BAB II

LANDASAN TEORI

2.1. Sejarah Perusahaan

PD Jembar merupakan perusahaan swasta yang didirikan pada tahun 1979 di Jalan Babakan Ciparay No.170 Bandung. PD Jembar ini adalah semacam toko yang menjual beras baik untuk eceran ataupun dalam jumlah besar. Karena toko ini semakin berkembang, PD Jembar kemudian membuka cabang di Caringin pada tahun 1981, kemudian pada taun 2000 di Gempol, pada taun buka cabang lagi di Cimindi dan terakhir tahun 2009 membuka cabang di Cibaduyut. Semua cabang berada di daerah sekitar Bandung dan Cimahi. Sedangkan untuk penelitian dilakukan di Jl. Raya Cimindi no 236 Cimahi.

PD Jembar mendapat pasokan dari daerah Ciawi, Cianjur, Purwokerto, Purworejo, Sumedang, Jatiwangi, Cirebon, Semarang, Kudus, Sragen, Ngawi, Majenang, Banjar, dan Ciamis. Beras–beras tersebut dikirim dengan menggunakan truk/puso/tronton. Kapasitas seperti tronton kapasitasnya yaitu 24 ton, puso dapat memuat 17 ton , colt diesel 8 ton, dan pick up 1.5 ton. Dalam sehari PD Jembar dapat memasok beras sekitar 20–30 ton, tergantung situasi dan stok yang ada di gudang.

2.2. Model

Model adalah suatu penyederhanaan dari suatu realitas yang kompleks. Model dikatakan lengkap apabila dapat mewakili berbagai aspek dari realitas yang

sedang dikaji. Sebagai contoh, boneka adalah model dari bentuk manusia. Boneka yang dapat tertawa, menangis, dan berjalan adalah model manusia yang lebih lengkap, tidak hanya mewakili bentuk tetapi juga beberapa perilaku manusia. [3]

2.2.1. Permodelan Sistem

Permodelan merupakan kumpulan aktivitas pembuatan model. Sebagai landasan pengertian permodelan diperlukan suatu penelahan tentang model itu sendiri secara spesifik ditinjau dari pendekatan sistem. Sebelum sampai pada tahap permodelan, perlu diketahui lebih dahulu jenis dan klasifikasi model-model secara terperinci. Salah satu dasar utama untuk mengembangkan model adalah guna menemukan peubah-peubah apa yang penting dan tepat. Penemuan peubah-peubah tersebut sangat erat hubungannya dengan pengkajian hubungan-hubungan yang terdapat diantara peubah-peubah. Teknik kuantitatif seperti persamaan regresi dan simulasi digunakan untuk mempelajari keterkaitan antar peubah dalam sebuah model.

2.2.2. Jenis Model

Klasifikasi perbedaan dari model memberikan pertambahan pendalaman pada tingkat kepentingannya, karena dapat dijelaskan dalam banyak cara. Model dapat dikategorikan menurut jenis, dimensi, fungsi, tujuan pokok pengkajian atau derajat keabstrakannya. Kategori umum adalah jenis model yang pada dasarnya dapat dikelompokan menjadi ikonik, analog dan simbolik.

2.2.2.1. Model Ikonik

Model ikonik adalah perwakilan fisik dari beberapa hal baik dalam bentuk ideal ataupun dalam skala yang berbeda. Model ikonik mempunyai karakteristik yang sama dengan hal yang diwakili dan terutama amat sesuai untuk menerangkan kejadian pada waktu yang spesifik.

Model ikonik dapat berdimensi dua (foto, peta, cetak biru) atau tiga dimensi (*prototip*e mesin, alat). Apabila model berdimensi lebih dari tiga dimensi maka tidak mungkin lagi dikonstruksi secara fisik sehingga diperlukan kategori model simbolik.

2.2.2.2. Model Analog (Model Diagramatik)

Model analog dapat mewakili situasi dinamik, yaitu keadaan berubah menurut waktu. Model ini lebih sering dipakai daripada model ikonik karena kemampuannya untuk mengetengahkan karakteristik dari kejadian yang dikaji. Model analog banyak berkesesuaian dengan penjabaran hubungan kuantitatif antara sifat dan kelas-kelas yang berbeda. Dengan melalui transformasi sifat menjadi analognya, maka kemampuan membuat perubahan dapat ditingkatkan. Contoh model analog ini adalah kurva permintaan, kurva distribusi frekuensi pada statistik, dan diagram alir.

2.2.2.3. Model Simbolik (Model Matematik)

Pada hakekatnya, ilmu sistem memusatkan perhatian kepada model simbolik sebagai perwakilan dari realitas yang sedang dikaji. Format model

simbolik dapat berupa bentuk angka, simbol, dan rumus. Jenis model simbolik yang umum dipakai adalah suatu persamaan (*equation*). Bentuk persamaan adalah tepat, singkat, dan mudah dimengerti. Simbol persamaan tidak saja mudah dimanipulasi daripada kata-kata, namun juga lebih cepat ditangkap maksudnya. Suatu persamaan adalah bahasa *universal* pada penelitian operasional dan ilmu sistem, dimana dipakai suatu logika simbolis. Permodelan mencakup suatu pemilihan dari karakteristik dari perwakilan abstrak yang paling tepat pada situasi yang terjadi. Pada umumnya, model matematis dapat diklasifikasikan menjadi dua bagian, yaitu [3]:

1. Model statik

   Model statik memberikan informasi tentang peubah-peubah model hanya pada titik tunggal dari waktu.

2. Model dinamik

   Model dinamik mampu menelusuri jalur waktu dari perubahan model. Model dinamik lebih sulit dan mahal pembuatannya, namun memberikan kekuatan yang lebih tinggi pada analisis dunia nyata.

Pemilihan model tergantung pada tujuan dari pengkajian sistem dan terlihat jelas pada formulasi permasalahan pada tahap evaluasi kelayakan. Sifat model juga tergantung pada teknik permodelan yang dipakai. Model yang mendasarkan pada teknik peluang dan memperhitungkan ketidaktentuan (*uncertainty*) disebut model probabilistik atau model stokastik. Dalam mengkaji suatu sistem, model ini sering dipakai karena perihal yang dikaji umumnya mengandung keputusan yang

tidak tentu. Kebalikan dari model ini adalah model kuantitatif yang tidak mempertimbangkan peluang kejadian, dikenal sebagai model deterministik.

2.3. Peramalan (*Forecasting*)

Peramalan merupakan bagian awal dari suatu proses pengambilan suatu keputusan. Sebelum melakukan peramalan harus diketahui terlebih dahulu apa sebenarnya persoalan dalam pengambilan keputusan itu. Peramalan adalah pemikiran terhadap suatu besaran, misalnya permintaan terhadap satu atau beberapa produk pada periode yang akan datang. Pada hakekatnya peramalan hanya merupakan suatu perkiraan(*guess*) tetapi dengan menggunakan teknik-teknik tertentu, maka peramalan menjadi lebih sekedar perkiraan. Peramalan dapat dikatakan perkiraan yang ilmiah (*educated guess*). Setiap pengambilan keputusan yang menyangkut keadaan di masa yang akan datang, maka pasti ada peramalan yang melandasi pengambilan keputusan tersebut.[2]

Dalam kegiatan produksi, peramalan dilakukan untuk menentukan jumlah permintaan terhadap suatu produk dan merupakan langkah awal dari proses perencanaan dan pengendalian produksi. Dalam peramalan ditetapkan jenis produk apa yang diperlukan (*what*), jumlahnya (*how many*), dan kapan dibutuhkan (*when*). Tujuan peramalan dalam kegiatan produksi adalah untuk meredam ketidakpastian, sehingga diperoleh suatu perkiraan yang mendekati keadaan yang sebenarnya. Suatu perusahaan biasanya menggunakan prosedur tiga tahap untuk sampai pada peramalan penjualan, yaitu diawali dengan melakukan peramalan lingkungan, diikuti dengan peramalan penjualan industri, dan diakhiri dengan peramalan penjualan perusahaan.

2.3.1. Pendefenisian Tujuan Peramalan

Tujuan peramalan dilihat dengan waktu[2]:

a. Jangka pendek (*Short Term*)

Menentukan kuantitas dan waktu dari item dijadikan produksi. Biasanya bersifat harian ataupun mingguan dan ditentukan oleh *Low Management*.

b. Jangka Menengah (*Medium Term*)

Menentukan kuantitas dan waktu dari kapasitas produksi. Biasanya bersifat bulanan ataupun kuartal dan ditentukan oleh *Middle Management*.

c. Jangka Panjang (*Long Term*)

Merencanakan kuantitas dan waktu dari fasilitas produksi. Biasanya bersifat tahunan, 5 tahun, 10 tahun, ataupun 20 tahun dan ditentukan oleh *Top Management*.

2.3.2. Beberapa Sifat Hasil Peramalan.

Dalam membuat peramalan atau menerapkan suatu peramalan maka ada beberapa hal yang harus dipertimbangkan yaitu [2]:

1. Ramalan pasti mengandung kesalahan, artinya peramal hanya bisa mengurangi ketidakpastian yang akan terjadi, tetapi tidak dapat menghilangkan ketidakpastian tersebut.
2. Peramalan seharusnya memberikan informasi tentang beberapa ukuran kesalahan, artinya karena peramalan pasti mengandung

kesalahan, maka adalah penting bagi peramal untuk menginformasikan seberapa besar kesalahan yang mungkin terjadi.

3. Peramalan jangka pendek lebih akurat dibandingkan peramalan jangka panjang. Hal ini disebabkan karena pada peramalan jangka pendek, faktor-faktor yang mempengaruhi permintaan relatif masih konstan sedangkan masih panjang periode peramalan, maka semakin besar pula kemungkinan terjadinya perubahan faktor-faktor yang mempengaruhi permintaan.

2.3.3. Model Peramalan Dengan Menggunakan Analisis Deret Waktu

Deret waktu adalah serangkaian nilai-nilai variabel yang disusun berdasarkan waktu. Analisis ini mempelajari pola gerakan nilai interval tertentu (misalnya minggu, bulan, dan tahun) yang teratur sehingga memperoleh ukuran-ukuran yang dapat digunakan untuk membuat keputusan. Keungulan analisis ini dengan regresi adalah bahwa penyusunan analisis regresi didasarkan pada teori atau logika ekonomi, sementara analisis deret waktu dapat dikatakan tanpa landasan teori namun semua metode didasarkan pada asumsi bahwa pola lama akan terulang. Ada beberapa model yang digunakan dalam analisis deret waktu yaitu tren linier, kuadratis, dan eksponensial.[7]

2.3.3.1.Tren linier

Tren merupakan suatu garis atau kurva yang halus yang menunjukan suatu kecendrungan umum suatu variabel. Arah tren dapat terlihat naik ataupun turun.

Untuk melihat tren yang ada sebaiknya digunakan suatu periode sekurang-kurangnya meliputi 1 siklis.[8]

Siklis adalah pergerakan disekitar rata-rata nilai variabel deret waktu, di atas atau dibawah tren jangka panjang. Dari gerakan siklis diperoleh beberapa titik tertinggi (puncak) dan titik terendah (lembah). Pola berulang ini berlangsung dalam jangka waktu tertentu. Pergerakan dari puncak ke lembah dinamakan kontraksi dan pergerakan dari lembah ke puncak dinamakan ekspansi.

Kadang-kadang dalam suatu deret waktu terjadi gerakan yang berbeda tetapi dalam waktu singkat, tidak diikuti dengan pola yang teratur dan tidak dapat diperkirakan. Gerakan yang tidak teratur ini dapat disebabkan oleh faktor-faktor random seperti bencana, perubahan pemerintahan, dan lain-lain. Karena gerakan-gerakan ini tidak dapat diperkirakan, maka ukuran ketidakteraturan masa lalu tidak berguna untuk model ini.

Rumus Umum dari tren linier adalah :

$$\hat{Y} = a + bX \qquad (1)$$

Dimana :

$\hat{Y}$ = variabel yang akan diramalkan, dalam hal ini adalah penjualan produk perusahaan

a = konstanta, yang akan menunjukkan besarnya harga $\hat{Y}$ apabila X = 0

b = variabilitas per X, yaitu menunjukkan besarnya perubahan nilai Y dari setiap perubahan satu unit X

X = unit waktu/periode, yang dapat dinyatakan dalam minggu, bulan, semester, tahun, dan lain sebagainya tergantung pada kesesuaian yang ada di data perusahaan

Tren linier dapat dihitung dengan beberapa metode, antara lain [5]:

a. Metode *Least Square*
b. Metode Bebas
c. Metode setengah rata-rata (*Semi Everage*)
d. Metode rata-rata bergerak (*Moving Average*)

a) Metode *Least Square*

Dari metode-metode yang ada metode *least square* merupakan metode yang dianggap paling mudah dan gampang dipraktikkan. Metode ini digunakan pada waktu data yang tersedia mempunyai kecenderungan berbentuk garis lurus. Untuk melihat tren linier sebaiknya digunakan suatu periode sekurang-kurangnya meliputi satu siklis, jika lebih dari satu siklis akan lebih baik.

Seperti yang telah dijelaskan sebelumnya bahwa antara nilai-nilai data penjualan berkala Y1, Y2, Y3, ……Yn dengan nilai-nilai tren $\hat{Y}1$, $\hat{Y}2$, ……. $\hat{Y}n$ yang diperoleh dari persamaan tren linier $\hat{Y}i = a + bX$ mempunyai selisih atau eror sebesar $e_i = Yi - \hat{Y}i$, sehingga jumlah seluruh selisih dari semua titik adalah $\sum e_i$. Oleh karena itu nilai e_i bisa bertanda positif atau bertanda negatif, maka agar menjadi nilai bertanda positif, dapat di ambil kuadrat dari semua e_i, yaitu e_i^2 sehingga diperoleh jumlah kuadrat selisih, yaitu $\sum e_i^2 = \sum(Y_i - \hat{Y}_i)^2$. Dengan

meminimumkan bentuk kuadrat ini, maka akan di peroleh persamaan tren linier yang mempunyai kesalahan atau selisih atau eror minimum (paling kecil). Hal ini akan terpenuhi jika [5] :

$$a = \frac{\sum Y}{n} \quad (2)$$

$$b = \frac{\sum XY}{\sum x^2} \quad (3)$$

$$\bar{t} = \frac{\sum t}{n} \quad (4)$$

$$X = t - \bar{t} \quad (5)$$

Dimana :

Y= Penjualan

X = Kode periode

t = Waktu

$\bar{t}$ = Waktu rata-rata dalam 1 periode

n = Banyaknya data

Dengan syarat $\sum X = 0$, dimana n adalah sama dengan jumlah data.

b) Metode Bebas

Metode bebas merupakan cara yang paling sederhana dan mudah untuk menetukan tren dari data berkala. Langkah-langkah yang diperlukan untuk menetukan persamaan tren dengan cara ini adalah sebagai berikut [5]:

4. Buatlah sumbu datar X dan sumbu tegak Y dalam sistem koordinat *Cartesius*.
5. Buatlah diagram pencar (*scatter diagram*) dari pasangan titik pasangan titik (X,Y) yang menyatakan kaitan antara waktu dan nilai data berkala.
6. Tariklah garis linieryang arahnya mengikuti arah penyebaran nilai-nilai data berkala.
7. Pilihlah dua titik kapal sembaranguntuk menentukan persamaan tren linier, misalnya (X1,X1) dan (X2,Y2). Pilih salah satu periode waktu data berkala sebagai titik asal (X=0).
8. Masukanlah atau substitusikanlah nilai-nilai X dan Y dari dua titik yang telah dipilih pada rumus persamaan umum tren linier atau memakai persamaan berikut:

$$Y - Y1 = \frac{(Y2 - Y1)}{(X1 - X2)}(X - X1) \qquad (6)$$

9. Selanjutnya tentukanlah nilai-nilai tren linier dengan memakai persamaan yang telah diperoleh tersebut.

c) Metode Setengah Rata-Rata (*Semi Everage*)

Penetuan persamaan tren linier dengan metode setangah rata-rata (semi rata-rata) dilakukan dengan tahapan-tahapan berikut [5]:

1. Bagilah data berkala menjadi dua kelompok yang sama banyak, katakanlah kelompok 1 dan kelompok 2.
2. Tentukanlah rata-rata hitung masing-masing kelompok, katakanlah Y1 dan Y2.
3. Tentukanlah dua titik yaitu (X1, Y1) dan (X2, Y2), dimana absis X1 dan X2 ditentukan dari periode waktu data berkala.
4. Tentukanlah nilai a dan b dengan mensubstitusikan nilai-nilai X dan Y dari dua titik tersebut pada persamaan tren.

Masalah akan muncul ketika membagi data berkala menjadi dua kelompok yang sama banyak. Dalam hal banyak data berkala genap, maka tidak akan ada masalah, karena tiap kelompok akan terdiri atas nilai data berkala yang sama banyaknya. Akan tetapi, bila banyaknya data berkal ganjil, agar masing-masing kelompok terdiri atas nilai data berkala yang sama banyaknya, maka dapat dilakukan dengan dengan dua cara, yaitu pertama menghilangkan nilai data paling tengah atau kedua memasukan nilai data paling tengah tersebut pada masing-masing kelompok.

d) Metode rata-rata bergerak (*Moving Average)*

Metode ini melakukan peramalan dengan menggunakan *rata-rata* penjualan masa lalu. Ramalan tahun ini merupakan *rata-rata* tahun sebelumnya sedangkan tahun berikut merupakan rata-rata tahun sebelumnya dengan tahun sekarang. Metode rata-rata bergerak (*moving average)* ditetukan dengan cara berikut.

1. Hitunglah rata-rata Y dari sebanyak *n* nilai data yang paling awal.
2. Lupakan nilai data yang pertama
3. Ulangi tahap 1 dan 2 sampai data yang terakhir telah digunakan.

Nilai Y rata-rata :

$$\frac{Y1 + Y2 + Y3 \ldots . + Yn}{n} \tag{7}$$

Bagian pembilang masing-masing disebut total bergerak menurut total *n* yang bergantung pada periode waktu data berkala. Rata-rata ini dikatakan rata-rata bergerak karena setelah rata-rata dihitung kemudian diikuti gerakan 1 periode kebelakang. Rata-rata bergerak ditempatkan pada pusat dari *n* dari periode yang digunakan. Bila data berkala merupakan data tahunan, maka urutan *n* adalah dalam tahunan. Bila data berkala merupakan data bulanan, maka urutan dapat mengenal bergerak tiga bulan, rata-rata bergerak satu tahun, rata-rata bergerak lima tahun, rata-rata bergerak sepuluh tahun, rata-rata, dan seterusnya.

2.3.3.2. Tren Kuadratis

Kelemahan tren linier adalah kurang baik dipakai untuk mewakili data berkala yang jangka waktunya panjang, karena sering meleset atau tidak tepat akibat perubahan waktu. [5]

Rumus Umum dari tren kuadrat adalah[8] :

$$\hat{Y} = a + bX + cX^2 \tag{8}$$

Dimana nilai X merupakan tahun kode dan X= t - $\bar{t}$.

Dengan metode *least square* diperoleh :

$$a = \frac{(\sum Y)(\sum X^4) - (\sum X^2 Y)(\sum X^2)}{n(\sum X^4) - (\sum X^2)^2} \quad (9)$$

$$b = \frac{\sum XY}{\sum x^2} \quad (10)$$

$$c = \frac{(\sum X^2 Y) - (\sum X^2)(\sum Y)}{n(\sum X^4) - (\sum X^2)^2} \quad (11)$$

2.3.3.3. Tren Eksponensial

Bentuk umum dari tren eksponensial adalah[8] :

$$Y_t = a\,(1 + b)^{t-\bar{t}} \quad (12)$$

atau dalam bentuk kode

$$Y_t = a\,(1 + b)^{X} \quad (13)$$

Persamaan ini disebut eksponensial karena waktu $\bar{t}$ sebagai pangkat. Dalam persamaan itu t adalah tahun ditengah dari periode yang dipelajari. Sedangkan menunjukkan tingkat perubahan nilai variabel Y_t per tahun.dan tingkat perubahan itu konstan. Nilai koefisien yang dicari dari persamaan itu adalah a dan b, jika diketahui pasangan data X dan Y dalam deret waktu.

Berikut adalah pesamaannya [8] :

$$a = anti\ Ln\ \frac{\sum Ln\,Y}{n} \quad (14)$$

$$b = anti\ ln\ \frac{\sum X\ Ln\,Y}{\sum (X)^2} \quad (15)$$

2.3.4. Penentuan tingkat kesalahan

Setiap peramalan pastinya memiliki ukuran kesalahan. Berikut ini adalah ukuran-ukuran yang dapat dijadikan acuan untuk itu.

2.3.4.1. *Mean Squared Error MSE*

Merupakan cara kedua untuk mengukur kesalahan peramalan keseluruhan. MSE merupakan rata-rata selisih kuadrat antara nilai yang diramalkan dan yang diamati. Rumusnya adalah [7]:

$$MSE = \frac{Kesalahan\ Peramalan}{n} \tag{16}$$

Kekurangan penggunaan MSE adalah bahwa cenderung menonjolkan deviasi yang besar karena adanya pengkuadratan.

2.3.4.2. *Mean Absolute Percent Error*

Masalah yang terjadi dengan MAD dan MSE adalah bahwa nilai mereka tergantung pada besarnya unsur yang diramal. Jika unsur tersebut dihitung dalam satuan ribuan, maka nilai MAD dan MSE bisa menjadi sangat besar. Untuk menghindari masalah ini, kita dapat menggunakan *mean absolute percent error* (MAPE). MAPE dihitung sebagai rata-rata diferensiasi absolut antara nilai yang diramal dan aktual, dinyatakan sebagai persentase nilai aktual. jika kita memiliki nilai yang diramal dan aktual untuk *n* periode, MAPE dihitung sebagai [7]:

$$MAPE = \frac{\frac{\sum_{i=1}^{n} |\ aktual_i - ramalan_i|}{aktual_i}}{n} \text{x } 100 \tag{17}$$

MAPE mungkin merupakan perhitungan yang paling mudah diartikan. Sebagai contoh, MAPE merupakan pernyataan yang jelas, yang tidak bergantung pada permasalahan seperti banyaknya data input.

2.4. Definisi dan Simbol Alat Bantu Analisa dan Perancangan Sistem

Dalam melakukan analisa dan perancangan sistem, digunakan beberapa perangkat yang akan mendeskripsikan hasil analisa yang akan membantu dalam merancang suatu sistem. Penggunaan perangkat tersebut akan memudahkan bagi sistem analis maupun programmer dalam membaca atau memahami sistem yang sedang berjalan.

2.4.1. Diagram Hubungan Entitas (*Entity Relationship Diagram*)

Diagram hubungan entitas adalah diagram yang menggambarkan hubungan entitas yang tersimpan. Adapun simbol-simbol yang digunakan dalam Diagram Hubungan Entitas (*Entity Relational Diagram*) adalah sebagai berikut :

a. Entitas (*Entity*)

   Entitas (*Entity*) adalah obyek yang dapat di definisikan dalam lingkungan pemakai.

b. Hubungan (*Relationship*)

   Hubungan (*Relationship*) adalah hubungan yang terjadi antara dua atau lebih entity. Biasanya setiap relationship berisikan kata kerja untuk menghubungkannya.

c. Atribut (Atribute)

Atribut (*Atribute*) adalah karakteristik dari entity yang menyediakan penjelasan detail tentang karakter entity tersebut.

2.4.2. Diagram Alur Data (*Data Flow Diagram*)

Diagram alur data (*Data Flow Diagram*) pada umumnya digunakan untuk memudahkan pemakaian suatu aplikasi. Adapun pengertiannya adalah suatu gambaran sistem secara logika, dan tidak mempertimbangkan lingkungan fisik dimana data tersebut mengalir. Adapun simbol-simbol yang digunakan adalah sebagai berikut :

a. Proses.

Proses merupakan suatu tindakan yang akan diambil terhadap data yang masuk. Karena proses adalah tindakan, maka proses berisi kata kerja. Dan proses memiliki satu atau lebih input dan output data.

b. *Terminator*.

Terminator adalah sesuatu yang berada diluar sistem, dapat berupa sistem lain atau bagian sistem. Dalam pemberian nama pada terminator menggunakan kata benda tulis didalam kotak perseginya.

c. Alur.

Alur adalah merupakan aliran data yang masuk dan keluar dari sistem.

d. Data *Store* (Penyimpanan Data).

Data *Store* (Penyimpanan data) biasa digunakan untuk menyimpan data-data.

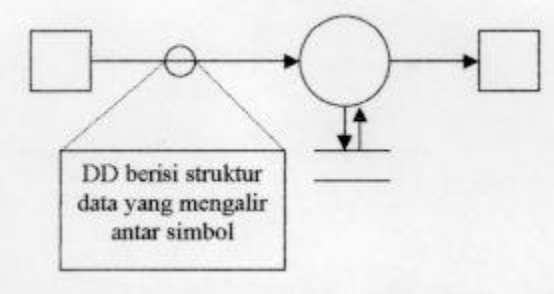
2.4.3. Alir Dokumen (*flow map*)

Alir Dokumen adalah bagan yang menunjukan arus pekerjaan secara keseluruhan dari sistem. Bagan ini menjelaskan urutan-urutan dari prosedur-prosedur yang ada didalam sistem.

2.4.4. Kamus Data

Kamus data (*Data Dictinory*) adalah katalog fakta tentang data dan kebutuhan-kebutuhan informasi dari suatu sistem informasi. Dengan kamus data analis sistem denganpemakai sistem tenteng cara yang mengalir ke sistem, yaitu tentang data yang masuk ke sistem dan tentang informasi yang dibutuhkan oleh pemakai sistem.[4]

Pada tahap perancangan sistem, kamus data digunakan untuk merancang input, merancang laporan-laporan dan *database*. Kamus data dibuat berdasarkan arus data yang ada di DFD (*Data Flow Diagram*). Arus data di DFD sifatnya adalah global, hanya ditujukkan nama arus datanya saja. Berikut adalah gambar yang menunjukkan hubungan antara DFD dengan kamus data.



Gambar 2.1 Hubungan dengan DFD dan Kamus Data

2.4.4.1. Isi Kamus Data

Kamus data harus dapat mencerminkan keterangan yang jelas tentang data yang dicatatnya, seperti :

1. Nama arus data

   Karena kamus data dibuat berdasarkan arus data yang mengalir di DFD, maka nama arus data juga dicatat di kamus data.

2. Alias

   Alias atau nama lain dari data dapat ditulis jika ada nama lain. Alias digunakan hanya untuk membedakan nama data yang satu dengan yang lainnya.

3. Bentuk data

   Bentuk data harus dicatat dalam kamus data, karena dapat digunakan untuk mengelompokan kamus data ke dalam kegunaannya sewaktu perancangan sistem.

4. Arus data

   Arus data menunjukan dari mana data mengalir dan kemana data akan menuju. Keterangan arus data harus dicatat di kamus data supaya memudahkan mencari arus data ini di DFD.

5. Penjelasan

   Penjelasan dapat diisi dengan keterangan-keterangan tentang arus data tersebut.

6. Periode

   Periode ini menunjukan kapan terjadinya arus data ini.

7. Volume

Volume yang perlu dicatat adalah tentang volume rata-rata dan volume puncak arus data.

8. Struktur data

   Struktur data menunjukan arus data yang dicatat di kamus data yang terdiri dari *item-item* apa saja.

2.4.5. Relasi *Database*

Dalam sebuah relasi *database*, tabel yang memuat kunci utama yang digunakan sebagai referensi kunci utama tabel lainnya disebut tabel induk (*parent*) sedangkan tabel yang memuat kunci tamu disebut tabel anak (*child*).

Ada 3 bentuk relasi yang lazim digunakan, antara lain:

1. Relasi One to One (1 : 1)

   Relasi ini meyatakan bahwa hanya satu *record* pada tabel anak yang dapat direlasikan dengan satu *record* pada tabel induk. *Field* kunci utama yang digunakan sebagai kunci utama pada tabel lainnya disebut *field* relasi.

2. Relasi One to Many (1 : N)

   Relasi ini meyatakan bahwa beberapa *record* pada tabel anak dapat direlasikan dengan satu *record* pada tabel induk.

3. Relasi Many to Many (N : N).

   Relasi ini meyatakan bahwa beberapa *record* pada tabel anak dapat direlasikan dengan satu *record* pada tabel induk, dan sebaliknya beberapa *record* pada tabel induk dapat direlasikan pada satu *record* pada tabel anak. Namun dalam

perancangan tabel. Relasi *many to many* dari 2 tabel tidak boleh dilakukan, karena hal itu menjadikan *field* kunci tidak lagi bersifat unik.

2.5. Normalisasi

Suatu *file* yang terdiri dari beberapa grup elemen yang berulang-ulang perlu diorganisasikan kembali. Proses untuk mengorganisasikan *file* untuk menghilangkan grup elemen yang berulan-ulang ini disebut dengan Normalisasi (*Normalization*).

Pada proses normalisasi selalu diuji pada beberapa kondisi, apakah ada kesulitan pada saat menambah (*insert*), menghapus (*delete*), mengubah (*update*), membaca (*retrieve*) pada suatu *database*. Bila ada kesulitan pada pengujian tersebut maka relasi tersebut dapat dipecahkan menjadi beberapa tabel lagi atau dengan kata lain perancangan yang dilakukan belum mendapatkan suatu *database* yang optimal. Ada beberapa macam kunci (*key Function*) yang digunakan untuk proses pencarian, penyaringan, hapus, dan lain sebagainya yang biasa digunakan dalam pengolahan *database*, yaitu sebagai berikut :

a. Kunci Calon (*Candidat Key*)

   Kunci kandidat adalah satu atribut atau set minimal atribut yang mengidentifikasi secara unik suatu kejadian yang spesifik dari suatu entity. Satu set minimal atribut menyatakan secara tak langsung dimana kita tak dapat membuang beberapa atribut dalam set tanpa merusak kepemilikan unik. Jika satu kunci kandidat berisi lebih dari satu atribut, maka biasanya disebut sebagai kunci gabungan (*composite key*).

b. Kunci Primer (*Primary Key*)

Kunci primer adalah satu atribut atau satu set minimal atribut yang tidak hanya mengidentifikasi secara unik suatu kejadian yang spesifik, akan tetapi juga dapat mewakili setiap kejadian dari suatu entity. Setiap kunci kandidat punya peluang menjadi *Primary Key*, akan tetapi sebaiknya dipilih satu saja yang dapat mewakili secara menyeluruh terhadap entity yang ada.

c. Kunci Alternatif (*Alternative Key*)

Kunci alternatif adalah kunci kandidat yang tidak dipakai sebagai *Primary Key*. Sering kali kunci alternatif ini digunakan sebagai kunci pengurutan dalam pembuatan laporan.

d. Kunci Tamu (*Foreign Key*)

Kunci tamu adalah satu atribut atau set atribut yang melengkapi satu hubungan (*relationship*) yang menunjukan ke induknya. Kunci tamu ditempatkan pada entity anak dan sama dengan kunci primary key induk yang direlasikan. Hubungan antara entity induk dengan anak adalah hubungan satu lawan banyak anak (*one to many relationship*).

e. Kunci Super (*Super Key*)

Kunci super adalah himpunan dari satu atau lebih entitas yang dapat digunakan untuk mengidentifikasi secara unik sebuah entitas dan entitas set.

Teknik normalisasi ini juga merupakan suatu teknik yang menstrukturkan data dalam cara tertentu untuk membantu mengurangi atau mencegah timbulnya

masalah yang berhubungan dengan pengolahan *database*. Proses normalisasi menghasilkan struktur *record* yang konsisten secara logic yang mudah untuk dimengerti dan sederhana dalam pemeliharaannya. Adapun langkah-langkah pembentukan normalisasi adalah sebagai berikut :

1. Bentuk Tidak Normal (*Unnormalized Form*)

   Bentuk ini merupakan kumpulan data yang direkam, tidak ada keharusan mengikuti suatu format tertentu. Dapat saja data tidak lengkap atau terduplikasi, data dikumpulkan apa adanya sesuai dengan saat pemasukan.

2. Bentuk Normal Kesatu (1NF / *First Normal Form*)

   Suatu relasi jika dan hanya jika sifat dari setiap relasi atributnya bersifat atomik, yaitu jika tidak dipecah maka tidak memiliki sifat induknya.

3. Bentuk Normal Kedua (2 NF / *Second Normal Form*)

   Bentuk mormal kedua mempunyai syarat yaitu bentuk data telah memenuhi criteria bentuk normal kesatu. Atribut bukan kunci haruslah bergantung secara fungsi pada kunci utama atau *primary key*. Sehingga untuk membentuk normal kedua haruslah sudah ditentukan kunci-kunci *field*. Kunci *field* haruslah unik dan dapat mewakili atribut lain yang menjadi anggotanya.

4. Bentuk Normal Ketiga (3 NF / *Third Normal Form*)

   Untuk menjadi normal ketiga maka relasiharuslah dalam bentuk normal kedua dan semua atribut bukan primer tidak mempunyai hubungan yang transitif. Dengan kata lain, setiap atribut bukan kunci haruslah bergantung hanya pada *primary key* dan pada *primary key* secara menyeluruh.

5. *Boyce Codd Normal Form* (BCNF)

   BCNF mempunyai paksaan yang lebih kuat dari bentuk normal ketiga, dan relasi harus dalam bentuk normal kesatu dan setiap atribut harus bergantung fungsi pada atribut *super key*.

6. Bentuk Normal Keempat (4 NF)

   Relasi R adalah bentuk 4 NF jika dan hanya jika relasi tersebut juga termasuk BCNF dan semua ketergantungan *multivalue* adalah juga ketergantungan fungsional.

7. Bentuk Normal Kelima (5 NF)

   Disebut juga PJNF (*Projection Join Normal Form*) dari 4 NF dilakukan dengan menghilangkan ketergantungan join yang bukan merupakan kunci kandidat.

2.6. *Database*

Biasanya aplikasi berbasis komputer yang digunakan pada berbagai institusi menggunakan *database* atau basis data. Basis data merupakan suatu bentuk pengelolaan data yang ditujukan agar pengaksesan terhadap data dapat dilakukan dengan mudah. Sistem yang ditujukan untuk menangani basis data biasa disebut DBMS (*Database Management System*). Dengan menggunakan DBMS, pemakai dapat melakukan hal-hal berikut dengan mudah, seperti :

a. Menambah data.
b. Menghapus data.
c. Mengubah data.

d. Mencari data.

e. Menampilkan data dengan kriteria tertentu.

f. Mengurutkan data.

Sistem basis data adalah suatu sistem informasi yang mengintegrasikan kumpulan dari data yang saling berhubungan dengan yang lain dan membuat menjadi beberapa aplikasi dalam suatu organisasi. Adapun keuntungan dari basis data adalah :

a. Salah satu komponen penting dalam sistem informasi.

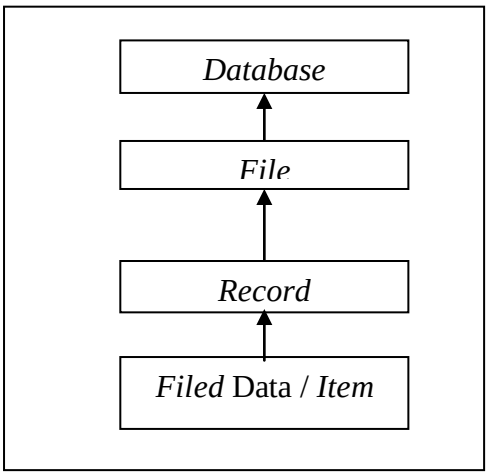
b. Menentukan kualitas informasi yang bersifat akurat.

c. Mengurangi duplikasi data (*data redundancy*).

d. Hubungan data dapat ditingkatkan (*data relatability*).

e. Mengurangi pemborosan tempat simpanan luar sampai dengan membentuk suatu basis data.

Dalam basis data memiliki hirarki data, gambar berikut :



Gambar 2.2 Hirarki Data

2.6.1. Elemen-Elemen Sistem *Database*

1. Karakter (*character*)

   Merupakan bagian data yang terkecil, dapat berupa karakter numeric, huruf, ataupun karakter khusus (*special characters*) yang membentuk suatu item data atau *field*.

2. *Field*

   Mewakili suatu atribut dari *record* yang menunjukan suatu item dari data, seperti misalnya nama, alamat, dan lain sebagainya. Kumpulan dari *field* membentuk suatu *record*.

   a. Field name, harus diberi nama untuk membedakan *field* yang satu dengan yang lainnya.

   b. Field representation, tipe *field* dapat berupa karakter, teks, tanggal, angka, dan sebagainya. Lebar *field* yaitu ruang maksimum yang dapat diisi dengan karakter data.

   c. *Field value* adalah isi dari *field* untuk masing-masing *record*.

3. *Record* adalah Kumpulan dari *field* membentuk suatu *record*. *Record* mengambarkan suatu unit data individu yang tertentu. Kumpulan dari *record* menbentuk suatu *file*.

4. *File*

File terdiri dari *record* yang menggambarkan satu kesatuan data yang sejenis. Misalnya *file* mata pelajaran berisi data tentang semua mata pelajaran yang ada.

5. *Database*

Kumpulan dari *file* atau tabel yang membentuk suatu *database*.

2.7. Pengenalan Microsoft *Visual basic* 6.0

Dalam situs www.geocities.com, *Visual basic* (VB) adalah salah suatu sarana (*tool*) untuk membangun aplikasi dalam lingkungan Windows. Dalam pengembangan aplikasi, *visual basic* menggunakan pendekatan visual untuk merancang *user* interface dalam bentuk form, sedangkan untuk kodingnya menggunakan bahasa basic (standar) yang cenderung mudah dipelajari oleh penggunanya. *Visual basic* telah menjadi sarana (tool) yang terkenal bagi pemula maupun yang sudah mahir.

Microsoft *visual basic* memiliki lingkungan pengembangan terpadu bagi programmer dalam mengembangkan aplikasinya atau biasa dikenal dengan IDE *(Integrated Developement Environment)*. Dengan menggunakan IDE programmer dapat membuat *user* interface, melakukan koding, melakukan testing dan debuging serta menkompilasi program. Penguasaan yang baik akan IDE akan sangat membantu programmer dalam mengefektifkan tugas-tugasnya sehingga dapat bekerja dengan efisien.

2.7.1. Tampilan Microsoft *Visual basic* 6.0

Microsoft *visual basic* memiliki beberapa komponen yang dapat digunakan dalam membuat suatu aplikasi. Komponen-komponen tersebut adalah :

(1) *Control Menu*, (2) *Menu*, (3) *ToolBar*, (4) *Project Explorer*, (5) *Properties*, (6) *Form Layout*, (7) *ToolBox*, (8) *Form*, dan (9) Jendela Kode.

1. *Control Menu*

   Control Menu adalah menu yang dapat digunakan untu memanipulasi layar *visual basic*. Di menu ini bisa digunakan untuk merubah ukuran, memindahkan, dan menutup layar *visual basic*. Di menu ini juga memiliki beberapa perintah.

2. *Menu*

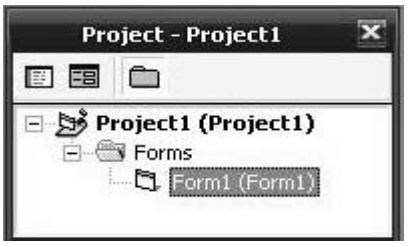
   Menu dalam layar *visual basic* berisikan perintah tertentu untuk melakukan tugas tertentu. Sebagian dari isi *menu* ini hampir sama dengan program-program windows yang lainya, seperti menyimpan *project*, membuka *project*, dan lain-lain.

3. *ToolBar*

   ToolBar adalah tombol-tombol yang mewakili suatu perintah dari *visual basic*. Biasanya tombol-tombol ini yang sering digunakan oleh *user* dalam *visual basic*.

4. *Project Explorer*

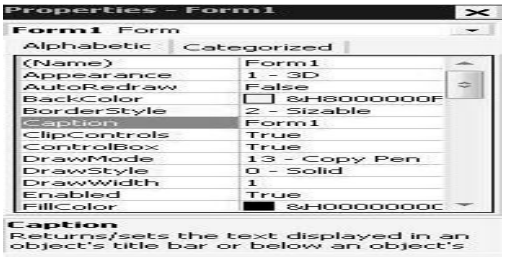
   Project Explorer adalah yang mengandung semua *file* di dalam aplikasi *visual basic*. Dalam *project explorer* ditampilkan semua *file* yang terdapat pada aplkasi yang sedang dibuat. Berikut gambar *project explorer*.



Gambar 2.3 *Project Explorer Visual basic*

5. *Properties*

   Properties adalah jendela yang mengandung semua informasi mengenai objek yang terdapat di dalam aplikasi *visual basic*. Biasanya di dalam *properties* terdapat properti yang dapat diatur langsung dari jendela *properties*. Berikut adalah tampilan *properties*.



Gambar 2.4 *Poperties Visual basic*

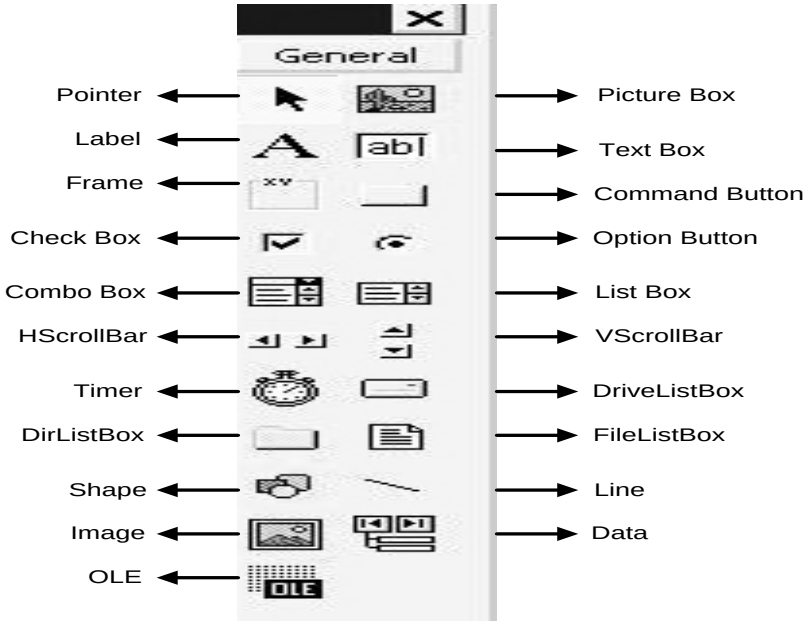
6. *Form Layout*

   Form layout adalah jendela yang akan menunjukan bagaimana form bersangkutan ditampilkan ketika *runtime*. Didalam form *layout* juga terdapat menu-menu yang akan muncul jika mengklik kanan pada mouse.

7. *ToolBox*

   Toolbox adalah layar yang berisikan komponen-komponen yang dibutuhkan untuk membuat aplikasi. Dari *toolbox* ini dapat mengambil komponen-komponen (*object*) yang akan dibutuhkan pada form untuk

dibuat dalam aplikasi *visual basic*. Berikut gambar *toolbox* dalam *visual basic*.



Gambar 2.5 *ToolBox Visual basic*

Funsi dari kontrol-kontrol tersebut, adalah :

a. *Pointer* berfungsi untuk memilih kontrol yang sudah berada pada form.
b. *PictureBox (Pic)* berfungsi untuk menampilkan image dengan format: BMP, DIB (*bitmap*), ICO (*icon*), CUR (*cursor*), WMF (*metafile*), EMF (*enhanced metafile*), GIF dan JPEG atau JPG.
c. *Label* berfungsi untuk menampilkan teks tetapi pemakai tidak dapat berinteraksi dengannya atau mengubahnya.
d. *TextBox* berfungsi untuk menampilkan teks dimana pemakai dapat mengisi atau melihat teks yang ditampilkan sebagai output suatu proses.
e. *Frame* berfungsi untuk mengidentifikasi sebuah grup pengontrol.

f. *Command Button* (cmd) berfungsi untuk memberikan sebuah perintah atau tindakan ketika digunakan.

g. *CheckBox* berfungsi untuk memberikan pilihan benar/salah (*True/False*) atau ya/tidak (*Yes/No*).

h. *OptionButton* berfungsi untuk memberikan pilihan lebih dari satu sebagai pilihan terhadap beberapa option yang hanya dapat dipilih satu.

i. *ListBox* berfungsi untuk menampilkan daftar item yang dapat dipilih salah satu.

j. *ComboBox berfungsi untuk unit kontrol* kombinasi antara TextBox dengan ListBox dimana pemasukkan data dapat dilakukan dengan pengetikkan maupun pemilihan.

k. *HScrollBar* atau *Horizontal Scroll Bar* berfungsi untuk menampilkan balok gulung horizontal.

l. *VScrollBar* atau *Vertical Scroll Bar* berfungsi untuk menampilkan balok gulung vertikal.

m. *Timer* berfungsi untuk mengeksekusi waktu kejadian pada rutin program termasuk selang waktu (interval).

n. *DriveListBox* berfungsi untuk menampilkan daftar *drive*.

o. *DirListBox* berfungsi untuk menampilkan daftar direktori pada drive terpilih.

p. *FileListBox* berfungsi untuk menampilkan daftar *file* pada drive dan direktori terpilih.

q. *Shape* berfungsi untuk membuat lingkaran, *elips,* dan kotak pada *form.*

r. *Line* berfungsi untuk membuat garis lurus pada *form.*

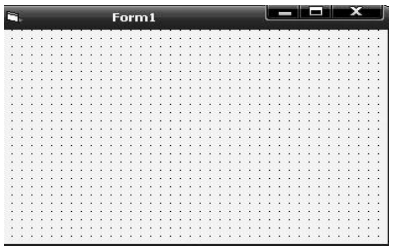
s. *Image* berfungsi untuk menampilkan gambar : .bmp, .gif, .jpg, .wmf, dan .ico. Perbedaan dengan *picture box* adalah *control* ini memberikan perintah seperti *command button* saat pemakai mengklik gambar dengan menggunakan *mouse*.

t. *Data* berfungsi untuk menampilkan *database* pada suatu *form*.

u. *OLE* atau *Object Linking and Embedding* berfungsi untuk memungkinkan pemakai menempilkan suatu objek dari aplikasi yang mendukung OLE ke aplikasi *Visual basic*.

8. Form

   Form adalah daerah kerja utama yang berguna untuk merancang program aplikasi dalam *visual basic*. *Form* ini bisa diletakkan berbagai macam objek seperti *teks*, gambar, dan sebagainya yang terdapat di *toolbox visual basic*. Berikut gambar form dalam *visual basic*.



Gambar 2.6 *Form Visual basic*

9. Jendela Kode

   Jendela kode adalah jendela yang digunakan untuk menuliskan perintah-perintah yang merupakan instruksi dalam membuat aplikasi *visual basic*.

2.8. Microsoft SQL *Server*

Microsoft SQL (*Structured Query Language*) Server merupakan salah satu bahasa standar yang digunakan untuk mengakses basis data relasional. *Query* adalah nilai/*value* yang turut merepresentasikan deskripsi dari suatu objek atau kejadian (*event*) dan informasi merupakan hasil dari pengolahan data dalam suatu bentuk yang lebih berguna dan lebih berarti bagi penerimanya yang menggambarkan suatu kejadian-kejadian (*event*) yang nyata (*fact*) yang digunakan untuk pengambilan keputusan. Di SQL *Server* terdapat 3 buah fungsi utama, diantaranya :

1. DDL (*Data Definition Language*).

DDL berungsi untuk merancang sebuah basis data, tabel, maupun yang lainnya. Fungsi-fungsi DDL seperti :

a. *Create*, digunakan untuk membuat sebuah maupun lebih basis data, tabel dan fungsi-fungsi yanglainnya.
b. *Alter*, digunakan untuk merubah tabel yang telah dibuat sebelumnya.
c. *Drop*, digunakan untuk menghapus tabel atau basis data yang ada.

2. DML (*Data manipulation Language*).

DML berfungsi untuk mengambil dan memanipulasi data pada suatu *database*. Fungsi-fungsi DML diantaranya :

a. *Insert*, digunakan untuk memasukkan data kedalam tabel.
b. *Select*, digunakan untuk membaca (*query*) isi tabel.

c. *Update*, digunakan untuk melakukan perubahan data yang sudah disimpan.

d. *Delete*, digunakan untuk menghapus data dalam tabel.

3. DCL (*Data Control Language*).

DCL berfungsi untuk mengelola otorisasi *user*.

2.8.1. Tipe Data SQL *Server*

Dalam membuat tabel dalam SQL *Server*, harus mengetahui tipe-tipe data serta fungsinya. Tipe data dalam SQL ada 4, yaitu:

a. Integer, diantaranya :

1. Bit : Integer dengan nilai 0 dan 1.
2. Int : Nilai integer dengan nilai antara -2^-3 (2.147.483.648) sampai 2^3-1 (2.147.483.647).
3. Decimal atau Numeric : Angka antara -10^38 sampai 10^38-1.
4. Money : nilai yang berhubungan dengan mata uang dari -2^63(-922.377.203.685.477,5808) sampai 2^63-1 (922.377.203.685.477,5807).
5. Float : -214.748,3648 sampai 1.79E+308.
6. Real : -3.40E+38 sampai 3.40E+38.
7. DateTime : 1 Januari 1973 sampai 31 Desember 9999.
8. SmallDateTime : 1 Januari 1900 sampai 6 Juni 2079, dengan ketelitian hingga 1 menit.

b. String

1. Char : Jumlah karakter tetap dengan ukutan maksimal 8000 karakter.
2. Varchar : Jumlah karakter variabel dengan ukuran maksimal 8000 karakter.
3. Text : Jumlah karakter variabel dengan ukuran maksimal 2.147.483.647 karakter.

c. Unicode String

1. NChar : Jumlah karakter tetap dengan ukuran maksimal 4000 karakter.
2. NVarchar : Jumlah karakter variabel dengan ukuran maksimal 4000 karakter.
3. NText : Jumlah karakter variabel dengan ukuran maksimal 1.073.741.823 karakter.

d. Binary String

1. Binary : Jumlah biner tetap dengan ukuran maksimal 8000 bytes.
2. Varbinary : Jumlah biner variabel dengan ukuran maksimal 8000 bytes.
3. Image : Jumlah karakter variabel dengan ukuran maksimal 2.147.483.647 bytes.