

Perkembangan Microsoft SQL Server

Microsoft SQL Server diperkenalkan pada tahun 1990 untuk platform Microsoft OS/2 dalam kerjasamanya dengan Sybase. Produk ini berasal dari Sybase SQL Server 4.x untuk platform Unix. Dengan adanya Windows NT, muncul inisiatif untuk membangun SQL Server versi Windows NT sehingga dihasilkan Microsoft SQL Server versi 4.2 untuk platform Windows NT. Kerjasama dengan Sybase masih berlanjut dan diluncurkan SQL Server 6.0 pada tahun 1995 dan setahun kemudian SQL Server versi 6.5 diluncurkan.

SQL Server 6.5 memperbarui kemampuan transaksi dan menjadi produk database client/server yang banyak dipakai pada platform Windows NT. Untuk memenuhi kebutuhan pengguna yang makin meningkat, maka SQL Server perlu didisain ulang dan kerjasama dengan Sybase dihentikan. Kemudian Microsoft mengembangkan SQL Server 7.0 yang difokuskan pada tiga area yaitu : easy to use, scalability dan data warehousing. Pada tahun 2000, kemudian Microsoft meluncurkan SQL Server 2000. Di tahun 2005 ini, Microsoft mengeluarkan produk SQL Server versi terbarunya yaitu Microsoft SQL Server 2005 seiring dengan dilauncingnya Microsoft Visual Studio 2005 beta 2.

Pada makalah ini akan difokuskan ke masalah SQL Server 2000. SQL Server 2000 mempunyai beberapa edisi. Setiap edisi memberikan performansi dan harga yang berbeda pula, sehingga pemakaiannya dapat disesuaikan dengan kebutuhan si pengguna. Adapun edisi yang dimaksud adalah :

Edisi	Fitur
Enterprise	Merupakan edisi terlengkap yang mendukung hingga 32 CPU dan RAM hingga 64 GB. Cocok digunakan untuk perusahaan besar yang membutuhkan performa yang maksimal.
Standard	Cocok digunakan untuk perusahaan kecil dan menengah. Edisi ini mampu mendukung hingga 4 CPU dan RAM hingga 2 GB

Personal	Edisi ini berisi alat bantu manajemen lengkap dan fungsi-fungsi umum dari edisi Standard serta cocok digunakan untuk keperluan individu. Edisi ini dijalankan pada sistem operasi yang bukan server, seperti Windows NT Workstation 4.0, Windows 9x, Windows 2000 Professional dan Windows XP. Edisi ini mendukung 2 processor dan performansinya dioptimalkan untuk pemakai tunggal dan workgroup kecil serta mampu menangani lima user yang konkuren.
Developer	Diluncurkan bersama-sama dengan Microsoft Visual Studio dan hanya disarankan untuk pengembangan aplikasi yang memakai Visual Studio
Desktop Engine (MSDE)	Edisi ini mempunyai fasilitas mesin database dasar dari SQL Server 2000. Edisi ini tidak mempunyai interface, alat bantu manajemen, kemampuan analisis, penggabungan replikasi, online book. Edisi ini membatasi ukuran database dan beban user
Windows CE	Merupakan versi SQL Server 2000 untuk alat-alat yang menjalankan Windows CE. Versi Windows CE biasa digunakan untuk PDA dan Pocket PC.

Arsitektur SQL Server 2000

SQL Server 2000 dikomersilkan pada tahun 2000 dan mempunyai desain yang sudah modern. SQL Server 2000 adalah sebuah mesin database client/server yang berbeda dengan database komputer tunggal tradisional yang memakai sistem pemakaian file secara bersama-sama (misalnya Dbase, Microsoft Jet, Microsoft Visual FoxPro). Database sistem memakai file secara bersama-sama bergantung pada sebuah proses tunggal per user untuk memanipulasi data pada file yang dipakai bersama pada server jaringan. Dalam lingkungan multi user akan muncul berbagai masalah, yaitu pengontrolan konkurensi yang memakai mekanisme locking pada lapisan network. Fasilitas securitas untuk sistem database ini hanya dibatasi pada izin untuk membaca dan menulis data pada jaringan, sehingga user yang ingin melakukan kecurangan dapat memakai alat

bantu lain untuk memanipulasi data. Oleh karena itu pengontrolan data menjadi sulit.

Bagi pengembang database, SQL Server kompatibel dengan beberapa data access interface yang digunakan dalam Development Tool seperti pada Visual Basic, Visual C++, Power Builder, Delphi, Visual FoxPro dan sebagainya. Database SQL Server dapat diakses dengan menggunakan Microsoft Jet Engine and Data Access Object (DAO), Remote Data Object (RDO), ActiveX Data Object (ADO), OLEDB, ODBC, SQL Server built-in Library dan interface dari third party lainnya.

Sistem database client/server seperti SQL Server 2000 memakai sejumlah proses server untuk memanipulasi data, dan mengharuskan proses client berhubungan dengan proses server menggunakan mekanisme **IPC** (*inter-process communication*) lokal atau remote, misalnya socket TCP/IP. Proses server adalah aplikasi server yang memproses perintah-perintah SQL. Proses server juga menangani konkurensi dengan memakai mekanisme locking yang lebih canggih dari sistem file jaringan yang dipakai secara bersama-sama. Selain itu server juga menangani masalah securitas dengan melakukan teknik autentifikasi pada setiap pemakai. Setelah proses server menjalankan perintah-perintah, hasilnya akan dikirim kembali ke proses client melalui mekanisme IPC. Dengan cara ini, sistem client/server memberikan pelayanan pengaksesan yang lebih baik pada data yang dipakai bersama-sama oleh banyak user.

SQL Server 2000 memberikan bahasa dan antarmuka (interface) yang baik untuk pemrograman dan komunikasi pada server. Transact-SQL merupakan bahasa pemrograman server yang merupakan superset dari ANSI-SQL. ANSI-SQL mendefinisikan empat perintah dasar untuk manipulasi data yaitu : SELECT, INSERT, UPDATE, DELETE dan sejumlah perintah untuk mendefinisikan struktur database. Transact-SQL menambahkan beberapa hal pada ANSI-SQL. Penambahan tersebut adalah konstruksi pemrograman yang memungkinkan pemakaian *stored procedure* untuk mengubah data dan trigger yang akan dijalankan karena terjadi event tertentu.

Istilah client/server dipakai untuk menggambarkan arsitektur *two-tier* untuk aplikasi enterprise yang mempunyai client “berat” karena mengimplementasikan



interface user dan proses bisnis yang rumit dan dihubungkan ke sebuah database backend yang canggih seperti SQL Server 2000.

Meskipun arsitektur ini berhasil untuk banyak aplikasi, kemudian muncul konsep baru yaitu three-tier atau n-tiered. Dengan cara ini, client hanya mengimplementasikan interface user, sedangkan proses bisnis yang rumit dijalankan oleh aplikasi server pada middle-tier. Aplikasi server tersebut berkomunikasi dengan database.

Jadi, istilah client/server menunjukkan sebuah proses client yang berhubungan dengan proses server, tanpa memandang di tier mana proses client berada. Misalnya sebuah proses client dapat berupa program middle tier yang menjalankan validasi kartu kredit dan proses server adalah SQL Server. SQL Server juga dapat berfungsi sebagai aplikasi khusus yang menangani operasi database untuk proses client. Proses client dan server dapat bersama-sama berada di komputer yang sama atau berkomunikasi pada jaringan menggunakan mekanisme IPC.


Arsitektur Client/Server

Microsoft SQL-Server dirancang agar efektif dalam sejumlah lingkungan :

-  Sebagai 2-tier atau multi-tier Client/Server Database System,
-  Sebagai Desktop Database System.

Client/Server Database System

System client/server dikembangkan sedemikian rupa dimana database ditempatkan pada suatu komputer pusat, disebut sebagai server, dan dibagi-pakai (share) kepada sejumlah user. User melakukan akses ke server melalui suatu aplikasi client atau aplikasi server :

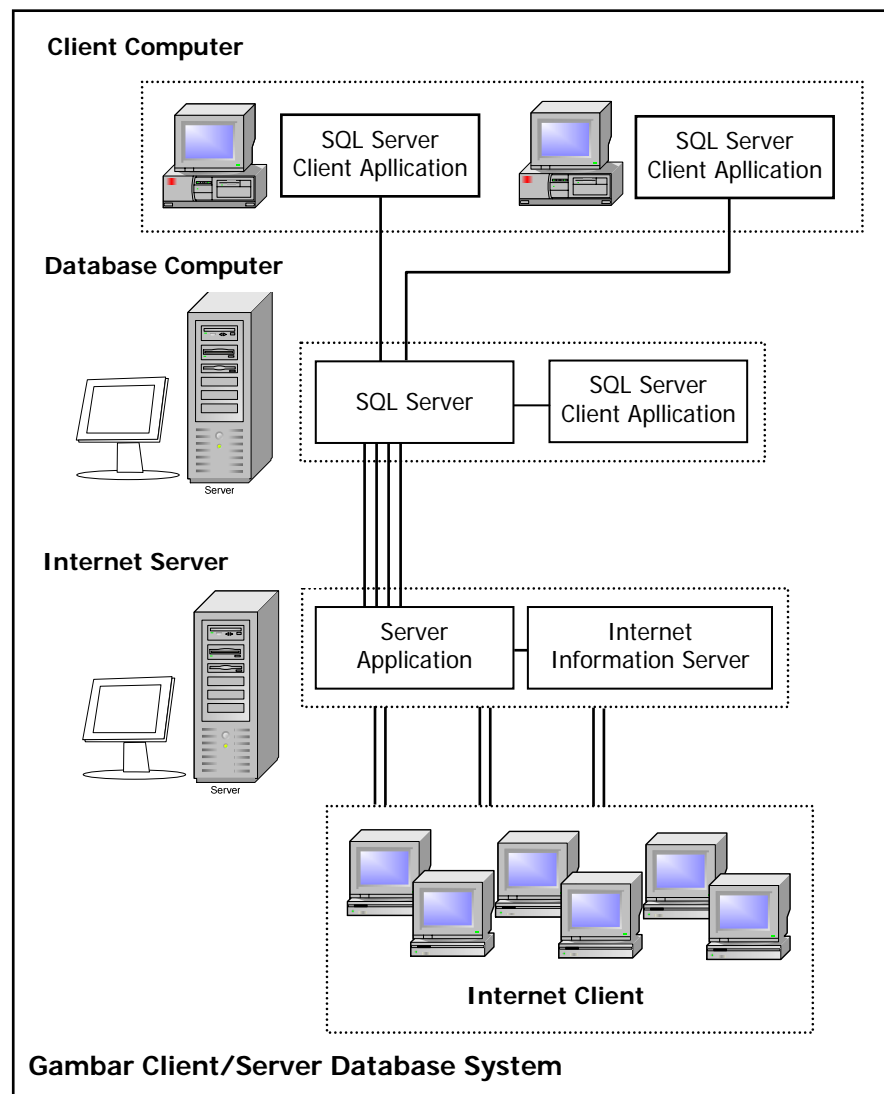
-  Dalam 2-tier client/server system, user menjalankan suatu aplikasi pada lokal komputer, disebut sebagai client, yang terkoneksi melalui network ke server yang menjalankan SQL-Server. Aplikasi Client yang menjalankan business logic dan code untuk menampilkan output kepada user, dikenal sebagai *thin client*.

■ Dalam multi-tier client/server system, logic aplikasi client dijalankan pada 2 lokasi :

- Thin client berjalan pada komputer lokal dan focus pada penampilan (display) hasil-hasil ke user.
- Business logic ditempatkan pada aplikasi server yang dijalankan pada suatu server. Thin client memanggil fungsi-fungsi dan aplikasi server, dimana server memiliki kemampuan *multithread* yang bekerja dengan beberapa user pada saat yang sama. Server aplikasi membuka suatu koneksi dengan server database. Server database dapat merupakan server yang sama atau dapat juga merupakan server database terpisah yang terkoneksi melalui jaringan.

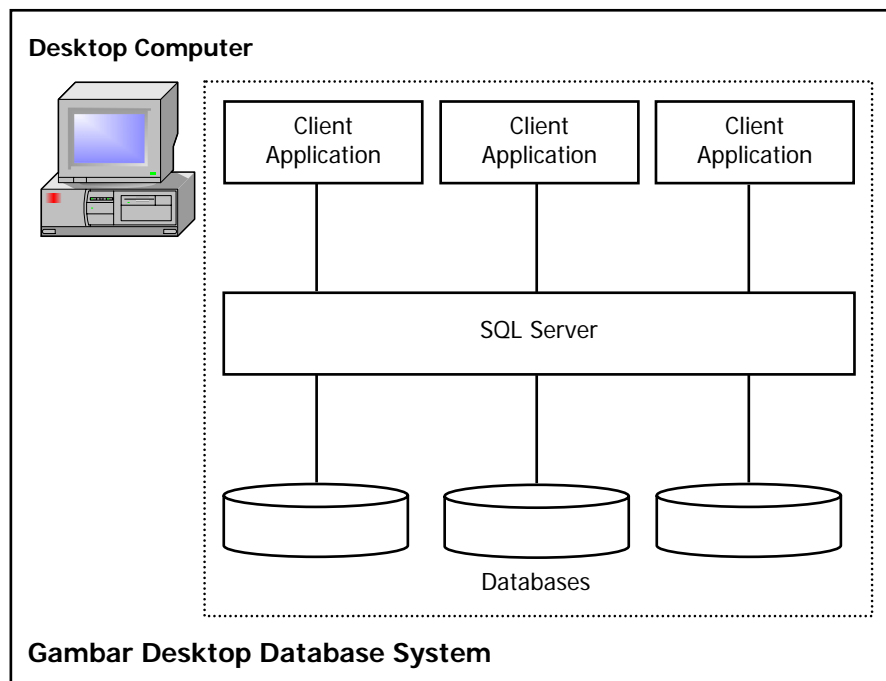
■ Memiliki data yang tersimpan dan terkelola dalam suatu lokasi memberikan beberapa keuntungan :

- Setiap item data tersimpan dalam suatu lokasi terpusat dimana semua user bekerja dengannya. Tidak ada data yang disimpan secara terpisah membuat user yakin bahwa mereka bekerja dengan informasi yang sama.
- Prosedur perusahaan dan keamanan dapat ditetapkan sekali pada server dan akan berlaku sama terhadap semua user, Ini dapat dilakukan melalui pemanfaatan constrain, stored procedure, dan trigger. Juga dapat dilakukan pada aplikasi server.
- Suatu relasi database pada server dapat mengoptimalkan network traffic dengan memberikan data hanya jika dibutuhkan suatu aplikasi.
- Biaya hardware dapat diminimalisasi. Karena data tidak disimpan pada setiap client, client tidak harus memiliki disk space yang besar. Client juga tidak membutuhkan kemampuan pemrosesan untuk mengelola data secara lokal, sementara server juga tidak perlu melakukan proses penampilan data tersendiri.
- Tugas pemeliharaan seperti backup data dan restore data menjadi sederhana karena dapat difokuskan pada server database.



Desktop Database System

SQL-Server juga dapat digunakan pada aplikasi yang membutuhkan database stand alone dan tersimpan secara lokal pada komputer client. SQL-Server dapat mengkoordinasi sendiri secara dinamis untuk berjalan efektif dengan resource yang tersedia pada client, tanpa memerlukan administrator database tersendiri. Pembuat aplikasi dapat menyertakan SQL-Server sebagai komponen data storage pada aplikasi mereka.

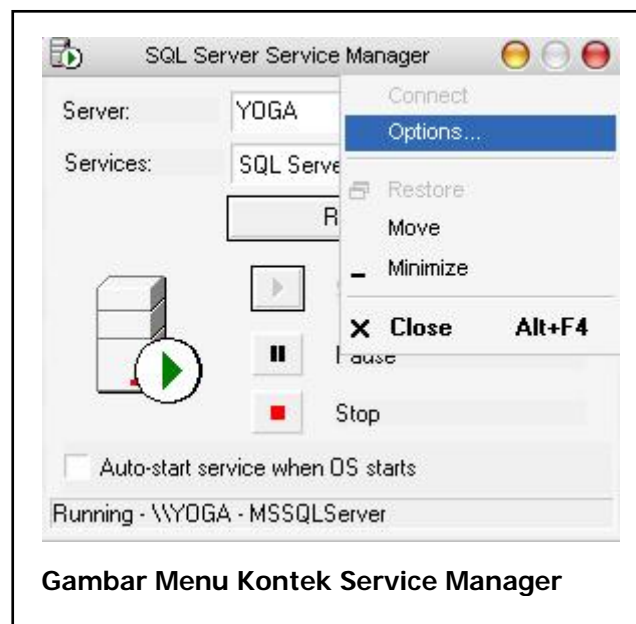


Service Manager

Utilitas Service Manager dipakai untuk menjalankan (start) atau memberhentikan (stop dan pause) komponen-komponen server. Komponen-komponen tersebut dijalankan sebagai service pada Microsoft Windows NT, Windows 9x, dan Windows XP sebagai program executable yang terpisah.

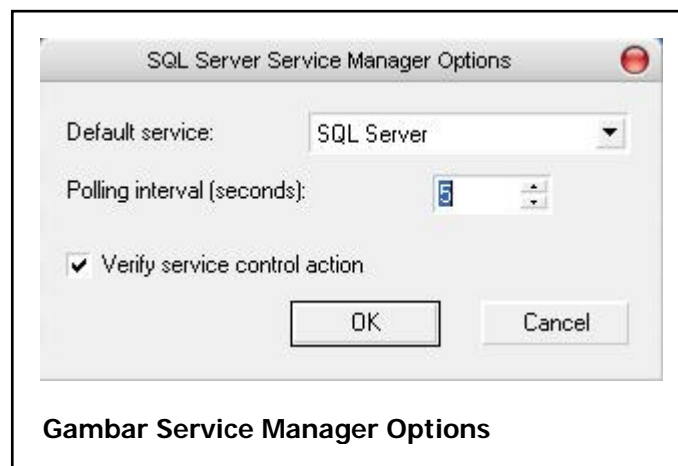


Field Server berisi nama server yang sedang dimonitor. Kotak Services menampilkan servis-servis yang ada serta tampilan grafis dari status servis. Jika sebuah service sedang aktif, ditampilkan tanda (icon) berwarna panah hijau. Service Manager mempunyai beberapa fungsionalitas yang tersembunyi. Klik kanan pada title bar akan menampilkan menu aplikasi dengan dua menu tambahan, yaitu *Connect* dan *Options*.



Gambar Menu Kontek Service Manager

Jika kita memilih sebuah komputer remote pada kotak server, kita dapat terhubung dengan menu Connect bukan dengan memilih pada kotak Services. Jika kita memilih Options dari menu konteks, maka akan ditampilkan interface dialog Option.

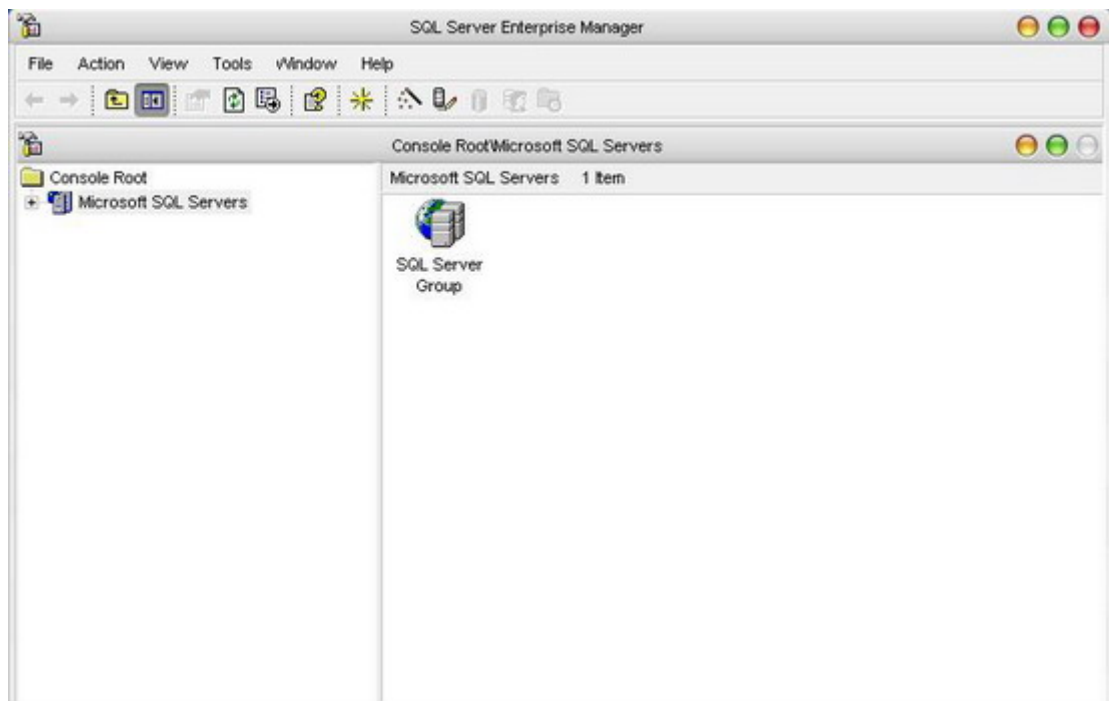


Gambar Service Manager Options

Enterprise Manager

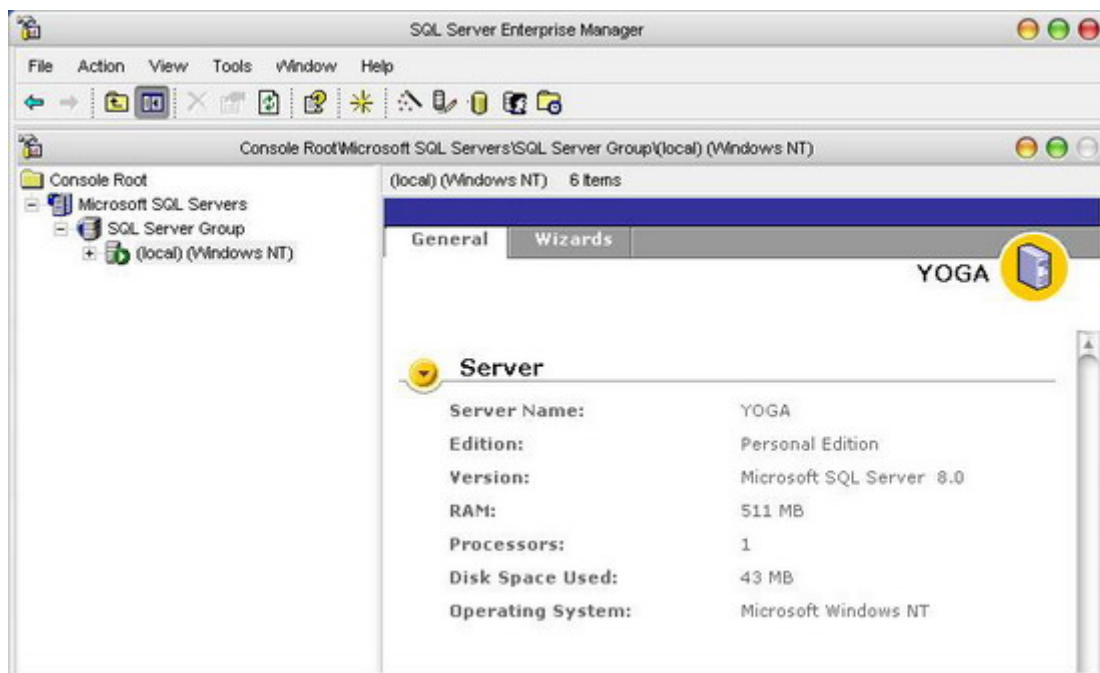
Enterprise Manager adalah alat bantu administratif yang bersifat Graphical User Interface (GUI). Hal-hal yang dapat kita lakukan melalui Enterprise Manager adalah :

- ❏ Mendefinisikan kelompok-kelompok server yang menjalankan SQL Server,
- ❏ Mendaftarkan sebuah server ke dalam sebuah group,
- ❏ Membuat database, objek, login user, dan hak-hak pada setiap server,
- ❏ Memanggil Query Analyzer untuk membuat dan mengeksekusi perintah SQL,
- ❏ Memanggil berbagai wizard yang tersedia.

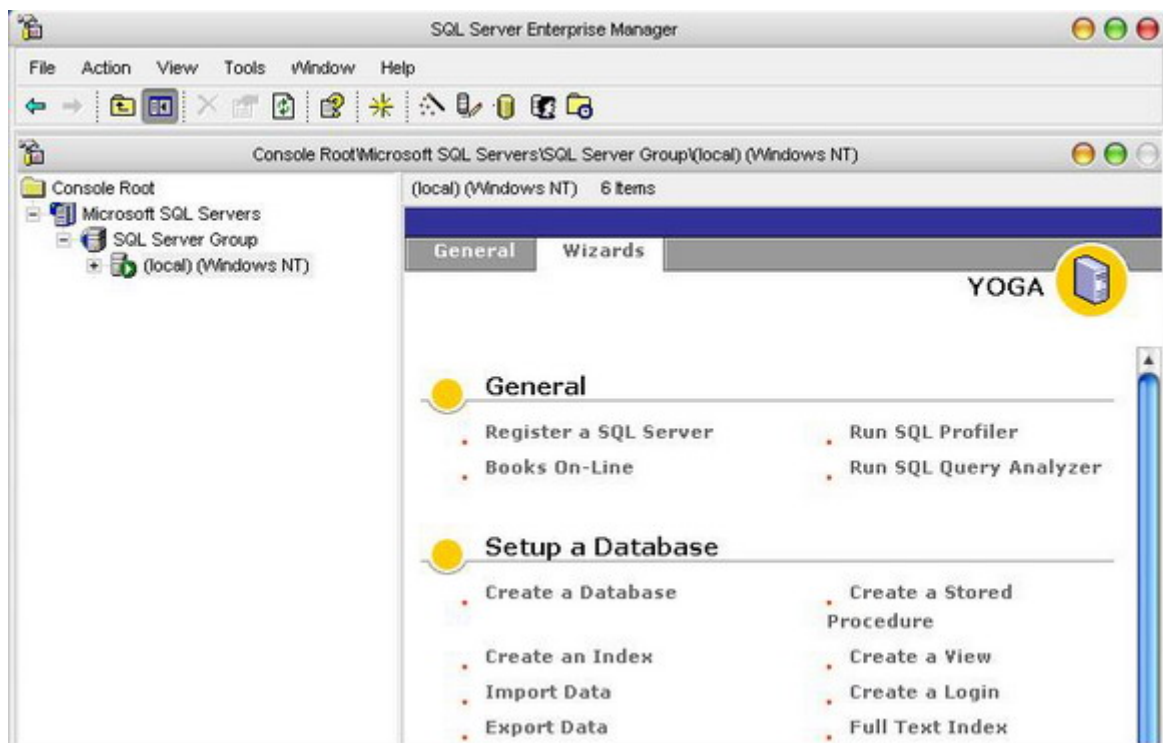


Gambar SQL Server Enterprise Manager

Klik tanda + pada simpul Microsoft SQL Servers, akan ditampilkan simpul SQL Server Group, klik tanda + lagi, maka akan ditampilkan nama server kita. Klik kanan pada simpul tersebut, kemudian pilih View ➔ TaskPad. Ada dua macam tab yang ditampilkan, yaitu *General* dan *Wizards*. Tab *General* menampilkan informasi umum dan tab *Wizards* akan menampilkan pilihan wizard yang tersedia dan kita dapat memilih dengan mengklik salah satu wizard untuk mengaktifkannya.



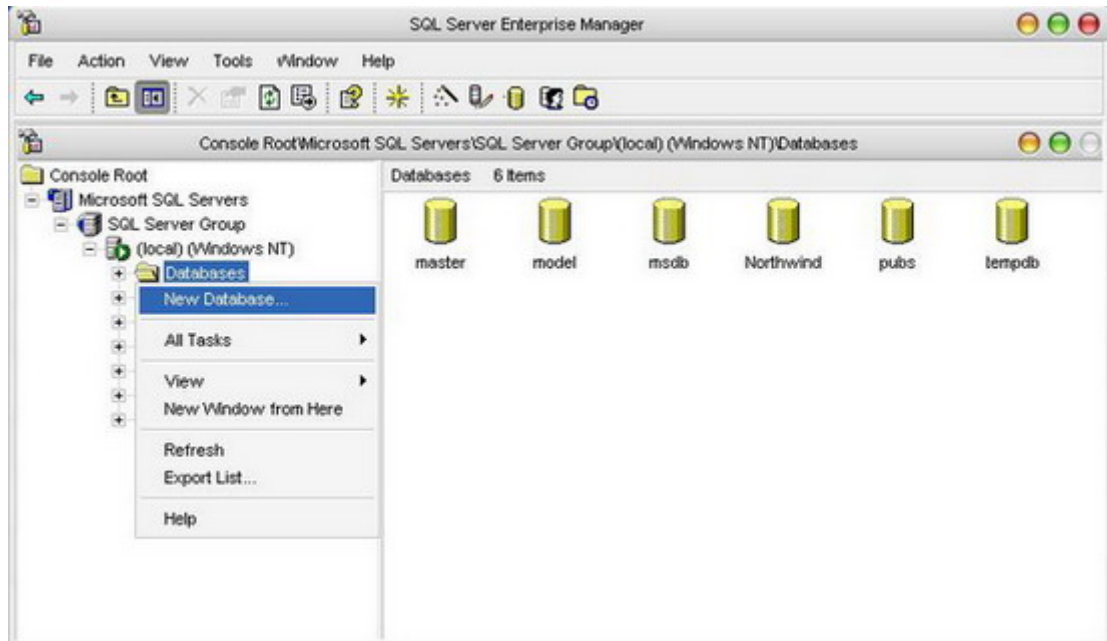
Gambar SQL Server Enterprise Manager Tab General (TaskPad)



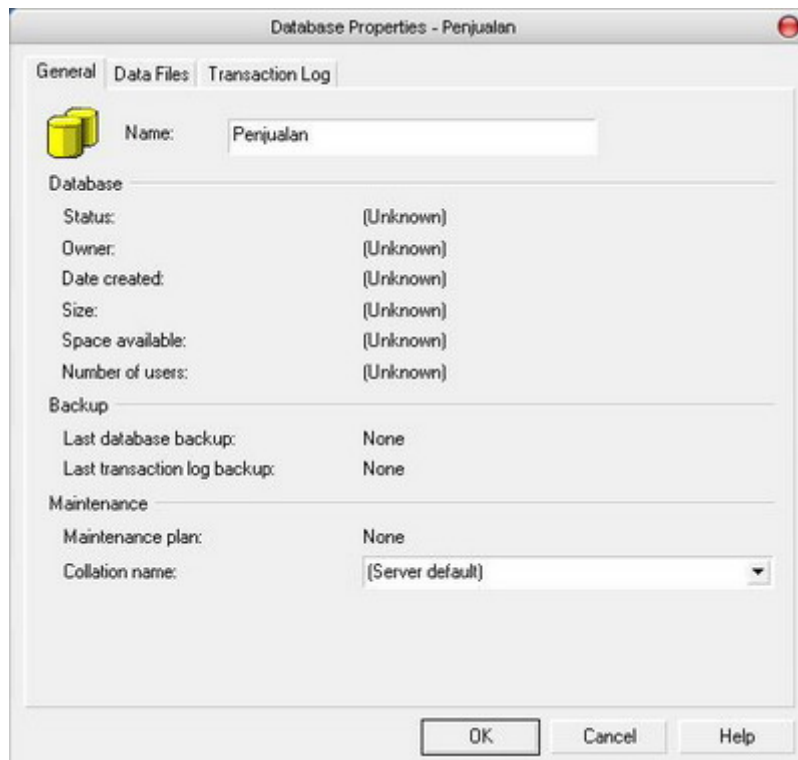
Gambar SQL Server Enterprise Manager Tab Wizards (TaskPad)

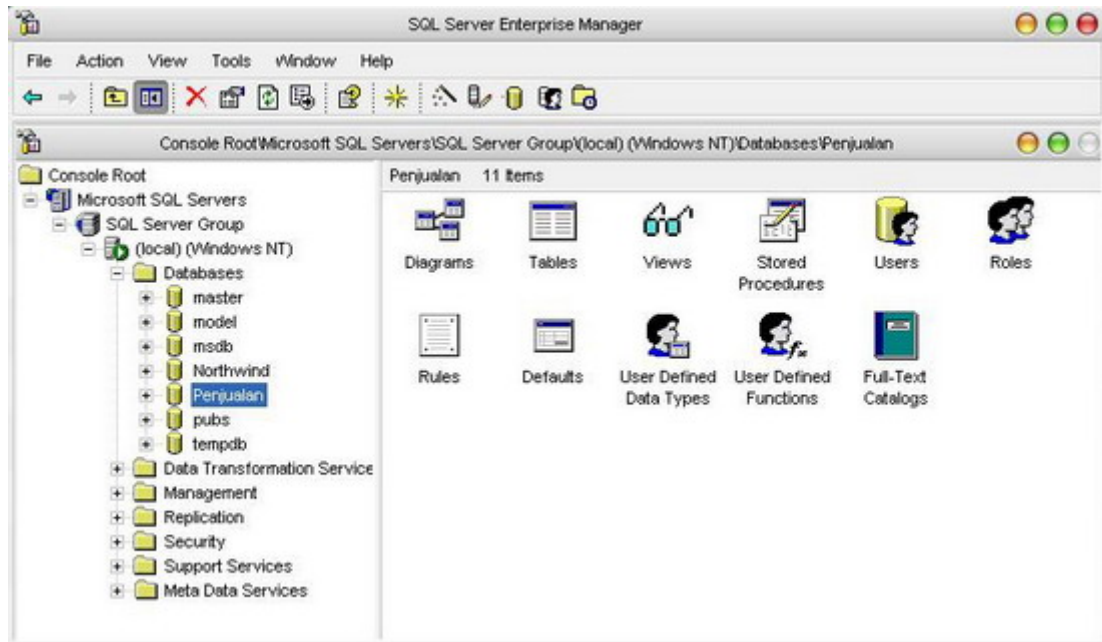
Membuat database menggunakan Wizard Enterprise Manager

- Buka tab Tree sampai kita menemukan server yang kita miliki pada program Enterprise Manager
- Klik kanan pada simpul Databases → Klik New Database, kemudian akan ditampilkan kotak dialog Database Properties. (lihat gambar)



- Ketikkan nama database yang ingin kita buat pada Tab General.
- Klik OK.

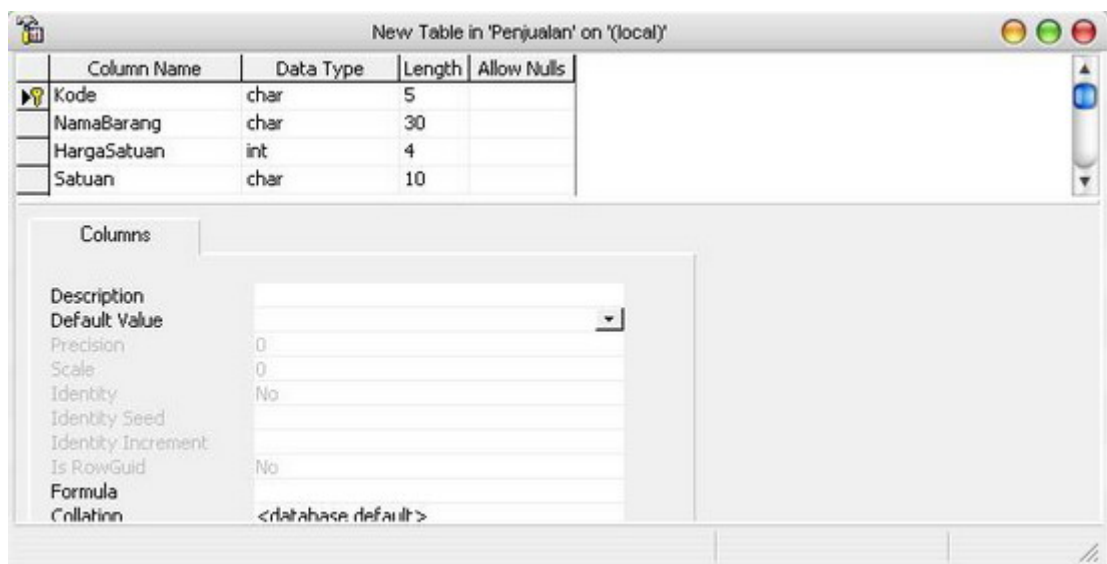




Gambar SQL Server Enterprise Database Penjualan

Membuat Tabel Menggunakan Wizard

- klik kanan pada database penjualan,
- Pilih New → Table, kemudian muncul kotak dialog new table, (lihat gambar)



- Isikan Nama Coloum Name, Data Type, Length dan Properties lainnya sesuai kebutuhan,
- Keluar dari kotak dialog new table,
- Save Tabel yang kita buat. (Lihat Gambar)



Mengentry Data Pada Tabel

- Klik kanan pada nama tabel (misal TabelBarang)
- Pilih Open → Return all rows



- Isikan data sesuai kebutuhan

Data in Table 'TabelBarang' in 'Penjualan' on '(local)'

Kode	NamaBarang	HargaSatuan	Satuan
B0001	Asus P5WD2	3500000	unit
B0002	Intel P4D 3.8 GHz EE	9000000	unit
B0003	Kingston 1 GB	2000000	keping
B0004	Western Digital SATA 500 GB	3000000	buah
B0005	DVD-RW Plextor SATA	2500000	buah
B0006	CD-R Verbatim	200000	box
B0007	Samsung SyncMaster 17" Flat	3500000	unit
B0008	Logitech Mouse & Keyboard	300000	unit
B0009	HIS Ati Radeon 9600	5000000	buah
B0010	Disket Sony	30000	box
*			

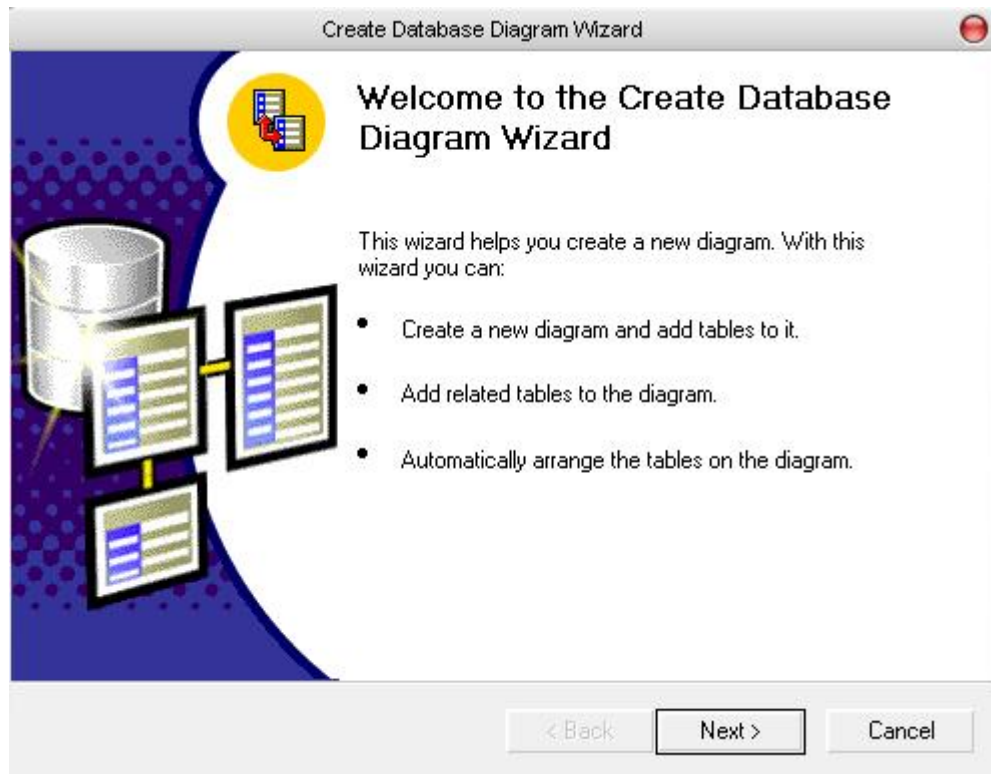
- Keluar dari dialog pengisian data.

Membuat Relasi Pada Tabel

- Pilih Diagram pada database kita (misal Penjualan)
- Klik Kanan pada Diagrams → New Database Diagram

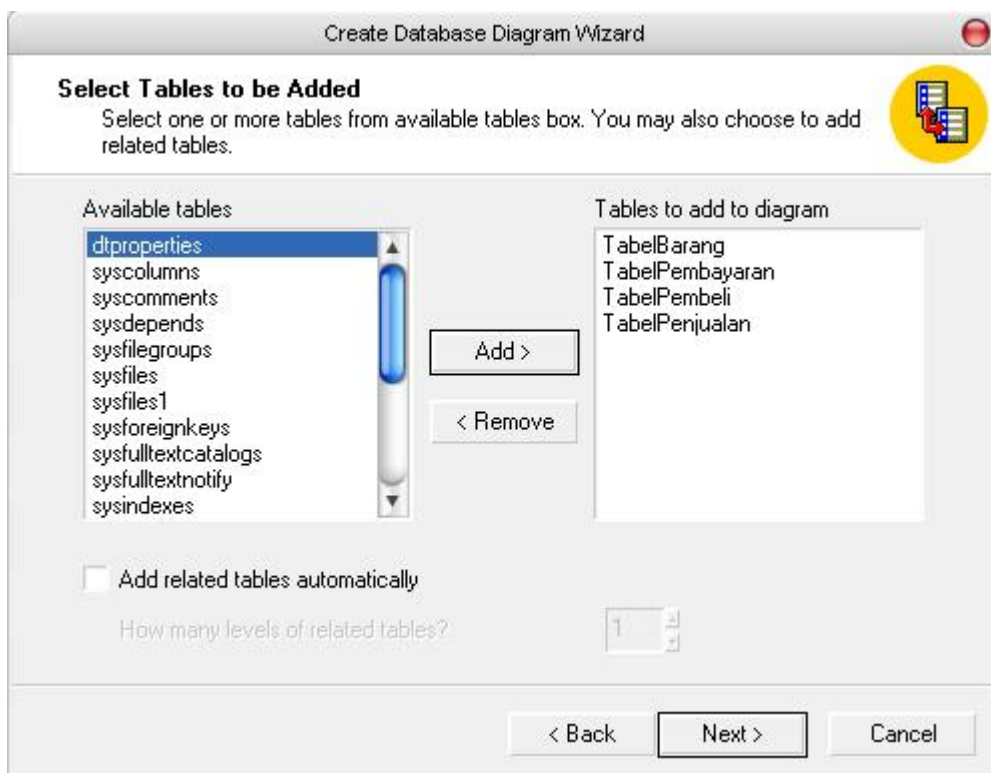


- c. Muncul dialog Create Database Diagram Wizard



- d. Pilih Next

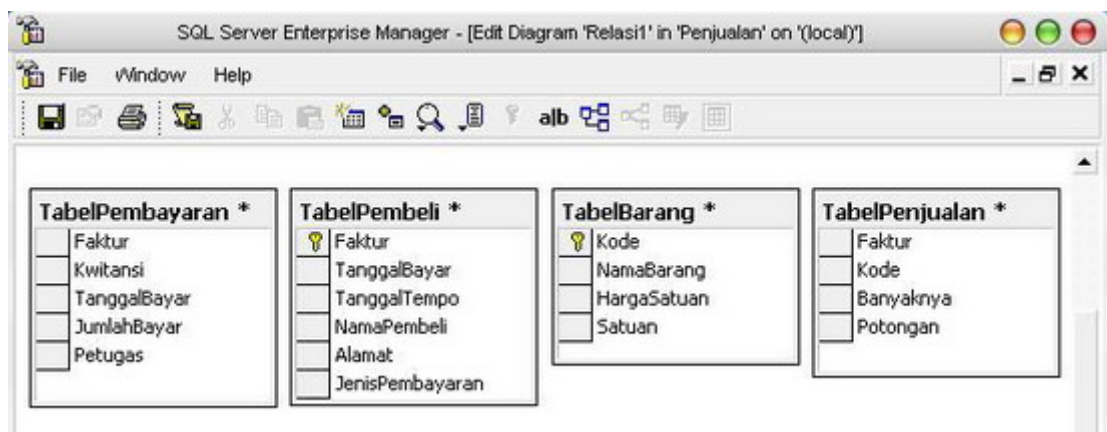
- e. Pilih Tabel sesuai kebutuhan → Klik Add → Klik Next



- f. Klik Finish.



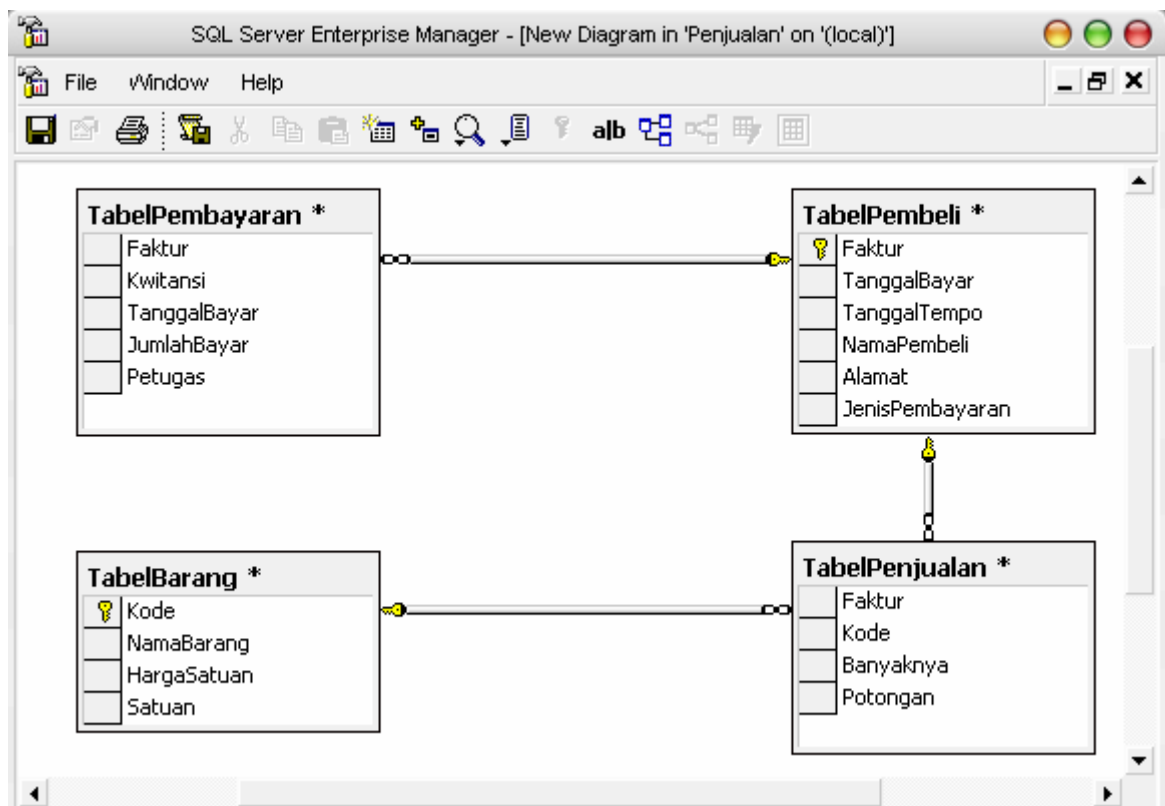
- g. Akan muncul tabel-tabel yang akan kita relasikan (lihat gambar)



- h. Drag pada Field yang berjenis Primary Key ke Field Tabel lain yang bersifat Foreign Key. Muncul dialog Create Relationship (lihat gambar)
- i. Aktifkan semua pilihan pada dialog yang tersedia.
- j. Berikan nama pada tiap relasi → Klik OK.
- k. Ulangi langkah H, I dan J hingga seluruh tabel berelasi satu sama lain.
- l. Berikan nama (save) database diagram yang telah dibuat (misal relasi1).



Gambar SQL Create Relationship



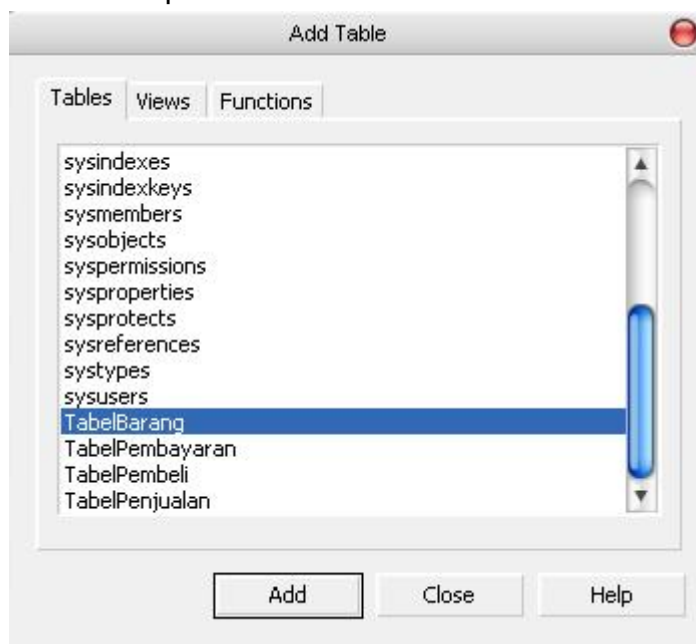
Gambar SQL Create Relationship

Membuat View

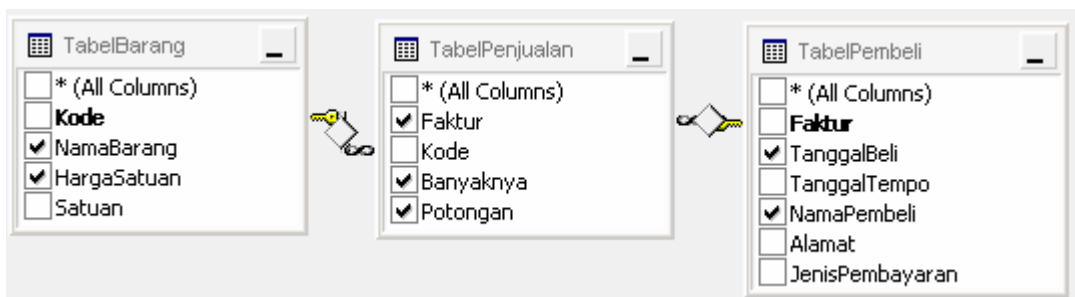
- Pilih database → Pilih View
- Pada View, klik kanan → New View



- Muncul dialog New View
- Klik kanan pada area New View → Pilih Add Table



- Masukan tabel yang diperlukan
- Pilih Field sesuai kebutuhan



- Berikan kriteria / kondisi jika diperlukan
- Jalankan View (RUN)
- Berikan nama (Save) pada view yang telah dibuat.

Column	Alias	Table	Output	Sort Type	Sort Order	Criteria	Or...
TanggalBeli		TabelPembeli	✓				
NamaPembeli		TabelPembeli	✓				
NamaBarang		TabelBarang	✓				
HargaSatuan		TabelBarang	✓				
Banyaknya		TabelPenjualan	✓				
Potongan		TabelPenjualan	✓				

```

SELECT  dbo.TabelPenjualan.Faktur, dbo.TabelPembeli.TanggalBeli, dbo.TabelPembeli>NamaPembeli, dbo.TabelBarang>NamaBarang,
        dbo.TabelBarang.HargaSatuan, dbo.TabelPenjualan.Banyaknya, dbo.TabelPenjualan.Potongan
FROM    dbo.TabelBarang INNER JOIN
        dbo.TabelPenjualan ON dbo.TabelBarang.Kode = dbo.TabelPenjualan.Kode INNER JOIN
        dbo.TabelPembeli ON dbo.TabelPenjualan.Faktur = dbo.TabelPembeli.Faktur

```

Gambar Field-Field Terpilih & Sintax SQLnya

	Faktur	TanggalBeli	NamaPembeli	NamaBarang	HargaSatuan	Banyaknya	Potongan
▶	F0001	9/22/2005	Yoga Prihastomo	Intel P4D 3.8 GHz EE	9000000	10	0.2
	F0001	9/22/2005	Yoga Prihastomo	DVD-RW Plector SATA	2500000	5	0.1
	F0001	9/22/2005	Yoga Prihastomo	Logitec Mouse & Keyboard	300000	10	0.1
	F0002	9/22/2005	Ahmad Yani	Asus P5WD2	3500000	5	0.2
	F0002	9/22/2005	Ahmad Yani	Kingston 1 GB	2000000	10	0.05
	F0003	9/22/2005	Suci Utari	Disket Sony	30000	20	0.1
	F0003	9/22/2005	Suci Utari	Logitec Mouse & Keyboard	300000	5	0.05
	F0004	9/23/2005	Retno Anjarwati	Samsung SyncMaster 17" Flat	3500000	8	0.1
	F0004	9/23/2005	Retno Anjarwati	Western Digital SATA 500 GB	3000000	10	0.25
	F0005	9/23/2005	Anisa Puspita	CD-R Verbatim	200000	15	0.1
	F0005	9/23/2005	Anisa Puspita	HIS Ati Radeon 9600	5000000	5	0.05
*							

Gambar Hasil Running View Yang Dibuat



Gambar Konfirmasi Save

Query Analyzer

Query Analyzer merupakan alat bantu grafis yang dapat digunakan untuk mendesain, mengetes, dan menjalankan perintah-perintah Transact SQL, stored procedure, batch, dan script secara interaktif. Kita dapat menjalankan Query Analyzer dengan cara :

- Memanggil dari Enterprise Manager
- Dari menu Start
- Dari command prompt dengan menjalankan utilitas *isqlw*

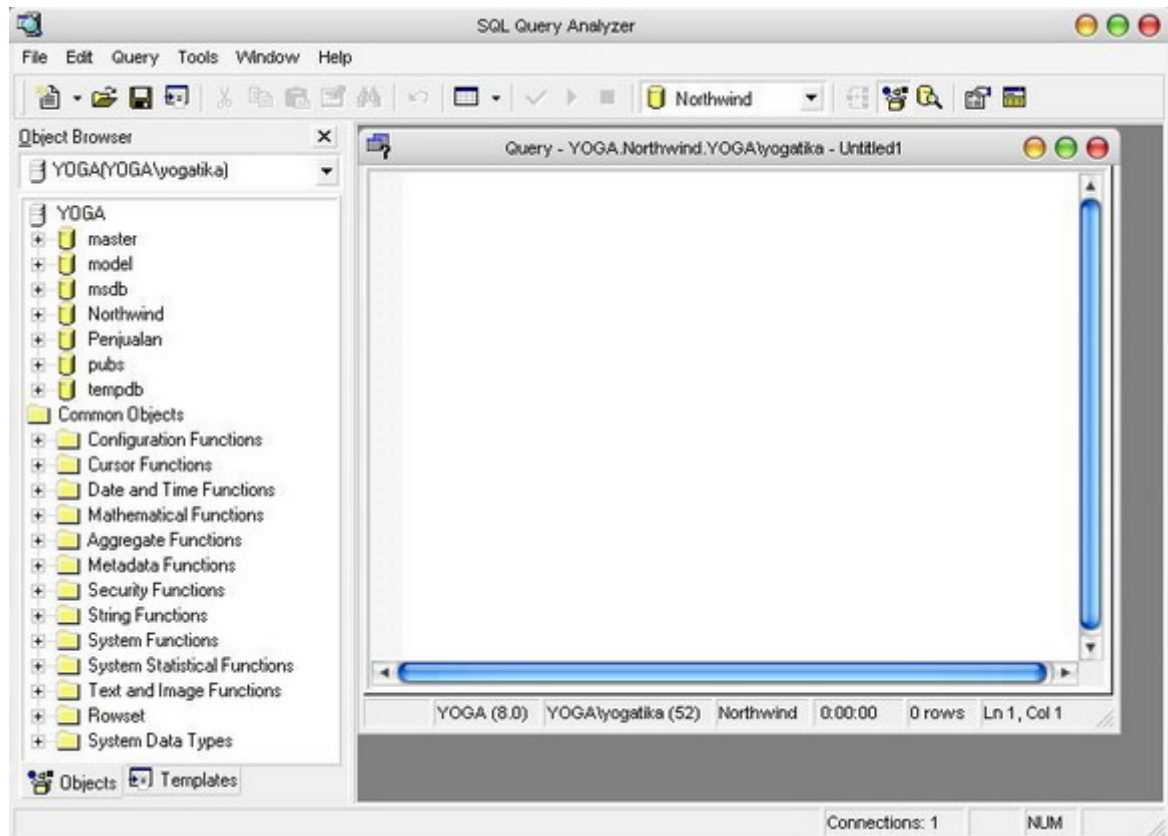
Menjalankan Query Analyzer

- a. Klik Start Menu → Programs → Microsoft SQL Server → Query Analyzer
- b. Pada daftar SQL Server, pilih Local atau Titik. Jika tidak ada, klik tombol ... untuk memilih komputer lokal yang kita gunakan.



Gambar Proses Authentication

- c. Pilih tombol radio Windows Authentication, artinya memakai autentikasi dari Windows.
- d. Klik OK. Query Analyzer akan ditampilkan dalam dua jendela yaitu Object Browser dan Query Windows.
- e. Jika jendela Object Browser tidak nampak, klik tombol Object Browser pada toolbar atau tekan F8, atau pilih menu Tool → Object Browser → Show/Hide

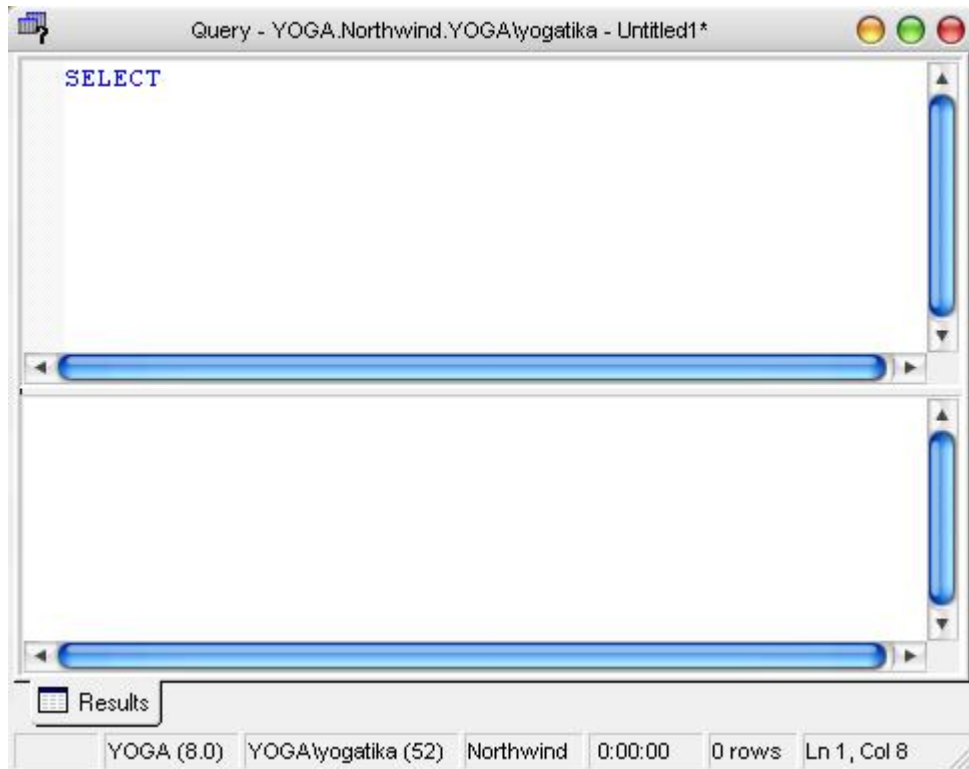


Gambar SQL Query Analyzer

Query Analyzer memberikan sejumlah jendela, kotak dialog, dan wizard yang dapat membantu kita dalam mengelola database serta data yang tersimpan di dalamnya. Bagian-bagian berikut ini akan membahas fasilitas-fasilitas yang ada di dalam Query Analyzer.

Query Windows

Query Windows dibagi menjadi dua bagian, yaitu jendela *Editor* dan jendela *Result*. Pada saat kita pertama kali memanggil Query Analyzer, hanya jendela Editor yang ditampilkan. Jendela Result akan nampak secara otomatis saat kita menjalankan sebuah perintah Transact SQL. Tetapi kita dapat menampilkan jendela Result secara manual dengan memilih tombol (icon) Show Results Pane pada toolbar atau menggunakan shortcut Ctrl + R. Kita dapat mengubah tampilan kedua jendela tersebut dengan mengakses kotak dialog Options dari menu Tool → Options



Gambar Jendela Editor dan Jendela Result

Jendela Editor (bagian atas) adalah jendela untuk memasukkan perintah SQL dan menjalankannya. Kita dapat memasukkan perintah dengan salah satu dari cara-cara berikut :

- Mengetikkan perintah SQL secara langsung pada jendela Editor
- Membuka sebuah script SQL yang telah disimpan. Isi script tersebut akan ditampilkan di jendela Editor dan kita dapat mengubahnya
- Membuka sebuah file template dan mengubahnya
- Memakai fasilitas scripting dari object browser untuk mengcopy perintah-perintah SQL dari object database yang dipilih.

Object Browser

Object Browser adalah alat bantu dalam bentuk pohon (tree) untuk melacak atau browse objek-objek dalam sebuah database. Selain fungsi navigasi, Object Browser memungkinkan pembuatan script, mengeksekusi stored procedure, dan mengakses table atau view.

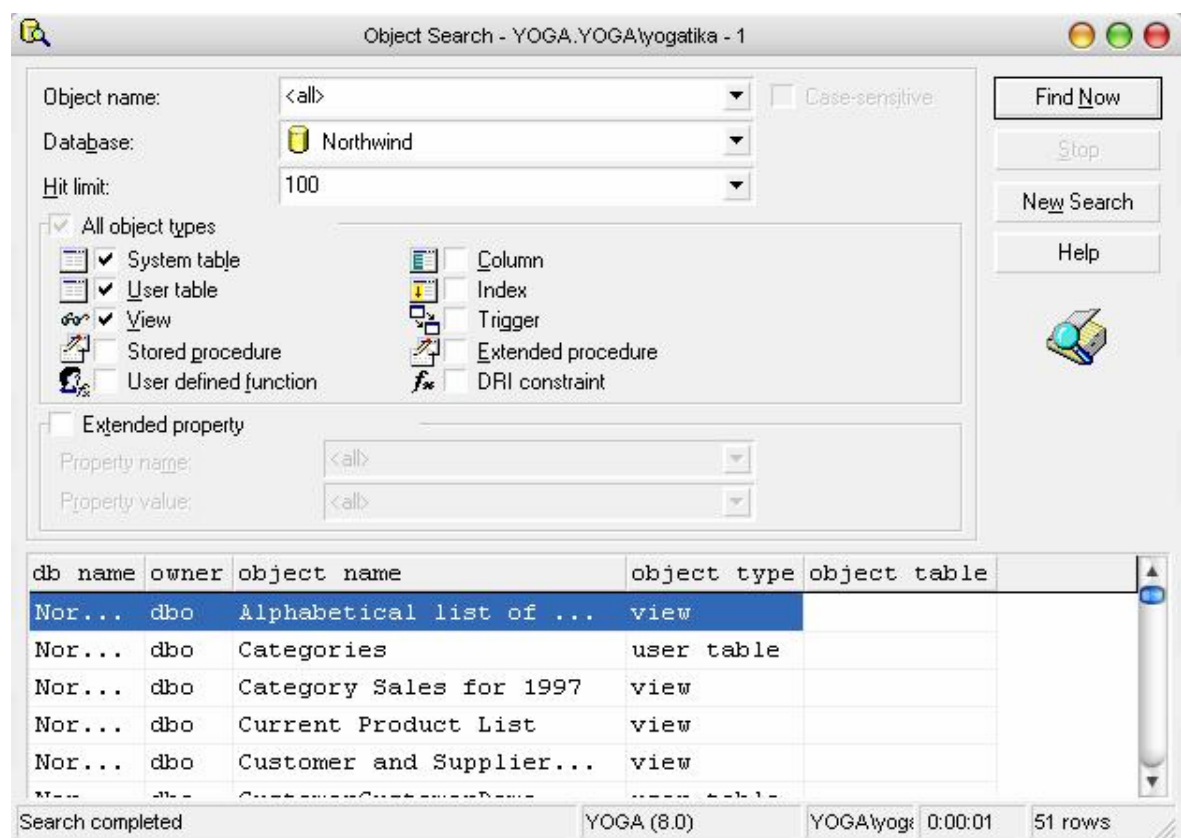
Object Browser mempunyai dua buah tab yaitu :

- ❏ Tab Objects yang menampilkan semua objek dalam sebuah database dan menampilkan objek-objek umum seperti function-function dan tipe data yang sudah diberikan oleh SQL Server.
- ❏ Tab Templates dipakai untuk mengakses folder Templates.

Bagian-bagian berikut akan membarikan beberapa hal yang dapat dijalankan pada Object Bowser :

Mencari Object Dalam Database

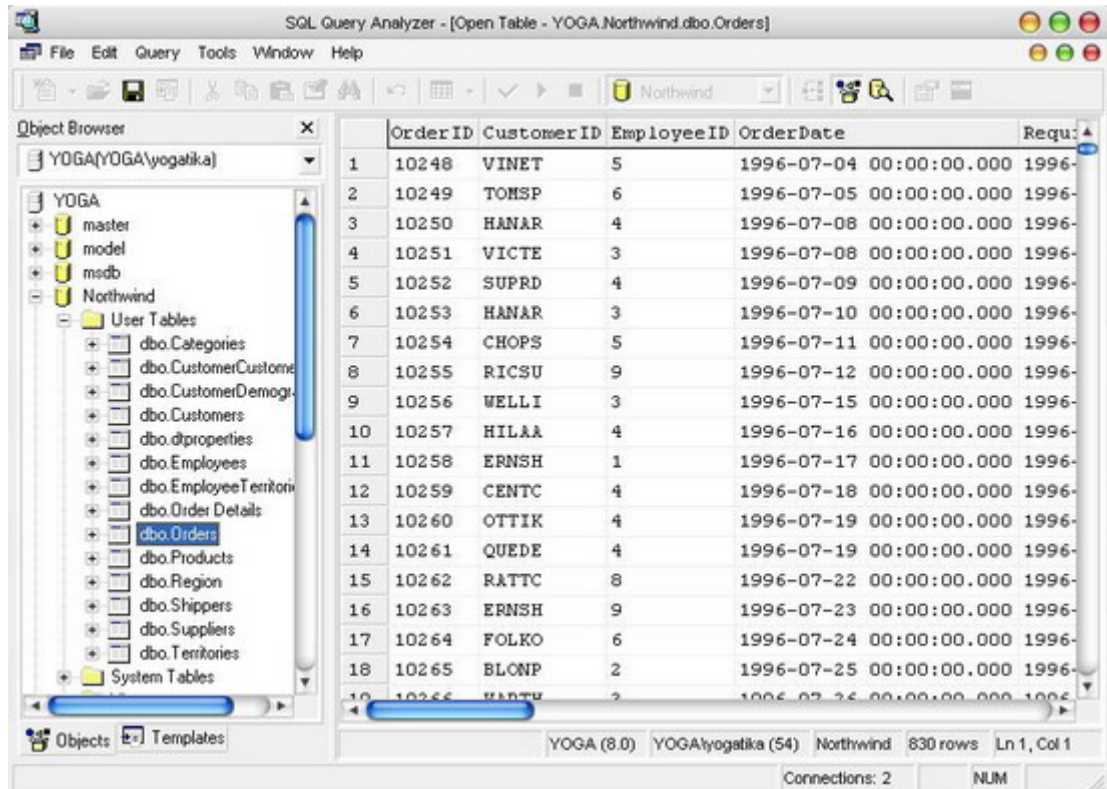
- a. Pada toolbar, tekan tombol Object Search atau tekan tombol F4, maka akan ditampilkan kotak dialog Object Search.
- b. Dari daftar Database, pilih Northwind.
- c. Pada bagian All Object Types ➔ pilih System Table, User Table dan View.
- d. Klik tombol Find Now. Objek-objek yang kita inginkan akan ditampilkan.



Gambar Kotak Dialog Object Search

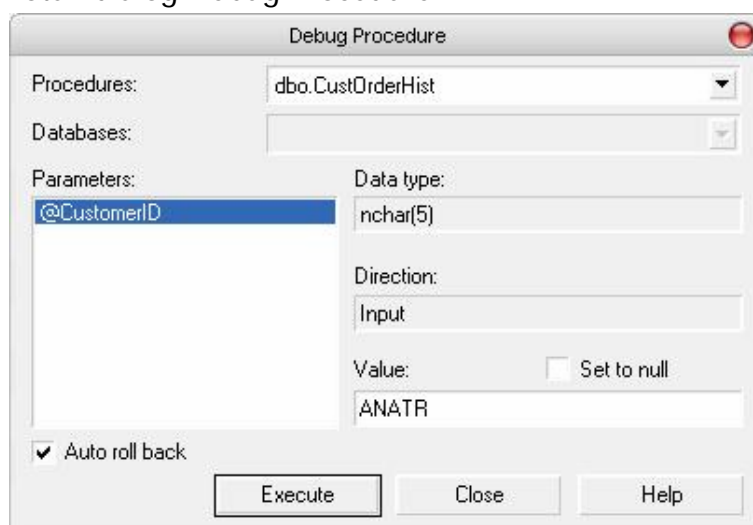
Melihat Isi Tabel

- Dari jendela Object Browser, pilih database Northwind → pilih User Table
- Klik kanan pada dbo.Orders → pilih Open, akan ditampilkan isi tabel.



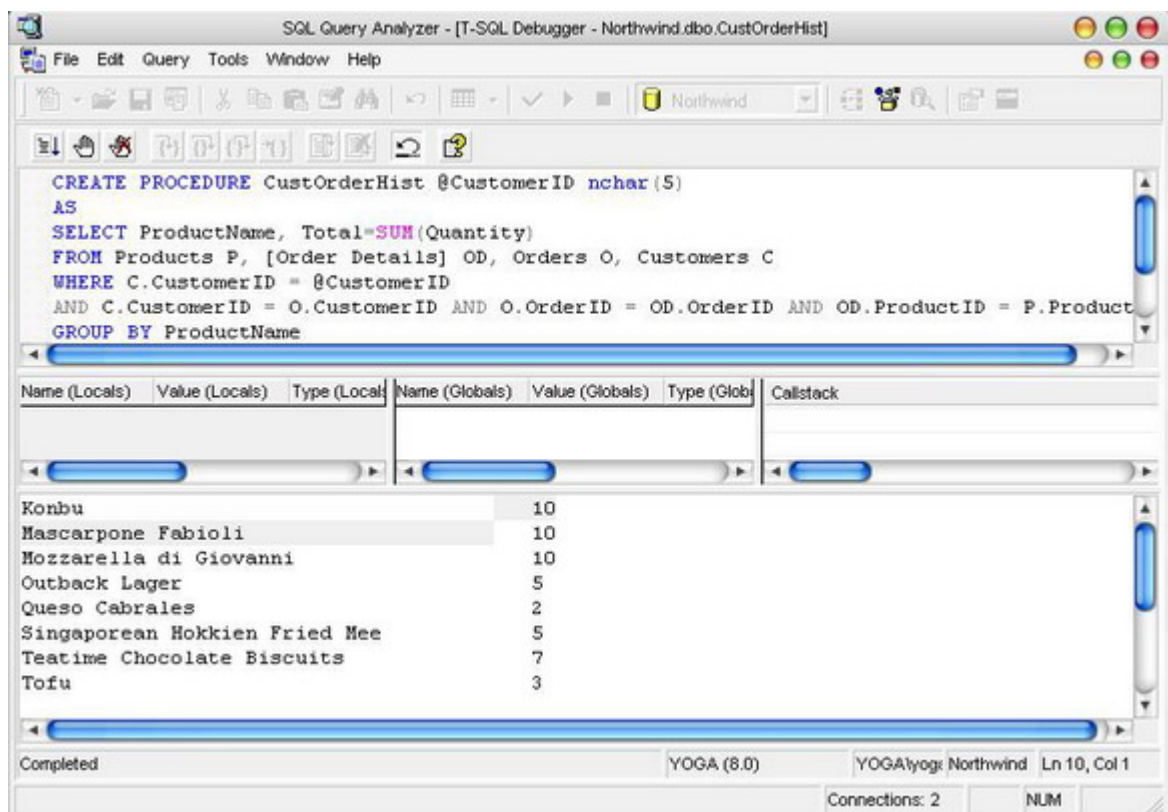
Menjalankan Stored Procedure

- Dari jendela Object Browser, buka Northwind ➔ pilih Stored Procedure.
- Klik kanan pada dbo.CustOrderHist lalu pilih Debug. Maka akan ditampilkan kotak dialog Debug Procedure.



- c. Procedure tersebut memerlukan sebuah parameter yaitu CustomerID. Pada field Value, isikan sebuah CustomerID (misal ANATR).
- d. Klik tombol Execute yang artinya proses debug dijalankan, maka akan ditampilkan jendela Transact-SQL Debugger.
- e. Untuk menjalankan Stored Procedure tekan tombol F5.

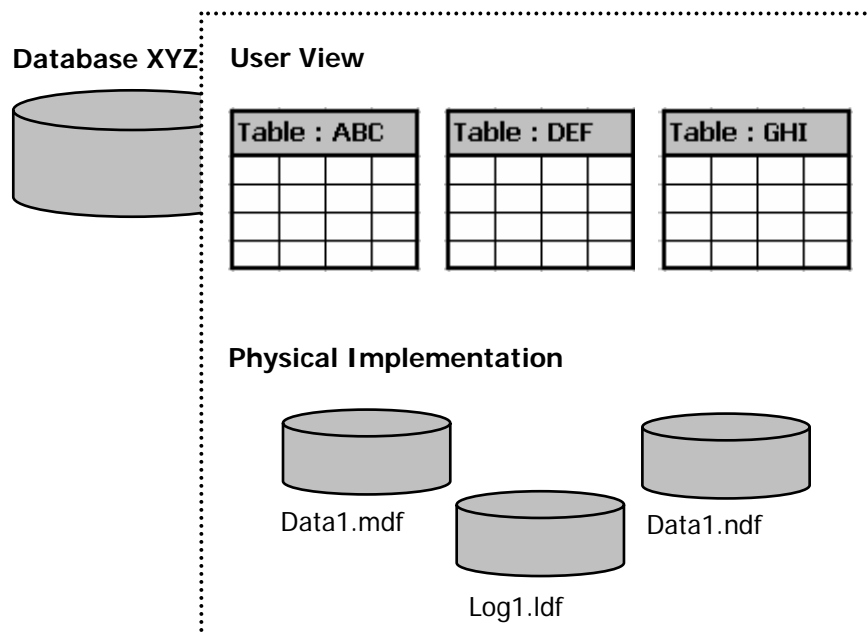
Stored Procedure tersebut menampilkan produk yang diorder oleh seorang customer serta kuantitasnya.



Gambar Menjalankan Stored Procedure

Sekilas Database

Data Microsoft SQL Server disimpan dalam beberapa database. Data dalam suatu database terorganisasi ke dalam komponen logis bagi user. Suatu database juga secara fisik terdiri dua atau lebih file pada disk. Bila menggunakan suatu database, pekerjaan terutama berlangsung pada komponen logis seperti table, view, file procedure, dan user. Sedangkan secara fisik, file database dapat terlihat pada disk, dan umumnya hanya database administrator yang perlu bekerja dengan komponen fisik tersebut.



Mendesain logika database mencakup:

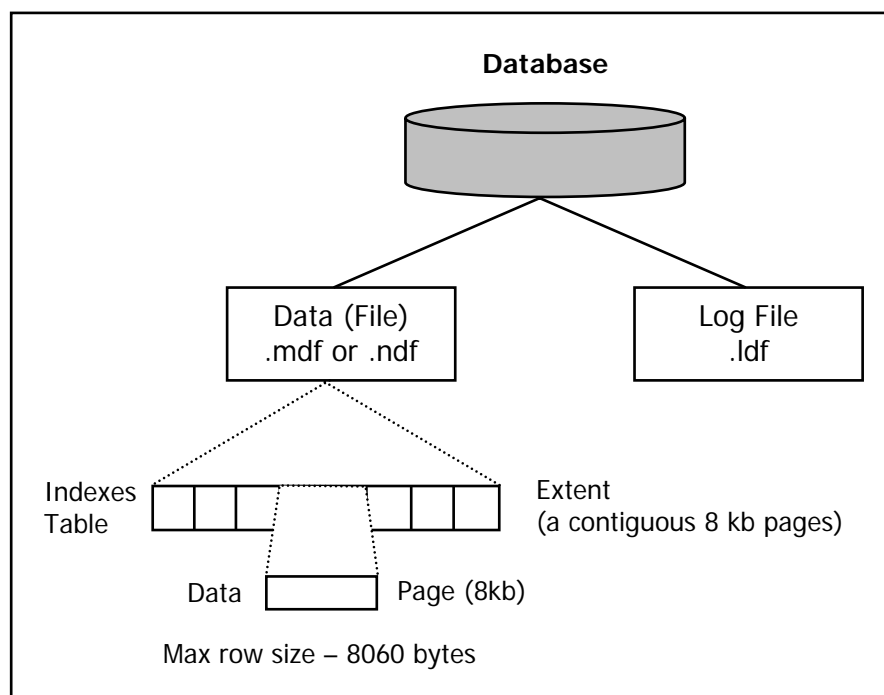
- ❏ Table dan table name - disebut juga sebagai entity
- ❏ Column name dalam setiap tabel - disebut sebagai attribute.
- ❏ Column characteristic - seperti nilai unik dan penyertaan nilai null.
- ❏ Primary key untuk setiap tabel — merupakan column yang menyimpan nilai unik untuk bisa mengidentifikasi setiap baris pada tabel. Walaupun terdapat column lain dengan nilai unik, hanya satu yang teridentifikasi sebagai kunci akses dalam melakukan akses data. Setiap tabel hanya dapat memiliki satu primary key.
- ❏ Relationship between table - baris di dalam beberapa tabel dapat tergantung pada satu atau lebih baris pada tabel lain. Ketergantungan antar tabel ini disebut sebagai relasi (relationship). Untuk mendefinisikan relasi, suatu

column atau beberapa column dalam suatu tabel harus diberikan key, disebut sebagai foreign key dan menjadi referensi bagi primary key dan tabel yang lain.

SQL-Server Database File

File Database dan SQL-Server dapat dikategorikan menjadi 3 jenis :

1. PRIMARY DATA FILE. Data disimpan dalam file dengan ekstensi “.mdf”. File ini merupakan database yang sesungguhnya, berisi tabel dan objek lain dan database. File ini disebut juga sebagai Primary Data File.
2. SECONDARY DATA FILE. File secondary memiliki ekstensi “.ndf”. File ini merupakan tempat penyimpanan data dan object yang tidak disimpan pada file primary. Database tidak harus memiliki file sekunder, tetapi dapat memiliki beberapa file sekunder.
3. LOG. File ini berisi catatan antara lain mengenai modifikasi tabel (UPDATE), input data baru (INSERT), dan penghapusan data (DELETE). Catatan tersebut akan dimanfaatkan oleh SQL-Server jika terjadi kegagalan sistem yang mengakibatkan crash. Dengan catatan yang ada pada file log, SQL-Server melakukan proses recovery, dan melakukan proses “rollback” untuk transaksi yang belum selesai. File LOG memiliki ekstensi “.ldf”. Secara default besar file log adalah 25% dan file data.



Saat kita menciptakan database dengan statement CREATE DATABASE, SQL-Server langsung membuat 2 file, yaitu PRIMARY DATAFILE dan LOG. Setiap data disimpan dalam 8 Kb block yang berdamping pada disk (SQL-Server 6.5 menempatkan 2Kb untuk setiap pages). Kumpulan block ini disebut sebagai pages. Ini berarti database dapat menyimpan 128 pages pada setiap megabyte hard disk. Sebuah baris (row) data tidak dapat melebihi besar pages. Maximum data pada satu baris adalah 8068 bytes. Selisih dari 8192 digunakan untuk header file. Sedangkan maximum space yang dapat digunakan dalam satu baris adalah 8094.

Tabel, object database lain, dan index disimpan pada Extent. Suatu extent adalah 8 pages yang berkesinambungan atau 64 Kb. Karena itu suatu database memiliki 16 extent pada setiap megabytes hard disk. Bila tabel berkembang sampai dengan 8 pages, maka tabel akan menggunakan extent sendiri (Unifor Extent).

Menggunakan Query Analyzer

Membuat Database

Sintax :

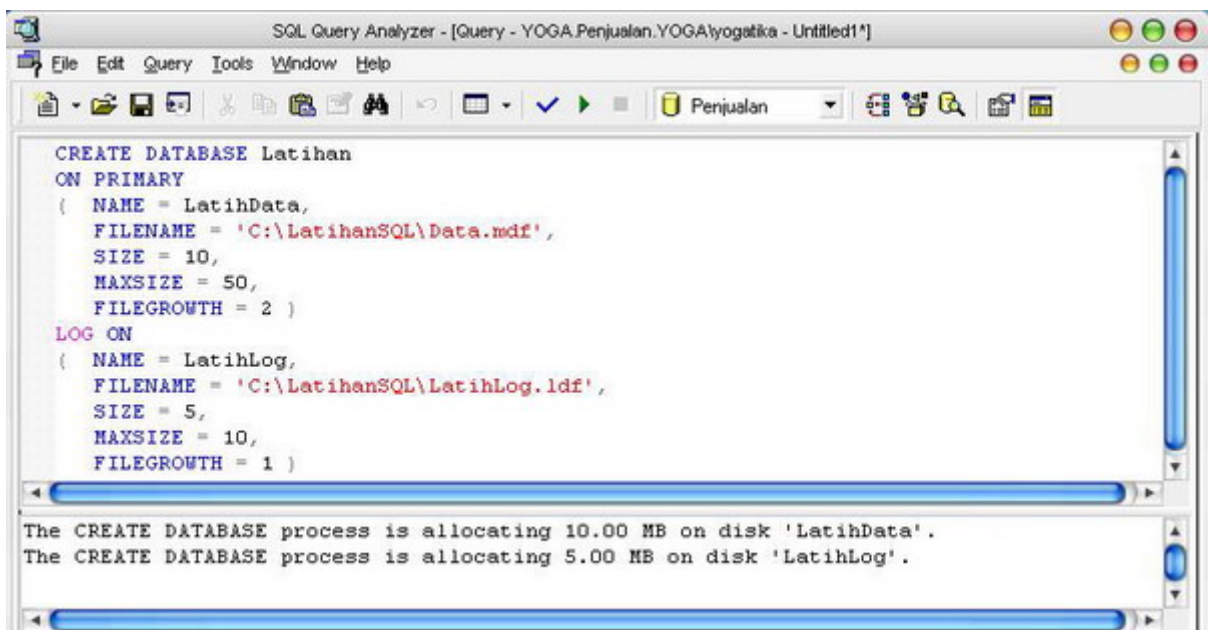
```
CREATE DATABASE database_name
[ ON
  [ <filespec> [,...n] ]
  [,<filegroup> [,...n] ]
]
[ LOG ON {<filespec> [,...n]} ]
[ COLLATE collation_name ]
[ FOR LOAD | FOR ATTACH ]
<filespec>::=
[ PRIMARY ]
( [ NAME = logical_file_name, ]
  FILENAME = 'os_file_name'
  [, SIZE = size ]
  [, MAXSIZE = { max_size | UNLIMITED } ]
  [, FILEGROWTH = growth_increment ] ) [,...n]
<filegroup>::=
FILEGROUP filegroup_name <filespec> [,...n]
```

Contoh Membuat Database "Latihan"

- Jalankan Query Analyzer
- Ketikkan Script Berikut (sesuaikan dengan lokasi penyimpanan kita)

```
ON PRIMARY
(
    NAME = LatihData,
    FILENAME = 'C:\LatihanSQL\Data.mdf',
    SIZE = 10,
    MAXSIZE = 50,
    FILEGROWTH = 2 )
LOG ON
(
    NAME = LatihLog,
    FILENAME = 'C:\LatihanSQL\LatihLog.ldf',
    SIZE = 5,
    MAXSIZE = 10,
    FILEGROWTH = 1 )
```

Hasil Eksekusi Pada Query Analyzer



- Simpan hasil eksekusi dari Query Analyzer

Membuat Table

Syntax :

```
CREATE TABLE table_name
(
    column_name_1 data_type ([size]),
    column_name_2 data_type ([size]),
    ...,
    column_name_n data_type ([size])
);
```

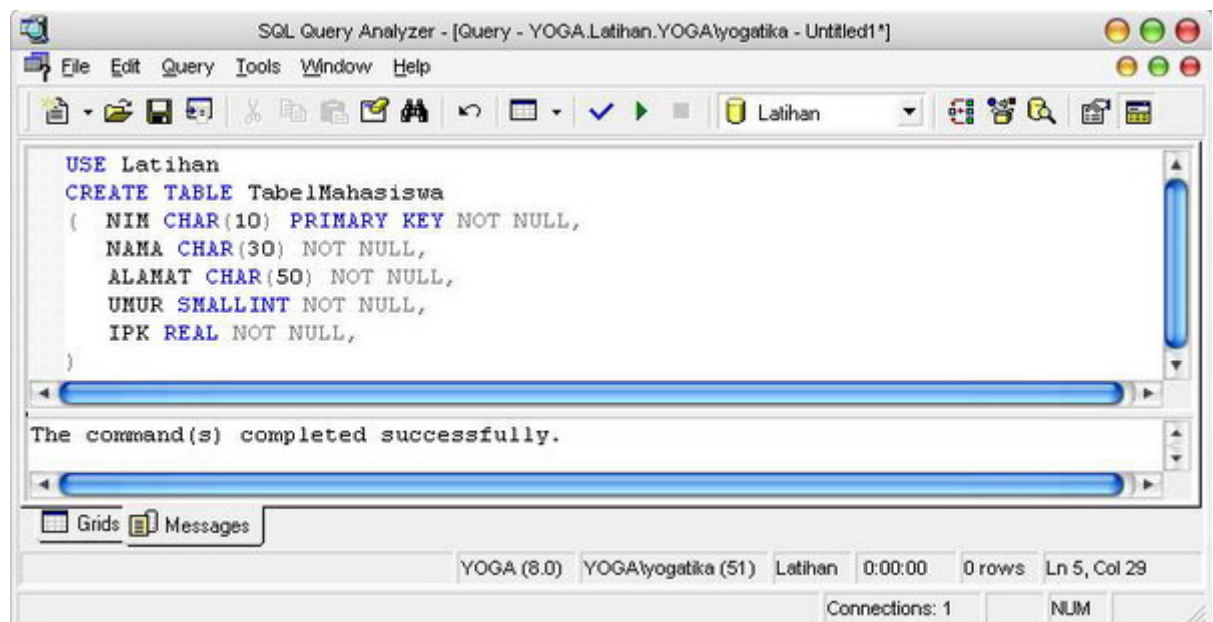
Contoh Membuat Tabel Mahasiswa :

- Jalankan Query Analyzer
- Ketikkan Script Berikut (sesuaikan dengan lokasi penyimpanan kita)

USE Latihan

```
CREATE TABLE TabelMahasiswa
( NIM CHAR(10) PRIMARY KEY NOT NULL,
  NAMA CHAR(30) NOT NULL,
  ALAMAT CHAR(50) NOT NULL,
  UMUR SMALLINT NOT NULL,
  IPK REAL NOT NULL,
)
```

Hasil Eksekusi Pada Query Analyzer



- Simpan hasil eksekusi dari Query Analyzer

Membuat View

Sintax :

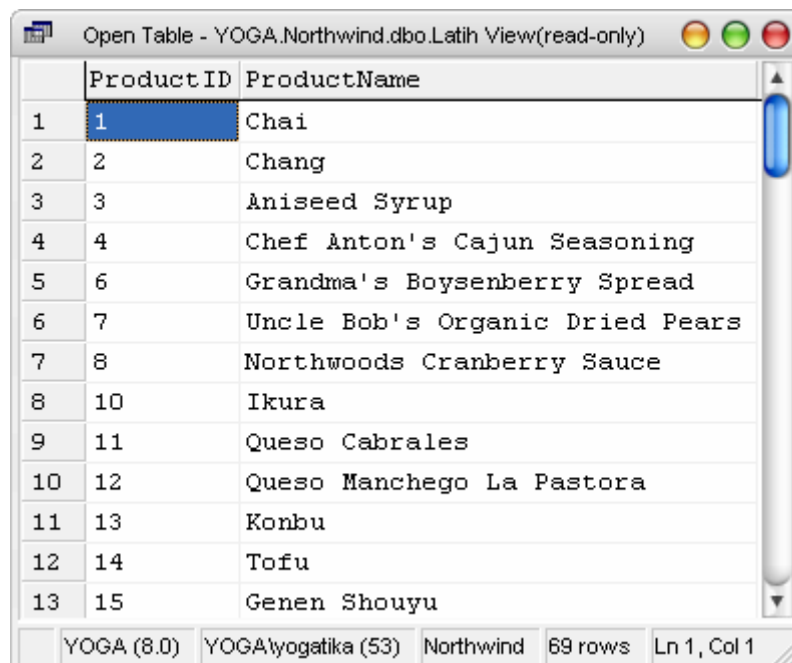
```
CREATE VIEW view_name
AS
    SELECT field_name
    FROM table_name
    [CONDITION]
```

Contoh Membuat View :

- Jalankan Query Analyzer
- Gunakan Database Northwind
- Ketikkan script berikut

```
CREATE VIEW "Latih View" AS
SELECT Product_List.ProductID, Product_List.ProductName
FROM Products AS Product_List
WHERE ((Product_List.Discontinued)=0))
```

Hasil Eksekusi Pada Query Analyzer



	ProductID	ProductName
1	1	Chai
2	2	Chang
3	3	Aniseed Syrup
4	4	Chef Anton's Cajun Seasoning
5	6	Grandma's Boysenberry Spread
6	7	Uncle Bob's Organic Dried Pears
7	8	Northwoods Cranberry Sauce
8	10	Ikura
9	11	Queso Cabrales
10	12	Queso Manchego La Pastora
11	13	Konbu
12	14	Tofu
13	15	Genen Shouyu

- Simpan hasil eksekusi dari Query Analyzer

Additional Review

Programming at Multiple Levels

In the early days of database management systems, a single computer handled all processing—what we now call the one-tier model. The advent of client/server computing introduced the classic two-tier model: a server that receives requests from and provides data to a separate client. The newest approach, the three-tier model, places an intermediate layer between the database server and the client application.

The three-tier model has gained supporters who view the model as preferable—for some applications—to the more traditional two-tier client/server model. Figures 9-1 and 9-2 show both models.

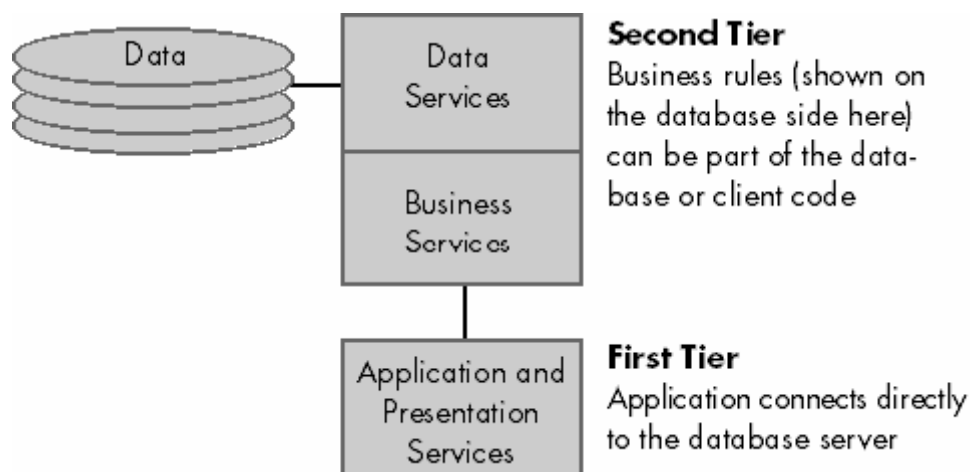


Figure The two-tier model.

The two-tier model has a client (first tier) and a server (second tier); the client handles application and presentation logic, and the server handles data services and business services. The two-tier model uses the so-called fat client—the client does a large amount of application processing locally. Many solutions deployed use this topology; certainly the lion's share of today's non-Internet client/server solutions are two-tier.

In the three-tier model, a thin client handles mostly presentation tasks. Supporters of the three-tier approach point out that this model allows the client computer to be less powerful; allows a variety of different client operating environments (obviously the case with Internet browsers); and reduces the complexity of installing, configuring, and maintaining software at the client. The client connects to an application server that handles the application process. The application server then connects to the database server, handling data services. The application server is typically a more powerful machine than the client, and it has all the issues of application code maintenance, configuration, operations, and so on.

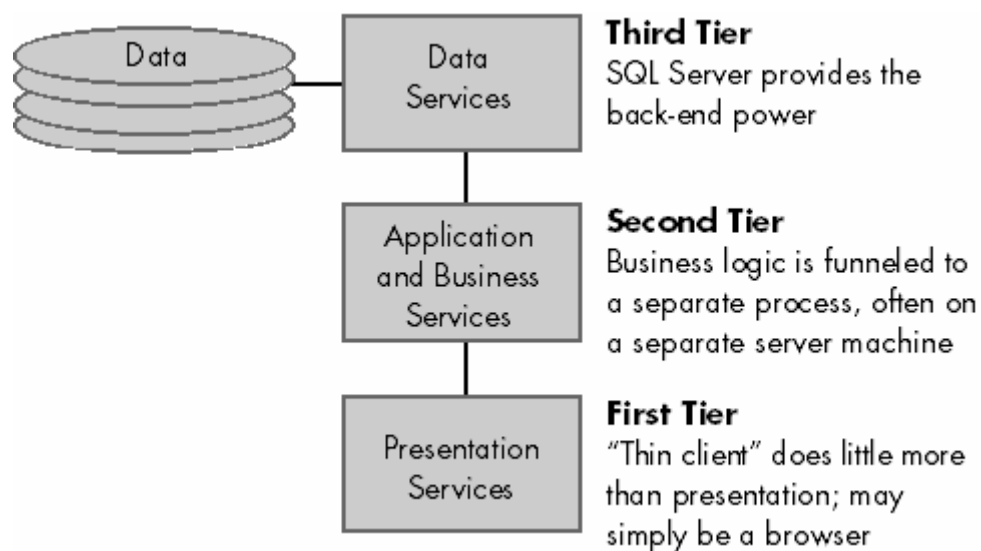
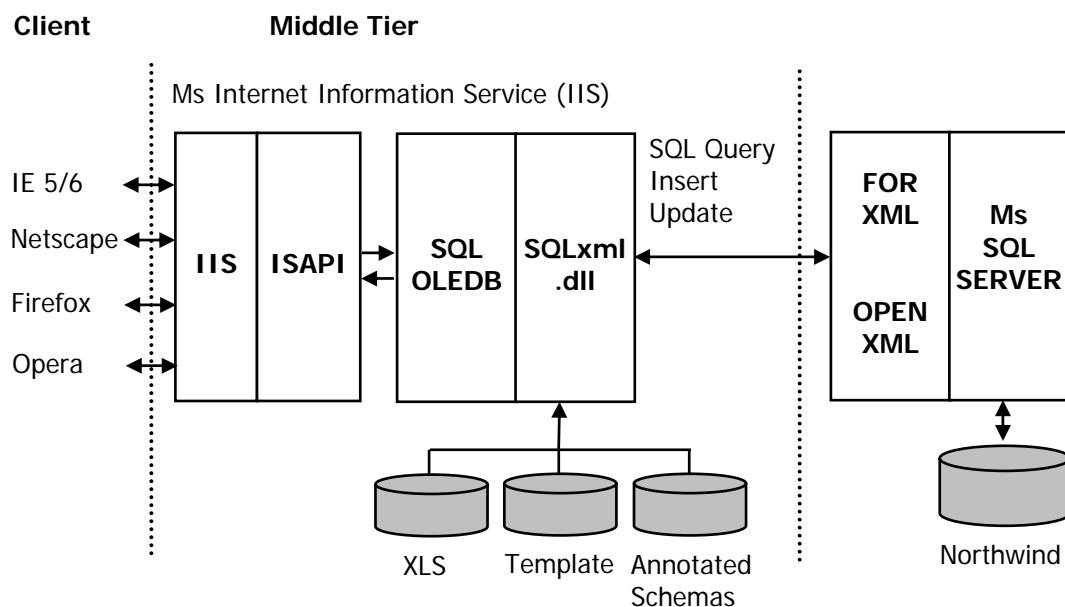


Figure The three-tier model.

Presumably, far fewer application servers exist than clients; therefore, server costs and issues should be reduced in the three-tier model. (The three tiers are logical concepts; the actual number of computers involved might be more or less than three.) Typically, you're part of a three-tier model when you use the Internet with a browser and access a Web page residing on an Internet server that performs database work. An example of a three-tier application is a mail-order company's Web page that checks current availability by using ODBC calls to access its inventory in SQL Server.

Both two-tier and three-tier solutions have advantages and disadvantages, and each can be viewed as a potential solution to a given problem. The specifics of the problem and the constraints imposed by the environment (including costs, hardware, and expertise of both the development staff and the user community) should drive the solution's overall design. However, some people claim that a three-tier solution means that such capabilities as stored procedures, rules, constraints, or triggers are no longer important or shouldn't be used. In a three-tier solution, the middle tier doing application logic is still a client from the perspective of the database services (that is, SQL Server). Consequently, for that tier, it's still preferable that remote work be done close to the data. If the data can be accessed by servers other than the application server, integrity checking must be done at the database server as well as in the application. (Otherwise, you've left a gaping hole through which the integrity checks in the application can be circumvented.)



Gambar Three-Tier Architecture

The middle tier is the Microsoft® Internet Information Services (IIS) server on which you must first create a virtual root using the IIS Virtual Directory Management for SQL Server utility. The IIS server name specified in the URL identifies the IIS server. The IIS server examines the virtual root specified in the URL and determines whether an ISAPI DLL extension (Sqlisapi.dll) has been registered for the virtual root that is specified in the URL. The IIS server loads the DLL and passes on the URL request to the DLL. The Sqlisapi.dll extension communicates with the OLE DB Provider for SQL Server (SQLOLEDB) and establishes connection with the instance of Microsoft SQL Server™ identified in the virtual root.

The entire XML functionality is implemented in Sqlxmlx.dll. When SQLOLEDB determines that the command is an XML command, the provider passes that command to Sqlxmlx.dll, which executes the command and returns the result to SQLOLEDB.

The template files, XML-Data Reduced (XDR) schema files, and Extensible Stylesheet Language (XSL) files reside on the IIS server. The XPath queries and the XDR schemas are handled on the IIS server. The XPath queries are translated into SQL commands by Sqlxmlx.dll.

The FOR XML clause and OPENXML are implemented on the server running SQL Server

Daftar Pustaka

Hartama, Dedy. 2001. **Microsoft SQL Server**. Jakarta : Penerbit Fajar PL3I.

Kadir, Abdul. 2002. **Penuntun Praktis Belajar SQL**. Yogyakarta : Penerbit Andi Yogyakarta.

Martina, Inge. 2004. **36 Jam Belajar Komputer Microsoft SQL Server 2000**. Jakarta : PT. Elex Media Komputindo.

Online Books :

SQL Server 2000 Books Online

Inside Microsoft SQL Server 7 Books Online

Website :

<http://www.sqlteam.com/>