bagian ini hanya dibahas mengenai tag *action* standar. Tag *action* standar JSP adalah sebagai berikut :

<jsp:useBean>
<jsp:setProperty>
<jsp:getProperty>
<jsp:param>
<jsp:include>
<jsp:forward>
<jsp:plugin>

Pada buku ini kita akan mempelajari penggunaan *action standard forward* dan *param*. *Forward action* ini digunakan untuk mentransfer kontrol dari sebuah halaman JSP ke halaman lain pada server lokal. Saat proses berlangsung, baris-baris kode sesudah *forward action* pada JSP asal tidak akan diproses lagi oleh JSP container. Proses berpindah pada halaman tujuan. Cara penulisannya adalah dengan menggunakan tag **<jsp:forward page="*localURL*" />**. *Forward action* biasanya digunakan bersama-saman dengan *Param action*. Perhatikan contoh berikut.

Contoh 14.4. Penggunaan *forward action* untuk operasi login.

Buat tiga file masing-masing bernama login.html, testlogin.jsp, dan validlogin.jsp. Isi dari masing-masing file adalah sebagai berikut.

File login.html

```
<html>
  <head>
    <title>Login </title>
  </head>
```

```
  <body>
      <FORM ACTION="testlogin.jsp" METHOD="post">
          Silakan masukkan nama dan password anda : <BR>
          Nama :<INPUT TYPE="text" NAME="nama"> <BR>
          Password :<INPUT TYPE="text" NAME="pass"> <BR>
          <INPUT TYPE="SUBMIT" VALUE="Login">

      </FORM>
  </body>
</html>
```

File testlogin.jsp

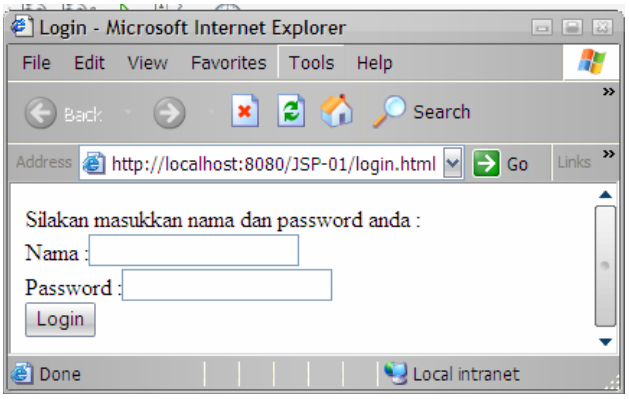
```
<html>
    <head>
              <title>Test Login </title>
    </head>
    <body>
     <%
        String nama = request.getParameter("nama");
        String pass = request.getParameter("pass");

        if ((nama.equals("aunurrm")) &&
(pass.equals("arm"))) {
     %>
        <jsp:forward page='validlogin.jsp' >
             <jsp:param name='id' value='<%= nama %>' />
        </jsp:forward>
     <% }
        else {
     %>
        <jsp:forward page='login.html' />
     <% }
     %>
    </body>
</html>
```

File validlogin.jsp

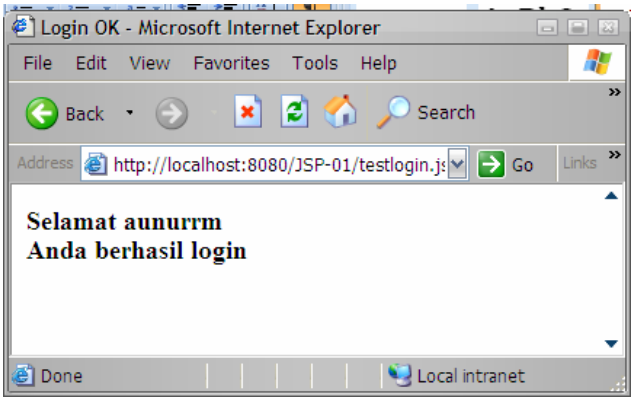
```
<html>
  <head>
    <title>Login OK</title>
  </head>
  <body>
          <H3>
          Selamat <%=request.getParameter("id") %> <BR>
          Anda berhasil login<BR>
          </H3>
  </body>
</html>
```

Jalankan halaman login.html maka kalian akan mendapatkan tampilan seperti pada Gambar 14.20. Isikan sembarang nama dan *password* kemudian tekan tombol *Login*. Kalian tidak akan mendapatkan hasil apa-apa. Nama dan *password* anda akan hilang. Demikian berulang-ulang. Sekarang masukkan

pada nama ‘aunurrm’ dan passwordnya ‘arm’ (tanpa tanda petik). Tekan tombol *Login*, maka kalian akan mendapatkan hasil seperti pada Gambar 14.21.



Gambar 14.20. Hasil eksekusi halaman login.html



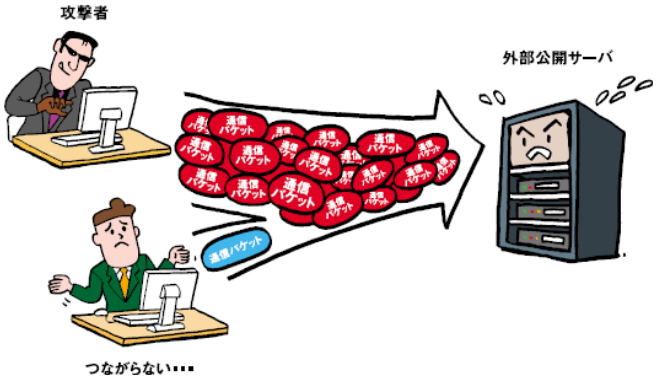
Gambar 14.21. Hasil eksekusi jika nama dan password benar.

Pada contoh di atas file login.html merupakan file untuk menerima input nama dan password. Pada file ini kita menggunakan fitur form pada HTML. Formulir dibentuk dengan menggunakan pasangan tag **<FORM>** dan **</FORM>**. Dua atribut utama dari FORM yang sering digunakan adalah **ACTION** dan **METHOD**.
ACTION menentukan alamat yang akan dijalankan dan menerima semua masukan pada FORM. Jika **ACTION** tidak disebutkan, informasi akan dikirim ke alamat yang sama dengan halaman FORM itu sendiri.
METHOD digunakan untuk menentukan bagaimana informasi dikirim ke alamat yang disebutkan dalam **ACTION**. Nilai yang umum digunakan adalah **GET** dan **POST**. **POST** membuat informasi akan dikirim secara terpisah dengan alamat, sedangkan **GET** akan membuat informasi dikirim menjadi satu dengan alamat yang dituju.
Contoh di atas menggunakan action=”testlogin.jsp” dan metode penyampaiannya adalah *post*. Artinya data pada form akan dikirim ke file dengan nama testlogin.jsp yang berada satu direktori dengan file login.html.

File kedua yang testlogin.jsp adalah file yang digunakan untuk menangkap input kemudian memeriksa apakah input tersebut valid atau tidak. Perintah request.getParameter digunakan untuk menerima input sesuai dengan variabel input yang ditetapkan. Sebagai contoh request.getParameter('nama') berarti akan menangkap isi dari variabel nama dari file login.html. Isi dari nama dan pass ini kemudian di masukkan pada variabel string dengan nama yang sama. Kemudian dicocokkan apakah nama = 'aunurrm' dan pass='arm'. Jika benar maka *action forward* dilakukan yaitu membawa isi yang didefinisikan di *action param* yaitu variabel 'id' ke file loginvalid.jsp. Jika tidak maka *action forward* juga dilakukan tetapi kembali ke file login.html.
File ketiga hanya dipanggil jika syarat nama dan pass dipenuhi. File ini juga menggunakan request.getParameter untuk menangkap variabel 'id'.

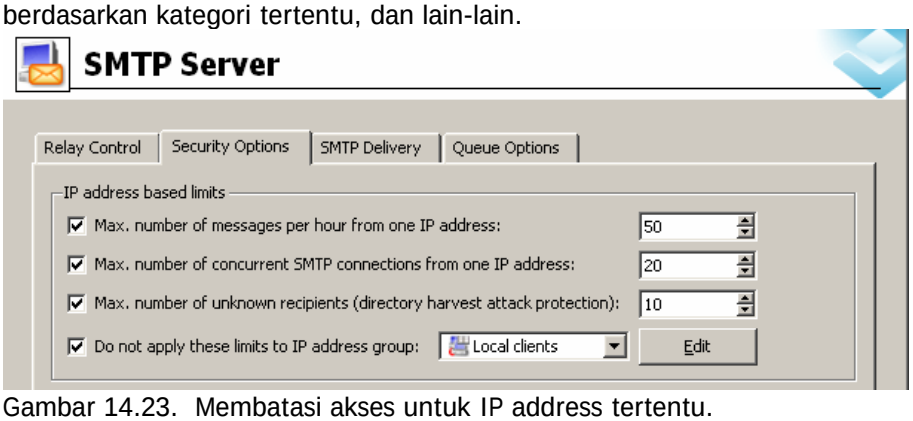
Keamanan dalam Web Dinamis

Web termasuk layanan internet yang paling rentan terhadap ancaman dan pelanggaran. Ada beberapa cara yang bisa dilakukan oleh pihak yang tidak bertanggung jawab terhadap suatu situs, antara lain:
Pemanfaatan *bug* sistem. Para penyerang menggunakan *bug* (kesalahan) yang dikandung oleh sistem operasi, web server, bahasa pemrograman web, atau kode-kode lain yang terinstal di komputer server. Jika ini berhasil ditembus, maka penyerang dapat mengambil alih sistem secara keseluruhan.
Pemanfaatan *bug* pada aplikasi *client*. Kadang-kadang kita tidak terlalu memperhatikan bug pada aplikasi *client*, misalnya adanya fitur berbahaya pada *web browser* yang dapat dijadikan batu loncatan oleh para penyerang.
Pengaksesan tidak sah. Cara ini dilakukan dengan mendapatkan hak akses (*user name* dan *password*) secara tidak sah. Dengan menggunakan perangkat lunak tertentu, seperti *password cracker*, penyerang berusaha mengintip apa yang dilakukan pengguna saat login pada aplikasi web. Setelah memperoleh hak akses tersebut, mereka menggunakan untuk mengakses dan mengganggu system secara tidak sah.
Penyadapan transmisi informasi (*eavesdropping*). Informasi di internet seperti mobil yang berlalu lalang di jalan-jalan. Teknik ini termasuk tingkat tinggi. Dilakukan dengan cara mencegat dan menyadap informasi yang ditransmisikan antara web server dan web browser. Biasanya yang dituju adalah data-data rahasia seperti password, nomor *account*, nomor kartu kredit, dan rekaman transaksi online.
Penyerangan dengan *Denial of Service* (DOS). Bentuk penyerangan ini lebih mengarah pada lingkungan jaringan bukan pada system. Serangan ini ditujukan untuk melumpuhkan suatu system dengan cara membanjiri jaringan dengan trafik yang sangat tinggi dan terus menerus melebihi kapasitas yang didukung. Hal ini mengakibatkan system jaringan berhenti bekerja dan tidak dapat melayani permintaan pengguna.



Gambar 14.22. Penyerangan dengan DOS.

Secara umum tidak ada perangkat lunak yang seratus persen menjamin bebas dari gangguan keamanan. Hal ini karena celah-celah keamanan selalu muncul di sana sini yang memungkinkan penyerangan. Namun ada beberapa langkah yang dapat digunakan untuk mengurangi resiko terserang gangguan, antara lain:
Menjalankan server secara aman, misalnya dengan tidak memberikan kesempatan pada pengguna untuk mengakses melalu *shell*, membuat fasilitas perekam kesalahan (log) dari *web server*.
Menerapkan *permissions* atau hak akses pada direktori dan file secara ketat (buka Bab 4 untuk mempelajari lagi permissions). *Permissions* ini juga harus diterapkan pada file-file konfigurasi web server.
Menuliskan program aplikasi web secara aman. Sering kali kesalahan yang tidak disengaja pada pembuatan aplikasi web menjadikan aplikasi menjadi tidak aman. Ada baiknya aplikasi diperiksa dan diuji berulangkali untuk memastikan celah keamanannya sangat minimal.
Mencegah dan memproteksi informasi dengan cara melakukan enkripsi (penyandian) agar tidak dapat dibaca oleh penyadap.
Mengontrol akses ke web server. Cara ini dilakukan untuk melindungi data-data internal yang tidak bisa diakses dari luar. Beberapa cara dapat dilakukan seperti membatasi akses berdasarkan alamat IP atau domain, membatasi pengguna berdasarkan kategori tertentu, dan lain-lain.



Gambar 14.23. Membatasi akses untuk IP address tertentu.

14.2 RINGKASAN

Pemrograman web merupakan usaha untuk membuat halaman web dengan menggunakan bahasa pemrograman web (script).
Ada dua model pemrograman web yaitu *client-side* dan *server-side.* Bahasa pemrograman untuk membuat web dinamis juga terbagi menjadi dua yaitu *client-side script* dan *server-side script*.
Untuk membangun aplikasi web dinamis, diperlukan persiapan pada web server, bahasa pemrograman web, lokasi penyimpanan web dinamis, dan konfigurasi dari aplikasi.
Java Server Pages (JSP) adalah bahasa scripting untuk web programming yang bersifat server side dan berjalan di Platform Java.
JSP menyediakan empat kategori tag, yaitu directive, elemen scripting, komentar, dan action.
Web termasuk layanan internet yang paling rentan terhadap ancaman dan pelanggaran oleh karena itu upaya pencegahan dan pengamanan web adalah sangat penting.

14.3 SOAL-SOAL LATIHAN

Sebutkan kebutuhan perangkat keras dan lunak untuk pembuatan web dinamis.
Apakah perbedaan client-side script dan server side script?
Cobalah berselancar ke internet. Kemudian pada beberapa situs yang kalian kunjungi, cermati pada halaman URL nya (lihat pada bagian Address Bar pada web browser). Bisakah kalian menentukan teknologi pemrograman web apa yang digunakan.
Buatlah aplikasi sederhana dengan menggunakan JSP untuk menghitung gaji karyawan. Buat dua halaman, yang pertama adalah halaman untuk memasukkan input, yaitu nama bulan, nama karyawan, jumlah jam kerja pada bulan tersebut dan upah per jam. Sedang halaman yang kedua adalah, untuk menghitung upah karyawan (jam kerja sebulan x upah per jam) sekaligus menampilkan nama dan total gaji yang diperoleh pada bulan tersebut.
Kembangkan soal No 4, dengan cara menambahkan halaman login sebelum masuk halaman memasukkan input. Jika seseorang berhasil login, maka halaman memasukkan input akan tampil, tetapi jika tidak maka akan menuju ke halaman baru, yaitu untuk halaman mendaftar sebagai user.

Daftar Pustaka

Anonymous. 2004. Guide to the Software Engineering Body of Knowledge (SWEBOK). The Institute of Electrical and Electronics Engineers, Inc.

Balter, A. 2006. Sams Teach Yourself Microsoft® SQL Server™ 2005 Express in 24 Hours. Sams.

Bass, L., P. Clements, and R. Kazman. 2003. Software Architecture in Practice. 2nd Edition. Addison-Wesley.

Cormen, T.H. 2001. Introduction to Algorithm: Second Edition. The MIT Press.

Deek, FP., J.A.M. McHugh, and O.M. Eljabiri. 2005. Strategic software engineering : An Interdisciplinary Approach. Auerbach Publications.

den Haan, P., L. Lavandowska, S.N. Panduranga, and K. Perrumal. 2004. Beginning JSP 2: From Novice to Professional. Apress.

Dobson, R. 1999. Programming Microsoft Access 2000: The Developer's Guide to Harnessing the Power of Access. Microsoft Press.

Felleisen, M, R.B. Findler, M. Flatt, and S. Krishnamurthi. 2001. How to Design Programs; An Introduction to Computing and Programming. The MIT Press.

Kak, A.C. 2003. Programming With Objects: A Comparative Presentation of Object Oriented Programming with C++ and Java. John Wiley & Sons, Inc.

Kaisler, S.H. 2005. Software Paradigm. John Wiley & Sons, Inc.

Kennedy, B. and C. Musciano. 2006. HTML & XHTML: The Definitive Guide, 6th Edition. O'Reilly.

Lafore, R. 1998. Data Structures & Algorithm in Java. Waite Group Press.

Laurie, B and P. Laurie. 2001. Apache: The Definition Guide. 2nd Edition. O'Reilly and Associates, Inc.

Leffingwell, D. and D. Widrig. 2003. Managing Software Requirements: A Use Case Approach. 2nd Edition. Addison-Wesley.

Lischner, R. 2000. Delphi in a Nutshell. O'Reilly and Associates, Inc.

Luckey, T. and J. Phillips. 2006. Software Project Management for Dummies. Wiley Publishing, Inc.

McConnel, S. 2003. Professional Software Development: Shorter Schedules, Higher Quality Products, More Successful Projects, Enhanced Careers. Addison-Wesley.

Meyer, B. 2000. Object Oriented Software Construction. 2nd Edition. ISE, Inc.

Musciano, C. and B. Kennedy. 2002. HTML and XHTML: The Definition Guide. 4th Edition. O'Reilly and Associates, Inc.

Navarro, A. 2001. Effective Web Design. 2nd Edition. SYBEX, Inc.

Powell, G. 2006. Beginning Database Design. Wiley Publishing, Inc.

Riordan, R.M. 2005. Designing Effective Database Systems. Addison Wesley Professional.

Robbins, J. N. 2006. Web Design in a Nutshell, 3rd Edition. O'Reilly.

Suehring, S. 2002. MySQL Bible. Wiley Publishing, Inc.

Taylor, D.A. 1998. Object Technology: A Manager's Guide. Addison-Wesley.

Van Roy, P and S. Haridi. 2004. Concepts, Techniques, and Models of Computer Programming. The MIT Press.

Lampiran 1
Daftar Istilah / Glosari

Abstraction
Merupakan prinsip penyederhanaan dari sesuatu yang kompleks dengan cara memodelkan kelas sesuai dengan masalahnya

Algoritma
Urutan langkah-langkah logis penyelesaian masalah yang disusun secara sistematis

Array
Struktur data yang menyimpan sekumpulan elemen yang bertipe sama

Atribut
Karakteristik atau ciri yang membedakan antara entitas satu dengan entitas yang lainnya

Authentication
Proses memeriksa keabsahan seseorang sebagai user (pengguna) pada suatu system (misalnya pada DBMS)

Basic Input/Output System (BIOS)
Kode-kode program yang pertama kali dijalankan ketika komputer dinyalakan (booting)

Basis data (database)
Kumpulan dari data yang saling berhubungan satu dengan yang lainnya, tersimpan dalam perangkat keras komputer dan digunakan perangkat lunak untuk memanipulasinya

Command Line Interface (CLI)
Antar muka pengguna dengan model perintah-perintah teks

Compiler
Penerjemah bahasa pemrograman tingkat tinggi ke bahasa mesin dengan cara sekaligus seluruh kode program. Prosesnya disebut kompilasi.

Component Object Model (COM)
Infrastruktur yang disediakan oleh Visual Basic untuk mengakses obyek-obyek atau kontrol-kontrol lain sepanjang punya antar muka yang dapat diakses oleh Visual Basic.

Constraint
Batasan-batasan dari masalah

Control
Aktivitas monitoring dan evaluasi terhadap feedback untuk menentukan apakah system telah bekerja dengan baik atau tidak

Counter
Variable pencacah yang digunakan dalam struktur algoritma pengulangan

Database Management System (DBMS)
Perangkat Lunak yang khusus / spesifik ditujukan untuk pengelolaan basis data

Disk Operating System (DOS)
Salah satu sistem operasi lama berbasis CLI

Elektronika
Ilmu yang mempelajari alat listrik **arus lemah** yang dioperasikan dengan cara mengontrol aliran elektron atau partikel bermuatan listrik dalam suatu alat

Entitas
Individu yang mewakili sesuatu yang nyata (eksistensinya) dan dapat dibedakan dari sesuatu yang lain

Extensible Hypertext Markup Language (XHTML)
HTML versi terakhir (4.01) yang ditulis ulang dengan dengan aturan-aturan yang lebih ketat mengacu pada XML

Extensible Markup Language (XML)
Sekumpulan aturan untuk menyusun bahasa *markup*

Feedback
Data tentang kinerja sistem

Flowchart
Skema/bagan (*chart*) yang menunjukkan aliran (*flow*) di dalam suatu program secara logika

Gejala
Signal atau tanda terjadinya suatu masalah

Gerbang logika
blok-blok penyusun dari perangkat keras elektronik

Graphical User Interface (GUI)
Antar muka pengguna dengan model grafis

Identifier

Nama dari suatu variable atau konstanta

Ilmu komputer
Suatu studi sistematis pada proses-proses algoritma yang menjelaskan dan mentransfor-masikan informasi

Inheritance atau pewarisan
Prinsip pewarisan sifat dari orang tua ke anak atau turunannya yang diterapkan pada kelas

Inisialisasi
Instruksi yang dilakukan pertama kali pada suatu variabel atau ekpresi pemrograman

Input
Elemen-elemen yang masuk ke dalam system

Integrated Developement Environment (IDE)
Lingkungan pengembangan aplikasi terintegrasi. Perangkat lunak untuk membantu mempermudah pembuatan aplikasi komputer

Interpreter
Penerjemah bahasa pemrograman tingkat tinggi ke bahasa mesin dengan cara satu per satu baris dibaca dan langsung diterjemahkan

Kardinalitas
Jumlah maksimum entitas pada suatu himpunan entitas yang dapat berelasi dengan entitas pada himpunan entitas yang lain

Konstanta
Variabel yang nilai datanya bersifat tetap dan tidak bisa diubah.

Loop
Proses pengulangan suatu perintah

Masalah (problem)
Perbedaan antara situasi aktual dan situasi yang diharapkan atau perbedaan antara kondisi sekarang dengan target atau tujuan yang diinginkan

Model
Penyederhanaan dari suatu system atau Tiruan dari suatu sistem dengan sedikit atau banyak penyederhanaan

Multi-tasking

Kemampuan sistem operasi untuk menjalankan beberapa tugas / aplikasi secara bersamaan

Multi-user
Kemampuan system operasi untuk dijalankan oleh pengguna yang berbeda pada waktu bersamaan

Output
Perpindahan elemen-elemen yang dihasilkan dari proses perubahan ke tujuan yang diinginkan

Pemecahan masalah
Sebuah proses dimana suatu situasi dianalisa kemudian solusi-solusi dibuat bila ditemukan ada masalah dengan cara pendefinisian, pengurangan atau penghilangan, atau pencegahan masalah

Pemrograman Berorientasi Obyek (*Object Oriented Programming – OOP*)
Paradigma pemrograman yang menggunakan obyek dan interaksinya untuk merancang aplikasi dan program komputer

Pemrograman web
Usaha untuk membuat halaman web dengan menggunakan bahasa pemrograman web (*script*)

Perangkat lunak
Seluruh instruksi yang digunakan untuk memproses informasi

Permissions
Proses untuk menentukan apa yang bisa dilakukan seorang pengguna pada suatu sistem

Pointer
Variabel yang menyimpan alamat pada memori komputer

Polymorphism
Kemampuan dari suatu obyek untuk mempunyai lebih dari satu bentuk

Programmer
Seseorang yang bekerja membuat program komputer

Prosedur
- Instruksi yang dibutuhkan oleh pengguna dalam memproses informasi
- Sekumpulan perintah yang merupakan bagian dari program yang lebih besar yang berfungsi mengerjakan suatu tugas tertentu

Proses
Perubahan atau transformasi input menjadi output

Prototyping
Salah satu pendekatan dalam pengembangan perangkat lunak yang secara langsung mendemonstrasikan bagaimana sebuah perangkat lunak atau komponen-komponen perangkat lunak akan bekerja dalam lingkungannya sebelum tahapan konstruksi aktual dilakukan

Pseudocode
Cara penulisan algoritma dengan menggunakan kode-kode yang mirip dengan bahasa pemrograman

Query
Permintaan atau pencarian pada data-data tertentu pada suatu basis data

Record
Baris data dari suatu tabel

Rekayasa Perangkat Lunak
suatu disiplin ilmu yang membahas semua aspek produksi perangkat lunak, mulai dari tahap awal yaitu analisa kebutuhan pengguna, menentukan spesifikasi dari kebutuhan pengguna, disain, pengkodean, pengujian sampai pemeliharaan sistem setelah digunakan

Relationship atau relasi
Hubungan yang terjadi antara sejumlah entitas

Sistem
Kumpulan dari elemen-elemen yang saling berinteraksi untuk mencapai tujuan tertentu

Sistem basis data
Kumpulan elemen-elemen seperti basis data, perangkat lunak, perangkat keras, dan manusia yang saling berinteraksi untuk mencapai tujuan yaitu pengorganisasian data.

Software
Lihat Perangkat Lunak

Software Engineering
Lihat Rekayasa Perangkat Lunak

Solusi
Bagian akhir atau output dari proses pemecahan masalah.

Stored procedure

Potongan kode program yang dapat menerima parameter input dan menghasilkan satu atau lebih parameter output dan digunakan untuk operasi-operasi basis data

Structured Query Language (SQL)
Bahasa query terstruktur untuk mengelola basis data

Strategi pemecahan masalah
Metode atau pendekatan yang digunakan seseorang ketika menghadapi masalah

Struktur algoritma
Cara atau urutan untuk membuat suatu algoritma

Tipe data
Jenis data yang dapat diolah oleh komputer untuk memenuhi kebutuhan dalam pemrograman komputer

Trigger
Tipe khusus dari stored procedure yang akan dieksekusi ketika suatu kejadian muncul

Variabel
Tempat dimana kita dapat mengisi atau mengosongkan nilainya dan memanggil kembali apabila dibutuhkan pada suatu program

View
Tabel virtual yang isinya berdasarkan pada query yang dilakukan pada basis data.

Web browser
Perangkat lunak yang berfungsi menerjemahkan kode-kode HTML menjadi tampilan yang kita kehendaki

Web dinamis
Halaman-halaman web yang isi dan informasinya berubah-ubah sesuai dengan permintaan pengguna

Web server
Perangkat lunak yang bertindak melayani permintaan-permintaan *client* terhadap halaman-halaman *web* tertentu

Web statis
Halaman-halaman web yang isi dan informasinya tidak berubah-ubah

Lampiran 2
Daftar Alamat Situs

Berikut ini daftar alamat situs-situs internet yang penting dan digunakan sebagai rujukan dalam buku ini.

Alamat	Keterangan
http://www.apache.org	Situs resmi web server Apache. Situs ini menyediakan kode sumber Apache dan file-file binary Apache yang siap diinstall di berbagai platform sistem operasi. Selain itu juga menyediakan dokumentasi Apache yang lengkap.
http://www.borland.com	Situs resmi Borland. Borland merupakan perusahaan perangkat lunak yang memproduksi Borland Delphi, Borland JBuilder, Turbo Pascal, Turbo Delphi, Borland C++ dan lain-lain.
http://www.debian.org	Situs resmi distribusi linux Debian.
http://www.eclipse.org	Situs resmi proyek eclipse, perangkat pengembang terpadu yang mendukung banyak bahasa pemrograman.
http://www.google.com	Situs resmi search engine Google.
http://www.ilmukomputer.com	Situs berbahasa Indonesia yang menyediakan dokumen-dokumen untuk belajar berbagai sub bidang dalam ilmu computer.
http://www.javasoft.com	Situs resmi yang diluncurkan Sun Microsystem dan berisi dokumentasi dan informasi online tentang bahasa pemrograman Java.
http://www.kambing.vlsm.org	Situs dengan server local di Indonesia. Situs ini menyediakan file-file iso dari berbagai jenis distribusi linux dan dapat didownload secara bebas. Selain itu situs ini juga sebagai mirror dari berbagai distribusi linux dan aplikasi yang berjalan di linux.
http://www.linuxdoc.org	Situs yang berisi dokumentasi bebas tentang linux. Sumber informasi online yang sangat bagus untuk mempelajari linux

http://www.microsoft.com	Situs resmi Microsoft. Microsoft merupakan perusahaan perangkat lunak yang memproduksi system operasi keluarga Windows, IDE Microsoft Visual Studio, Microsoft Office, Microsoft SQL Server, dan lain-lain.
http://www.mysql.com	Situs resmi MySQL Database Software. Situs ini menyediakan file-file instalasi MySQL untuk berbagai platform sistem operasi. Selain itu juga menyediakan dokumentasi MySQL yang lengkap.
http://www.netbeans.org	Situs resmi IDE Netbeans, perangkat lunak pengembang aplikasi Java
http://www.php.net	Situs resmi bahasa pemrograman dan interpreter PHP. Situs ini menyediakan kode sumber dan file-file instalasi PHP untuk berbagai platform sistem operasi. Selain itu juga menyediakan dokumentasi PHP yang lengkap.
http://www.w3.org	Situs resmi The World Wide Web Consortium (W3C). W3C adalah konsorsium yang menentapkan standar dalam teknologi internet, terutama tentang HTML, XML, CSS, XHTML dan teknologi lain. Dokumentasi tentang teknologi tersebut dapat dijumpai di situs ini.

Lampiran 3
Fungsi-fungsi Built-in Pada Visual Basic

IsNumeric(ekspresi)

Fungsi ini digunakan untuk menguji apakah suatu ekspresi menghasilkan nilai numeric atau bukan. Nilai yang dikembalikan adalah Boolean.

IsEmpty(ekspresi)

Fungsi untuk memeriksa apakah suatu ekspresi telah berisi nilai atau tidak. Nilai yang dikembalikan adalah Boolean..

IsNull(ekspresi)

Fungsi untuk memeriksa apakah suatu ekspresi mengandung data yang tidak valid, biasanya digunakan untuk memeriksa isi field recordset.

IsArray(varname)

Fungsi untuk memeriksa apakah suatu variabel adalah suatu array.

IsDate(ekspresi)

Fungsi untuk memeriksa apakah suatu ekspresi dapat dikonversi ke date.

IsError(ekspresi)

Fungsi untuk memeriksa apakah suatu ekspresi adalah nilai error

IsObject(ekspresi)

Fungsi untuk memeriksa apakah suatu ekspresi mengacu pada suatu OLE Automation object.

IsMissing(argname)

Fungsi untuk memeriksa apakah suatu argumen optional pada procedure ada dilewatkan atau tidak

CBool(ekspresi)

Konversi suatu ekspresi ke Boolean

CByte(ekspresi)

Konversi ekspresi ke Byte

CCur(ekspresi)

Konversi suatu ekspresi ke Currency

CDate(date)

Konversi suatu ekspresi ke date

CDbl(ekspresi)

Konversi suatu ekspresi ke Double

CInt(ekspresi)

Konversi suatu ekspresi ke Integer

CLng(ekspresi)

Konversi suatu ekspresi ke Long

CSng(ekspresi)

Konversi suatu ekspresi ke single

CStr(ekspresi)

Konversi suatu ekspresi ke string

CVar(ekspresi)

Konversi suatu ekspresi ke Variant

Asc(string)

Fungsi untuk menampilkan kode character dari huruf pertama di suatu string.

Chr(charcode)

Fungsi untuk menampilkan karakter dari suatu kode karakter

Format(ekspresi[, format[, hariPertamaDariMinggu[, mingguPertamaDariTahun]]])

Memformat suatu ekspresi berdasarkan ekspresi format

Hex(number) dan Oct(number)

Menampilkan string yang mewakili Octal atau Hexa dari suatu bilangan

Str(number)

Menampilkan string yang mewakili suatu angka.

Val(string)

Menampilkan angka yang terkandung dalam suatu string.

Now

Mengembalikan suatu Variant (Date) yang menunjukkan tanggal dan waktu berdasarkan sistem komputer.

Time

Mengembalikan waktu sistem sekarang

Timer

Mengembalikan suatu bilangan yang menunjukan jumlah detik sejak tengah malam

Date

Mengembalikan tanggal sistem sekarang

Time = Time dan Date = Date

Mengatur waktu atau tanggal sistem

Untuk sistem yang menjalankan Microsoft Windows 95, tanggal yang dibutuhkan harus berupa tanggal dari 1 Jan 1998 sampai 31 Des 2099. Untuk sistem yang menjalankan Microsoft Windows NT, tanggal yang dibutuhkan harus berupa tanggal dari 1 Jan 1980 sampai 31 Desember 2079.

Hour(time), Minute(time) dan Second(time)

Mengembalikan suatu Variant (Integer) berupa bilangan 0 s/d 23 untuk jam, 0 s/d 59 untuk menit, dan 0 s/d 59 untuk detik.

Day(date), Month(date), dan Year(date)

Mengembalikan suatu Variant (Integer) berupa bilangan 1 s/d 31 untuk bulan, 1 s/d 12 untuk bulan, dan tahun.

Daftar Gambar

Daftar Tabel

ISBN 978-979-060-007-2
ISBN 978-979-060-010-2

Buku ini telah dinilai oleh Badan Standar Nasional Pendidikan (BSNP) dan telah dinyatakan layak sebagai buku teks pelajaran berdasarkan Peraturan Menteri Pendidikan Nasional Nomor 45 Tahun 2008 tanggal 15 Agustus 2008 tentang Penetapan Buku Teks Pelajaran yang Memenuhi Syarat Kelayakan untuk digunakan dalam Proses Pembelajaran.

HET (Harga Eceran Tertinggi) Rp. 13,794.00