BAB II

TEORI PENUNJANG

2.1. RISET DAN ANALISA PENJUALAN

Riset Analisa Penjulan meliputi analisa-analisa catatan-catatan yang ada mengenai penjualan, serta pembuatan ramalan penjualan, mengenai jumlah untuk setiap jenis barang,menentukan potensi penjualan dan quota penjualan ,selain daripada itu mencakup juga analisa penjualan menurut jenis produksi, langganan, dan daerah geografis. Ramalan penjualan dapat menggambarkan kemampuan menjual pada waktu yang akan datang. Jadi ramalan penjualan berfungsi sebagai dasar perencanaan produksi, perencanaan bahan mentah, perencanaan peralatan, perencanaan tenaga kerja maupun perencanaan keuangan.

Analisa Penjualan adalah analisa sesungguhnya dari hasil penjualan. Tujuan analisa ini adalah untuk mencari daerah-daerah yang kuat dan yang lemah, jenis-jenis produk yang sangat menguntungkan, para langganan /nasabah yang mampu membeli dalam jumlah banyak dan sebagainya. Keterangan-keterangan atau data-data demikina memungkinkan perusahaan untuk mengkonsentrasikan usah penjualannya dimana akan memberikan hasil yang optimal. Analisa penjualan biasanya didasarkan atas 3 hal, yaitu :

a. Analisa Penjualan menurut Daerah

Untuk melakukan analisa ini adalah dengan menentukan satuan daerah geografis yang digunakan apakah tingkat kecamatan, tingkat kabupaten atau tingkat propinsi atau cukup tingkat daerah tertentu. Perbandinagn antara antara potensi dan hasil yang sesungguhnya dicapai menjadi lebih sederhana apabila keduanya dalam satuan yang sama. Hasil penjualan kemudian ditabulasikan (dibuat tabel-tabel) menurut satuan daerah,

sehingga dengan demikian bisa diketahui daerah-daerah yang kurang berpotensi dan daerah-daerah yang sangat berpotensi.

b. Analisa Penjualan Menurut Barang Produksi (Produk)

Dengan melakukan analisa hasil penjualan menurut produk, memungkinkan untuk mengetahui produk mana yang memberikan kontribusi yang paling besar terhadap hasil penjualan dan produk mana yang kurang menguntungkan. Analisa penjualan menurut produk akan lebih efektif apabila dikombinasikan dengan analisa daerah. Analisa semacam ini akan memudahkan untuk mengetahui di daerah mana usaha penjualan harus dikonsentrasikan dan untuk jenis produk yang mana.

c. Analisa Penjualan Menurut Langganan / nasabah.

Dengan melakukan analisa hasil penjualan menurut golongan langganan, memungkinkan untuk mengetahui golongan langganan yang memberikan kontribusi yang terbesar. Kombinasi antara analisa daerah, analisa produk dan analisa langganan sangat membantu dalam mengetahui kelemahan-kelemahan program penjualan untuk suatu daerah tertentu.

Untuk dapat melakukan analisa penjualan dengan baik diperlukan data-data antara lain data hasil penjualan. Time series sales data adalah data mengenai hasil penjualan yang terus menerus atau dari waktu ke waktu.

2.2. PENARIKAN GARIS TREND

2.2.1. METODE KUADRAD TERKECIL

Metode kuadrat terkecil digunakan untuk mencari rata-rata hitung suatu grup. Nilai rata-rata ini digunakan sebagai wakil atau pencerminan nilai dari grup tersebut. Rata-rata mempunyai 2 ciri-ciri matematik, yaitu :

1. Jumlah aljabar dari deviasi masing-masing data terhadap rata-ratanya adalah nol.
2. Jumlah deviasi kuadrat masing-masing data terhadap rata-ratanya adalah minimum.

Contoh :

Nilai Y	Deviasi dari rata-ratan $y = Y - Yr$	Deviasi Kuadrat y^2
1 4 10	-4 -1 5	16 1 25
Total = 15 Yr = 15/3 = 5	0 (Ciri 1)	42 (Ciri 2 - Kuadrat terkecil)

Konsep ini digunakan untuk mencari sebuah garis lurus, dimana harus dipilih penerapan yang terbaik. Penarikan garis lurus di sini harus dapat mewakili titik-titik sebaran variabel X dan variabel Y pada grafik. Garis lurus untuk dependen variabel Y berdasarkan metode kuadrat terkecil, juga memiliki 2 ciri matematik :

1. Jumlah aljabar deviasi nilai masing-masing data terhadap masing-masing nilai yang ditunjukkan oleh garis adalah nol atau $\sum (Y - Yc) = 0$
2. Jumlah deviasi kuadratnya adalah paling kecil atau $\sum (Y - Yc)^2$ adalah minimum.

Untuk memperoleh konstanta a dan b yang belum diketahui dalam persamaan garis lurus

$$Yc = a + bX \qquad (2.1)$$

dengan metode terkecil diperlukan 2 persamaan normal sebagai berikut :

$$\text{I} \quad n\,a + b \sum X = \sum Y$$

$$\text{II} \quad a \sum X + b \sum X^2 = \sum XY \qquad \text{(Pers. 2.2)}$$

Dua persamaan normal tersebut diperoleh sebagai berikut :

Misal $Y_1, Y_2, \ldots\ldots, Y_n$ dan $X_1, X_2, \ldots\ldots X_n$ yang menunjukkan variabel X dan Y, dan

$Y_1 = a + b X_1$

$Y_2 = a + b X_2$

..... = ... + ...

$Y_n = a + b X_n$

$\sum Y = n a + b \sum X$ (persamaan normal I)

Sedangkan untuk memperoleh persamaan normal kedua adalah sebagai berikut :

$X_1 Y_1 = a X_1 + b X_1 X_1$

$X_2 Y_2 = a X_2 + b X_2 X_2$

..... = ... + ...

$X_n Y_n = a X_2 + b X_n X_n$

$\sum XY = a \sum X + b \sum X^2$ (persamaan normal II)

Dengan menggunakan metode determinan kita dapat memperoleh nilai a dan b sebagai berikut :

$a = (\sum Y * \sum X^2 - \sum X * \sum XY) / (N * \sum X^2 - \sum X * \sum X)$

$b = (N * \sum XY - \sum Y * \sum X) / (N * \sum X^2 - \sum X * \sum X)$

(2.3)Namun biasanya, untuk memperoleh nilai konstanta a dan b terlebih dahulu persamaan 2.1 tersebut disederhanakan dengan membuat jumlah nilai X menjadi nol, atau $\sum(X) = 0$. Apabila $\sum(X) = 0$, persamaan normal I menjadi :

$na = \sum(Y)$, atau $a = \sum(Y) / n$ (2.4.a)

dan persamaan normal II menjadi

$$b\sum(X^2) = \sum(XY), \quad \text{atau} \quad b = \sum(XY) / \sum(X^2) \qquad (2.4.b)$$

Aturan membuat $\sum(X) = 0$, waktu dasar variabel X harus diletakkan dalam pertengahan waktu pada periode yang terkandung dalam runtut waktu. Meliputi dua kasus, yaitu :

a. Runtut Waktu yang mempunyai jumlah unit waktu gasal

Prosedur perhitungan trend garis lurus dengan metode kuadrat terkecil untuk jumlah unit waktu gasal dalam runtut waktu adalah sebagai berikut :

1. Buatlah sebuah tabel untuk menghitung nilai $\sum Y$, $\sum X^2$ dan $\sum XY$. Misalkan unit X = 1 tahun dan pertengahan (I Juli) pada pertengahan tahunnya menjadi tahun dasarnya, sehingga jumlah dari nilai X (0, -1, -2, -3, . . ., 1, 2, 3, . .) adalah 0, atau $\sum X = 0$.
2. Dapatkan nilai a dan b dengan mengganti nilai yang dihitung sesuai dengan rumus (2.4) tersebut di atas.
3. Hitunglah tiga diantara nilai trend dengan menggunakan persamaan Yc yang telah diperoleh dan gambarkan hasilnya untuk menarik garis lurus. (Dua titik dipilih untuk menggambar garis lurus, titik ketiga dibuat untuk menguji kebenaran hitungan pada titik-titik tersebut.

b. Perhitungan garis trend garis lurus untuk jumlah tahun yang genap

Prosedur perhitungan trend garis lurus dengan metode kuadrat terkecil hampir sama untuk jumlah tahun yang gasal. Tetapi unit X menjadi 1/2 tahun, karena tahun dasarnya pertengahan pada dua pertengahan tahun runtut waktu, atau 1 Januari pada pertengahan tahun kedua. Nilai X akan mempunyai perbedaan dua angka (-1, -3, -5, . . .,1, 3, 5, . . .) dan jumlah nilai X akan menjadi nol, atau $\sum X = 0$.

2.2.2. Metode Parabola

Bentuk Umum sebuah persamaan polinomial adalah :

$$Yc = a + bX + cX^2 + dX^3 + eX^4 + \ldots$$

Jika persamaan polinomial digunakan untuk menggambarkan gerakan trend non linear, biasanya ditulis dalam bentuk yang lebih sederhana :

$$Yc = a + bX + cX^2 \qquad (2.5)$$

Yang sering disebut persamaan parabola. Untuk mencari nilai dari konstanta a, b dan c diperlukan tiga persamaan normal sebagai berikut :

$$\begin{aligned}
&\text{I.} && n\,a && + b\textstyle\sum(X) && + c\textstyle\sum(X^2) = \textstyle\sum(Y)\\
&\text{II.} && a\textstyle\sum(X) && + b\textstyle\sum(X^2) && + c\textstyle\sum(X^3) = \textstyle\sum(XY)\\
&\text{III.} && a\textstyle\sum(X^2) && + b\textstyle\sum(X^3) && + c\textstyle\sum(X^4) = \textstyle\sum(X^2Y)
\end{aligned} \qquad (2.6)$$

Untuk memperoleh nilai konstanta, biasanya persamaan 2.6 tersebut disederhanakan terlebih dahulu. Penyederhanaan dilakukan dengan membuat jumlah X menjadi nol.

Jika $\sum X = 0$, maka $\sum X^3 = 0$, dan tiga persamaan normal yang ada menjadi :

$$\begin{aligned}
&\text{I.} && n\,a && && + c\textstyle\sum(X^2) = \textstyle\sum(Y)\\
&\text{II.} && && b\textstyle\sum(X^2) && = \textstyle\sum(XY)\\
&\text{III.} && a\textstyle\sum(X^2) && && + c\textstyle\sum(X^4) = \textstyle\sum(X^2Y)
\end{aligned} \qquad (2.7)$$

Sehingga dari persamaan 2.7 diperoleh hasil :

$$\det = n * \textstyle\sum(X^4) - \sum(X^2) * \sum(X^2)$$

$$a = (\textstyle\sum(Y) * \sum(X^4)) - (\sum(X^2) * \sum(X^2Y)) / \det \qquad (2.8.a)$$

$$b = \textstyle\sum(XY)\,\sum(X^2) \qquad (2.8.b)$$

$$c = (n * \textstyle\sum(X^2Y)) - (\sum(Y) * \sum(X^2)) / \det \qquad (2.8.c)$$

Aturan untuk membuat $\sum(X) = 0$ sama dengan aturan yang dipakai pada metode penarikan garis lurus dengan metode kuadrat terkecil. Dengan menggunakan rumus (2.8) tersebut di atas dapat diperoleh persamaan parabola yang dicari.

2.2.3. Metode Eksponensial

Suatu trend eksponensial akan berupa garis lurus pada grafik semilog tetapi berupa kurva pada grafik aritmatik. Persamaan eksponensial yang digunakan dalam menggambarkan trend sekuler ditulis :

$$Yc = a\,b^{x} \qquad (2.9.a)$$

Apabila kita ambil logaritma dari persamaan tersebut kita peroleh :

$$\text{Log } Yc = \text{Log } a + (\log b)\,X \qquad (2.9.b)$$

Untuk memperoleh konstanta log a dan log b yang belum diketahui dalam persamaan garis lurus semilog tersebut dengan metode terkecil diperlukan 2 persamaan normal sebagai berikut :

I. $n(\log a) + (\log b)\sum X = \sum(\log Y)$

II. $(\log a)\sum X + (\log b)\sum X^2 = \sum X(\log Y)$ (Pers. 2.10)

Dengan membuat $\sum(X) = 0$, maka kedua persamaan normal tersebut menjadi :

I. $n(\log a) = \sum(\log Y)$

II. $(\log b)\sum X^2 = \sum X(\log Y)$ (Pers. 2.11)

dan kita memperoleh nilai konstanta log a dan log b dengan rumus sebagai berikut :

$$\log a = (\sum \log Y)/n \quad \text{atau} \quad a = \text{antilog}\,((\sum(\log Y))/n) \qquad (2.12.a)$$

$$\log b = (\sum(X \log Y)) / \sum X^{n}) \qquad (2.12.b)$$

dengan log a : rata-rata logaritma dari Y dan,

log b : slope dari garis pada grafik semilog.

Karena rumus 2.10 diperoleh dengan metode kuadrat terkecil, maka jumlah dari deviasi kuadrat logaritma nilai Y dari nilai trend, atau $\sum (\log Y - \log Yc)^2$ minimum.

(catatan : $\sum (Y - Yc)^2$ dalam hal ini secara umum tidak inimum).

2.2.4. Metode Rata-rata Bergerak

Kalau kita mempunyai data berkala sebanyak n : Y1, Y2, Y3, ... , Yn, maka rata-rata bergerak (moving average) m waktu (tahun,bulan, hari, minggu), merupakan urutan rata-rata hitung sebagai berikut :

(Y1+Y2+ . . . . +Ym)/m , (Y2+Y3+ . . . . +Y(m+1))/m , (Y3+Y4+ . . . . +Y(m+2))/m dan seterusnya.

Sedangkan nilai dari :

(Y1+Y2+ . . . . +Ym), (Y2+Y3+ . . . . +Y(m+1)), (Y3+Y4+ . . . . +Y(m+2)) disebut total bergerak (moving total).

Didalam data berkala, rata-rata bergerak serimg digunakan untuk menghaluskan fluktuasi yang terjadi dalam data tersebut. Proses penghalusan ini disebut " *smoothing of time series*". Apabila rata-rata bergerak dibuat dari data tahunan atau bulanan sebanyak n waktu, maka rata-rata bergerak tahunan atau bulanan berderajat n (*moving average order n)*. Dengan menggunakan rata-rata bergerak untuk mencari trend,maka kita kehilangan beberapa data dibandingkan data asli. Artinya banyak rata-rata bergerak menjadi tidak sama dengan banyaknya data asli. Pada umumnya berkurang sebanyak (n - 1) data.

2.3. Sistem Manajemen Basis Data

Teori dan konsep sangat perlu dipelajari karena akan memberi kerangka kerja untuk memikirkan dan menyederhanankan persoalan. Pada bagian ini akan dibahas beberapa konsep dasar yang ada dalam sistim basis data.

2.3.1 Basis Data

Basis Data adalah suatu kumpulan data terhubung yang disimpan secara bersama-sama pada suatu media, tanpa mengatap satu sama lain atau tidak perlu suatu kerangkapan data dengan cara-cara tertentu sehingga mudah untuk digunakan atau ditampilkan kembali; dapat digunakan oleh satu atau lebih program aplikasi secara optimal; data disimpan tanpa mengalami ketergantungan pada program yang akan menggunakannya;dapat berkembang dengan baik dan memenuhi kebutuhan sistem-sistem baru.

Beberapa kriteria dari basis data antara lain :

1. Bersifat data oriented dan bukan program oriented.
2. Dapat digunakan oleh beberapa program aplikasi tanpa perlu mengubah basis datanya
3. Dapat berkembang dengan mudah, baik volume maupun strukturnya
4. Dapat memenuhi kebutuhan sistem-sistem baru secara mudah dan dapat digunakan dengan cara-cara yang berbeda
5. Kerangkapan data minimal.

2.3.2. Model-Model Data

Model data merupakan suatu cara untuk menjelaskan bagaiman pemakai (user) dapat melihat data secara logik. Beberapa model data yang ada antara lain

a. Object_based data model, merupakan himpunan data dan prosedur/relasi yang menjelaskan hubungan logik antar data dalam suatu basis data berdasarkan obyek datanya. Model yang ada antara lain :

- Model Entity Relationship (ER)
- Model Data Semantic.

b. Record_based data model. Model ini mendasarkan pada record untuk menjelaskan pada user tentang hubungan logik antar data dalam basis data. Model yang ada antara lain :

- Model Relasional
- Model Network (jaringan)
- Model hierarchycal

c. Physical _based data model, digunakan untuk menjelaskan kepadai pemakai tentang bagaimana data-data dalam basis data disimpan dalam media penyimpanan yang digunakan secara fisik.

2.3.3. Relasional Basis Data

Salah satu model data yang ada adalah model data relasional. Model ini menjelaskan kepada pemakai tentang hubungan logik antar data dalam basis data dengan cara memvisualisasikannya ke dalam bentuk tabel-tabel dua dimensi yang terdiri dari sejulah baris dan sejumlah kolom yang menunjukkan atribut-atribut.

Konsep Dasar Model Data Relasional

Istilah		Definisi
Record (tupel)	:	Sebuah baris dalam suatu relasi.
Atribut	:	Suatu kolom dalm suatu relasi.
Domain	:	Batasan-batsan nilai yang diijinkan dalam suatu atribut.
Entity	:	Obyek yang keberadaannya dibedakan dari obyek lain. Suatu entity diwakili dengan suatu himpunan atribut.
Himpunan entity	:	Suatu himpunan yang terdiri dari beberapa entity dengan type yang sama.
Relasi	:	Gabungan dari beberapa entity

Himpunan relasi	:	Himpunan yang terdiri dari beberapa relasi dengan tipe yang sama
Candidat key	:	Atribut atau sekumpulan atribut yang unik yang dapat digunakan untuk mengidentifikasi/membedakan suatu record.
Primary Key	:	Bagian/salah satu dari candidat key yang dipilih dan dipakai untuk membedakan suatu record
Alternat key	:	Candidat key yang tidak dipilih sebagai primary key.

Karakteristik-karakteristik dalam relasi pada model data relasional antara lain :

1. Semua elemen data pada suatu baris dan kolom tertentu harus mempunyai nilai tunggal dan tidak dapat dibagi lagi.
2. Semua data pada kolom tertentu dalam relasi yang sama harus mempunyai jenis yang sama.
3. Masing-masing kolom dalam suatu relasi mempunyai suatu nama yang unik.
4. Pada suatu relasi yang sama tidak ada dua baris yang identik.

Macam-macam Relasi

Terdapat tiga macam relasi diantara himpunan entity A dan entity B

a. Relasi One to one,yaitu : suatu entity di A berassosiasi dengan paling banyak satu entity di B dan sebaliknya.
b. Relasi One to many, yaitu : entity di A berassosiasi dengan sembarang entity di B, sedangkan entity di B hanya dapat berassosiasi dengan paling banyak satu entity di A.
c. Relasi Many to many, yaitu : suatu entity di A dapat berassosiasi dengan sembarang entity di B, demikian juga sebaliknya.

Kunci Record

Kunci record merupakan atribut field yang bersifat unik, sehingga dengan kunci ini dapat dicari/ditemukan satu record tertentu saja. Dengan kunci record ini beberapa file dapat dihubungkan sehingga integritas basis data diwujudkan. Pemasangan kunci record ke file terdiri dari 3 macam, yaitu :

a. Untuk relasi one to one, maka kunci record dapat dipasang pada kedua file yang berelasi.
b. Untuk relasi one to many , maka kunci record dapat dipasang pada file yang banyak (yang menunjuk ke satu).

Untuk relasi many to many, maka perlu dibuat file konektor (file baru) sedemikian sehingga relasi many to many berubah menjadi relasi tidak langsung one to many melalui file konektor. Isi file konektor adalah dua kunci utama, yaitu satu dari file yang pertama dan satu lagi dari file yang lain.

2.3.4. SISTEM BASIS DATA

Sistem Basis Data adalah sekumpulan basis data dengan para pemakai yang menggunakan basis data secara bersama-sama, personal-personal yang merancang dan mengelola basis data, teknik-teknik untuk merancang dan mengelola basis data, serta sistem komputer untuk mendukungnya.

Pengembangan sistem basis data bertujuan untuk mengatasi masalah-masalah yang sering timbul dlam file basis data. Tujuan-tujuan tersebut antara lain :

a) Fleksibilitas data, yaitu untuk memberikan kemudahan dalam menampilkan kembali data-data yang dipilih dan diperlukan dalam basis data. Teknik yang umum diterapkan adalah dengan menggunakan menu-menu untuk mengarahkan pemakai menembus sistem.

b) Integritas data, yaitu sebagai sarana untuk selalu meyakinkan bahwa nilai-nilai data dalam sistem basis data adalah benar, konsisten dan selalu tersedia.

c) Keamanan data, yaitu untuk melindungi data terhadap akses yang kurang legal / tidak berwenang. Fasilitas keamanan data yang lazim digunakan adalah password untuk individu-individu pemakai.

d) Mengurangi / minimalisasi kerangkapan data. Hal ini diperlukan karena kerangkapan data menyebabkan timbulnya beberapa masalah di dalam proses pengaksesan data. Kerangkapan data akan mengakibatkan penggunaan media penyimpanan secara sia-sia, waktu akses yang lebih lama dan akan menimbulkan masalah dalam integritas data.

e) Shareabilitas data, yaitu : sistem basis data yang digunakan harus dapat digunakan oleh pemakai-pemakai yang berbeda atau grup-grup pemakai yang berbeda dapat menggunakan data yang sama dalam basis data.

f) Relatabilitas data, yaitu : kemampuan untuk menetapkan hubungan logik antar tipe-tipe record yang berbeda dalam file-file yang berbeda.

g) Standarisasi data, yaitu untuk menunjukan definisi-definisi rinci data dalam batas presisi yang digunakan pada definisi nama rinci data dan format penyimpanan dalam basis data.

h) Produktivitas Personal. Sistem basis data yang dikembangkan diharapkan dapat meningkatkan produktivitas kerja setiap personal dalam beberapa hal.

2.3.5. NORMALISASI

Untuk bisa memenuhi tujuan tersebut maka perlu dilakukan desain yang baik terhadap struktur data yang ada, dengan salah satu caranya adalah dengan normalisasi. Normalisasi adalah suatu teknik yang menstrukturkan data dalam cara-cara tertentu untuk

membantu mengurangi atau mencegah timbulnya masalah yang berhubungan erat dengan pengolahan data dalam basis data. Proses normalisasi menghasilkan struktur record yang konsisten secara logik, yang mudah untuk dimengerti, dan sederhana dalam pemeliharaannya. Kriteria yang mendefinisikan level-level nirmalisasi adalah bentuk normal. Ada beberapa macam bentuk normal, namun dalam hal ini hanya akan dibahas tiga macam bentuk normal saja, yaitu :

a) Bentuk Normal kesatu (1NF).

Sesuatu relasi berada dalam bentuk normal pertama jika memenuhi kreiteria sebagai berikut : setiap data dibentuk dalam file datar, masing-masing baris (tuple) mempresentasikan satu order, nilai field berupa "atomic value" , tidak ada atribut yang berulang-ulang (ganda), biasanya masih terdapat data redundancy (kerangkapan data).

b) Bentuk Normal Kedua (2NF).

Suatu relasi berada dalam bentuk normal kedua jika relasi tersebut berada dalam bentuk normal kesatu namun sudah tidak terdapat data redundancy dan semua atribut bukan kunci bergantung secara fungsional pada kunci utama.

c) Bentuk Normal Ketiga (3NF)

Suatu relasi berada dalam bentuk normal ketiga jika relasi tersebut sudah berada dalam 2NF dan masing-masing atribut bergantung fungsional secara penuh dan tidak terdapat transitive dependence (bergantung secara transitif).

2.4. Bahasa Pemrograman DBase

2.4.1 Jenis-jenis Data

a) Data Numerik, data jenis ini berfungsi untuk menampung data bilangan yang tersusun angka 0 sampai 9 yang nantinya dapat diproses secara operasi matematik.

b) Data karakter, jenis data ini dapat menampung segala karakter yang bisa dipakai komputer, termasuk berupa angka.

c) Data tanggal, jenis data ini dipergunakan untuk menampung data yang menuruti hukum penanggalan masehi. format default adalah format USA dengan susunan MM/DD/YY.

d) Data Logika, untuk menerima data yang hanya dapat diisikan dengan jawaban ya atau tidak.

e) Data Terstruktur (Array), digunakan utnuk menerima data larik dimensi satu ataupun dua.

2.4.2. Jenis-jenis Operator

a) Operator Matematik, digunakan sebagai lambang untuk melakukan operasi aritmetika dengan perincian sebagai berikut :

Lambang	Fungsi
+	Penjumlahan
-	Pengurangan
*	Perkalian
/	Pembagian
**, ^	Perpangkatan

b) Operator Logika, digunakan untuk menilai apakah nilai suatu variabel sesuai dengan nilai boolean yang diinginkan dengan perincian sebagai berikut :

Lambang	Fungsi
.OR.	Atau
.AND.	Dan
.NOT.	Tidak

c) Operator Pembanding, digunakan untuk menilai apakah suatu variabel memiliki nilai sesuai dengan yang diinginkan, dengan perincian sebagai berikut :

Lambang	Fungsi
>	Lebih besar dari
<	Lebih kecil dari
=	Sama dengan
>=	Lebih besar atau sama dengan
<=	Lebih kecil atau sama dengan
<> ; #	Tidak sama dengan
$	Bagian dari

d) Operator String, dipergunakan untuk operasi dari karakter string, dengan perincian "+" dipakai untuk menggabungkan dua variabel lengkap dengan spasi yang ada dan operator "-" untuk menggabungkan dua variabel dengan menghilagkan spasi yang ada diantaranya.

2.4.3. Jenis-jenis File

Beberapa jenis file yang ada dalam paket DBase antara lain :

a) File Database (Namafile.DBF), file ini berisikan data sesuai dengan struktur yang telah disiapkan terlebih dahulu. Pada file inilah terdapat pusat data yang diperlukan.

b) File indeks (Namafile.NDX), file ini digunakan untuk menampung indeks kunci field yang merupakan kunci untuk pengurutan / pengindeksan dari file data base yang telah ada.

c) File Laporan (Namafile.FRM), file ini berisikan kerangka dari bentuk laporan yang telah dibuat terlebih dahulu. File ini dapat memberikan laporan dalam bentuk tabel.

d) File Format (Nmafile.FMT), file ini berfungsi untuk menampung penataan output di layar maupun di printer.

e) File Program (Namafile.PRG). File Program merupakan jenis file yang sangat penting untuk mengatur segala perintah yang diperlukan dalam masalah penerapan database ini untuk suatu aplikasi.

2.4.4. Pernyataan (Statemen)

Pernyataan-Pernyataan yang ada pada Dbase terdiri dari Pernyataan sederhana dan Pernyataan Terstruktur. Pernyataan sederhana adalah Pernyataan yang tidak berisi Pernyataan yang lain, sedangkan Pernyataan terstruktur merupakan Pernyataan yang tersusun dari sejumlah Pernyataan lain yang akan dieksekusi secara berurutan.

2.4.4.1. Pernyataan Sederhana

a) Pernyataan Masukkan

a.1. Pernyataan Input dan Accept

Pernyataan Input dapat digunakan untuk memasukkan variabel numerik maupun string (karakter) ke dalam suatu variabel memori. Sedangkan Pernyataan Accept akan menganggap semua masukkan sebagai variabel string dan sudah tentu akan disimpan dalam bentuk string. Contoh :

Input " Masukkan Nama Anda :" To Nama

Accept " Masukkan Nama Anda :"To Nama

a.2. Pernyataan Store

Pernyataan Store merupakan salah satu perintah yang dapat dipergunakan untuk memasukkan nilai ke dalam variabel memori. Contoh penggunaan :

Store "Nugroho AH" To Nama

a.3. Pernyataan Wait

Pernyataan Wait berfungsi untuk menerima masukkan dari keyboard tetapi hanya satu karakter saja tanpa harus menekan tombol enter seperti pada perintah input dan Accept, malahan enter itu sendiri merupakan karakter masukkan. Contoh penggunaan:

Wait " Press Any Key to Continue"

b) Pernyataan Keluaran

b.1. Pernyataan "?" dan "??" serta "???"

Pernyataan "?" berfungsi untuk mencetak nilai suatu variabel dengan sebelumnya pindah ke baris berikutnya. Sedangkan dengan menggunakan Pernyataanperintah "??", nilai variabel akan dicetak pada baris yang lama atau tidak pindah baris berikutnya. Pernyataan "???" berfungsi untuk mencetak keluaran langsung ke printer. Contoh penggunaan :

? "Nama anda adalah : ",Nama

c) Pernyataan @ <x,y>

Pernyataan ini dipergunakan untuk mengatur bentuk tanpilan pada layar menurut kehendak pemakai, baik untuk bentuk input atupun output. Bentuk Umum :

@ <baris, kolom> [SAY <ekspresi>][GET <variabel> PICTURE <batasan>]

@ <baris1, kolom1> [CLEAR] TO @ <baris2, kolom2>

@ <baris1, kolom2> [Double]

@ <baris, kolom>

Keterangan :

- Baris dan kolom berupa ekspresi numerik
- Nilai baris dari baris ke 0 sampai baris ke 24, sedangkan nilai kolom dari kolom ke 0 sampai kolom ke 79

2.4.4.2. Pernyataan Terstruktur

2.4.4.2.1. Pernyataan Kendali

Pernyataan kendali digunakan untuk memilih bagian program yang akan dikerjakan sesuai dengan kondisi yang diberikan. Adapun kondisi yang akan diuji dapat berupa kondisi tunggal maupun kondisi ganda. Pernyataan yang termuat di dalam struktur ini dapat berupa pernyataan tunggal maupun pernyataan majemuk.

a. Kondisi Tunggal

Pada struktur kondisi tunggal, kondisi yang diuji hanya satu. Pada pengujian tunggal dapat digunakan Pernyataan **If - Endif.** Pernyataan ini akan menguji kondisi dan menentukan apakah kondisi tersebut benar atau salah, kemudian melakukan instruksi sesuai dengan kondisi tersebut.

Bentuk Umum :

If *kondisi*

Pernyataan/instruksi

Endif

Pernyataan/Instruksi akan dikerjakan apabila *kondisi* terpenuhi, bila *kondisi* tidak terpenuhi maka Pernyataan yang berada di bawah Endif yang akan dikerjakan.

b. Kondisi Ganda

b.1. Pernyataan If - Else - Endif.

Pada struktur program dengan kondisi ganda, dapat digunakan pernyataan **Else** diantara pernyataan **if - Endif** . Pada pernyataan **Else** dapat dimasukkan pernyatan/instruksi yang akan dikerjakan apabila *kondisi* tidak terpenuhi.

Bentuk Umum :

If *Kondisi*

Pernyataan 1

Else

Pernyataan 2

Endif

Apabila *Kondisi* terpenuhi, maka Pernyataan 1 yang akan dikerjakan, sedangkan apabila tidak terpenuhi, maka Pernyataan 2 yang akan dikerjakan.

b.2. Pernyataan Iif ( )

Untuk keperluan sederhan, pemakaian pernyataan kendalai **If - Else - Endif** , dapat digantikan dengan fungsi **iif ( )** (Intermediate If).

Bentuk Umum :

Variabel = **iif** (*kondisi*, JikaBenar, JikaSalah)

Apabila *kondisi* benar, Variabel akan diisi dengan variabel JikaBenar, sedangkan bila kondisi salah, maka variabel akan diisi dengan variabel JikaSalah., misal :

MAKS = **iif** (A> B, A, B)

Catatan : Pemakaian fungsi **iif**() hanya berlaku untuk satu pernyataan setiap kondisinya. Untuk pemakain beberapa pernyataan, pemakaian fungsi **iif**() tidak dapat digunakan.

b.3. Pernyataan Do - Case.

Untuk pernyataan kondisi **If** untuk sejumlah keperluan yang sama/sejajar dapat digunakan pernyataan **Do - Case.**

Bentuk Umum :

Do Case

Case *Kondisi* 1

Pernyataan/Instruksi jika *Kondisi* 1 benar

Case *Kondisi* 2

Pernyataan/instruksi jika *Kondisi* 2 benar

.

.

Case *Kondisi* n

Pernyataan/instruksi jika *Kondisi* n benar

EndCase

Pengerjaan Pernyataan sesuai dengan *kondisi* yang terpenuhi.

2.4.4.2.2. Pernyataan Perulangan

Pernyataan perulangan digunakan untuk melakukan proses berulang terhadap suatu pernyataan/instruksi yang ditentukan. Banyaknya perulangan tergantung dari terpenuhinya kondisi tertentu.

a. Struktur Ulang Do - While

Struktur ulang **Do - While** merupakan struktur program pertama yang disediakan sistem dBase pada versi terdahulunya. Struktur program ini akan mengerjakan instruksi /pernyataan untuknya jika kondisi yang diujinya benar. Pengujian kondisi dilakukan sebelum pernyataan/instruksi untuknya dieksekusi. Pernyataan/instruksi pada struktur perulangan dapat berupa satu baris instruksi (pernyataan tunggal) atau sejumlah tertentu baris pernyataan (pernyataan majemuk). Di dalamnya juga dapat diletakkan struktur program lainnya.

Bentuk Umum:

DO WHILE *Kondisi*

Pernyataan/instruksi selama *kondisi* benar

ENDDO

Bila *kondisi* dipenuhi, maka pernyataan/instruksi akan dikerjakan terus menerus sampai *kondisi* tidak terpenuhi. Begitu *kondisi* tidak terpenuhi, maka perintah yang dilaksanakan selanjutnya adalah perintah yang ada di bawah **ENDDO**.

b. Pernyataan Do - Until

Struktur perulangan **Do - Until** merupakan struktur program yang disediakan sistem dBase untuk mendukung teknik pemrogram secara terstruktur. Pernyataan **Do** dipakai sebagai awal pernyataan dan **Until** dipakai sebagai akhir pernyataan yang akan dikerjakan berulang sekaligus untuk menguji apakah proses berulang tersebut masih dapat diteruskan atau sudah berakhir. Dalam struktur **Do - Until** pernyataan akan diulang terus menerus sampai kondisi terpenuhi. Bentuk Umum :

Do

Pernyataan/instruksi

Until *Kondisi*

c. Struktur Ulang For - Next.

Untuk perulangan proses berdasarkan suatu hitungan, disediakan struktur program **For - Next**. Bentuk Umum :

For Var=Awal **to** Akhit **Step** Penambahan

Pernyataan / Instruksi

Next

Pengerjaan pernyataan/instruksi dilakukan mulai dari nilai variabel Var = Awal sampai nilai variabel Var = Akhir dengan penambahan PENAMBAHAN.

d. Struktur Ulang Scan - EndScan

Struktur Ulang **Scan -EndSacan** hanya dapat dipakai untuk proses pada file tabel saja. Sedangkan untuk bentuk pemrograman lain tidak bisa. Bentuk Umum :

Scan [Jangkauan] [**For** *kondisi* 1] [**While** *kondisi* 2]

Pernyataan / instruksi yang akan dikerjakan

Endscan

Catatan :

- jangkauan *Jangkauan* tersebut dapat berupa ruang lingkup dari record file tabel yang akan diproses. Jangkauan tersebut dapat diisi **all** (untuk semua record dari file tabel), **rest** (untuk sisa record dari posisi pointer record yang aktif), **next** n (untuk n record dari posisi pointer record yang aktif), atau **record** n (untuk record ke n).
- Pernyataan tambahan **for** dan/atau **while** ditujukan untuk meminta sistem melakukan pencarian record yang dimaksud pada jangkauannya berdasarkan kondisi yang ditentukan.

Pada pernyataan perulangan dapat digunakan beberapa interupsi berupa fungsi **Exit** dan **Loop**. Fungsi dari pernyataan tersebut adalah :

- **Loop**, digunakan untuk mengulang pernyataan /instruksi diatasnya dimulai dari baris pernyataan dibawah pernyataan Do sebelum semua proses berulang selesai dilakukan..
- **Exit**, Untuk bisa keluar dari proses berulang sebelum syarat berhenti terpenuhi. Pengerjaan berikutnya adalah baris pernyataan dibawah Pernyataan Enddo.

2.4.4.3. Pernyataan Pengurutan Data

a. Pernyataan Sorting

Pernyataan Sorting digunakan untuk mengurutkan data secara fisik, susunan nomor record yang lama akan diurutkan dan dituliskan dengan nomor record baru pada file database yang baru pula. Jadi hasil dari proses ini adalah suatu database baru, yang dengan sendirinya mempunyai ekstensi DBF, dan file database yang diurutkan tidak hilang atau diubah. Bentuk Umum :

SORT ON FIELDNAME TO NEWFILE

Dengan Fieldname adalah nama field yang akan dijadikan kunci pengurutan, dan Nemfile adalah nama file baru yang terbentuk.

b. Pernyataan Index

Selain menggunakan Pernyataan sort, proses pengurutan data juga dilakukan dengan Pernyataan Index, atau data berindeks. Kalau pengurutan data dengan menggunakan Sort menghasilkan data urut secara fisik, dalam penggunaan Pernyataan Index , maka nomor record tetap seperti semula. Pengurutan data dimasukkan dalam file index dengan ekstensi .NDX, yang tidak dapat berdiri sendiri namun harus disertai dengan file induknya.

Bentuk Umum :

INDEX ON FIELDNAME TO FILENDX

Dengan FieldName adalah nama field yang menjadi kunci pengurutan dan FileNdx adalah file hasilpengindeksan.

2.4.4.3.2. Pernyataan Pencarian Data

a. Pernyataan Locate

Pernyataan **Locate** digunakan untuk melakukan pencarian data dengan segala kondisi terhadap file database. Kelemahan dari Pernyataan ini adalah waktu yang diperlukan untuk mencari data relatif lebih lama dibanding Pernyataan pencarian data lain. Apabila data yang memenuhi kondisi cukup banyak, Pernyataan **Locate** akan langsung menemukan data pertama yang sesuai dengan kondisi. Untuk melanjutkan pencarian data lain yang masih memenuhi kondisi yang sama digunakan Pernyataan **CONTINUE**.

Bentuk Umum :

Locate For *Kondisi*

Dengan *Kondisi* adalah kondisi data yang akan dicari.

b. Pernyataan Find dan Seek

Pada database yang telah diindeks, dapat dipergunakan cara pencarian data yang lebih cepat, yaitu dengan Pernyataan **Find** dan **Seek**. Kedua perintah tersebut hanya dapat dipakai pada database yang telah diindeks saja. Kelemahan dari kedua Pernyataan ini adalah tidak dapat mencari data dengan kondisi yang sama, sebab data yang dicari adalah data yang pertama kali ditemukan. Pencarian dilakukan terhadap field yang menjadi kunci indeks.

Bentuk Umum Pernyataan **Find** : **Find** &NamaVariabel

Contoh :

```
Use Siswa Index Xsiswa
Nama="Sri"
Find &Nama
atau
Find "Sri"
```

Untuk mencari data yang mempunyai spesifikasi sesuai dengan suatu nilai dari suatu variabel, maka diperlukan suatu jenis operator macro (&). Operator jenis ini mewakili variabel string yang dikandung dalam suatu variabel.

Bentuk Umum Pernyataan **Seek** : **Seek** NamaVariabel

Dalam pemakaian Pernyataan **Seek**, data yang akan dicari harus terlebih dahulu disimpan dalam suatu variabel memori, baru kemudian dicari dengan menggunakan variabel tersebut. Penggunaan Pernyataan **Seek** tidak menggunakan operator & di depan variabel yang akan dicari. Contoh penggunaan :

Store "Sri" To Nama

Seek Nama

2.4.4.4. Pernyataan Untuk Merelasikan Dua File DataBase.

Hubungan antara satu file database dengan file database yang lain dapat digunakan dengan memakai Pernyataan **Set Relation**. Untuk menghubungkan satu file database dengan beberapa file database lain dapat dilakukan hanya dengan menambahkab Pernyataan **Additive** pada akhir Pernyataan **Set Relation**. Database yang akan direlasikan harus sudah diindeks menurut kuncinya sendiri-sendiri. Dua database dapat direlasikan jika dalam kedua database tersebut terdapat field yang memiliki nama, tipe data dan ukuran data yang sama.

Bentuk Umum :

Set Relation To *FieldName* **Into** *FileName*

Denagn *FieldName* adalah Nama Field kunci yang akan direlasikan dengan field yang ada pada File indeks *FileName*.

2.4.5. Fungsi dan Procedure

Fungsi digunakan untuk memproses suatu data agar menghasilkan informasi yang diinginkan. Pada sistem dbase disediakan beberapa fungsi internal yang siap pakai. Namun apabila fungsi yang disediakan tidak mencukupi, maka dapat pula dibentuk fungsi sendiri yang biasa dikenal dengan istilah UDFs (User Defined Functions). Untuk mendefinisikan suatu fungsi buatan digunakan pernyataan **procedure.** Bentuk Umum :

Format 1 :

Procedure *namaprosedur* [(parameter)]

Pernyataan/instruksi pelaksana fungsi/prosedur

Return [variabelhasil]

Format 2 :

Procedure *namaprosedur*

Parameter *parameter*

pernyataan/instruksi pelaksana fungsi

Return [variabelhasil]

Catatan:

- Variabel *parameter* pada format pertama bersifat **local** (yang bersifat local pada setiap prosedur), sedangkan variabel *parameter* pada format kedua bersifat **private** (yang akan terhapus isi variabel sebelumnya apabila terdapat nama variabel yang sama pada prosedur).
- Pemakaian tanda [ ] pada format pernyataan sebagai tanda bahwa bagian tersebut bersifat optional (dapat dipakai atau tidak dipakai sesuai dengan kebutuhan).
- Parameter digunakan untuk mengirim suatu nilai sebagai variabel referensi atau variabel nilai. Variabel referensi dapat dipakai untuk mengembalikan nilai hasil proses dari suatu prosedur.

Sedangkan variabel nilai tidak mengembalikan hasil proses dari suatu prosedur. Pada variabel yang dipakai sebagai referensi, eksekusinya dilakukan dengan pernyataan **with** serta variabel parameter. Sedangkan pada variabel nilai , eksekusinya dilakukan dengan pernyataan **with** beserta variabel parameternya dalam tanda kurung.

- Untuk melakukan eksekusi suatu prosedur digunakan pernyataan **Do** diikuti nama prosedurnya.

Pemberian nama dari suatu prosedur harus unik, yang tidak boleh sama dengan nama program lain. Prosedur-prosedur biasanya disertakan pada akhir file-file program, tetapi dapat pula dicantumkan dalam sebuah file program tunggal yang mencakup sederet prosedur-prosedur. Apabila prosedur-prosedur dikumpulkan menjadi sebuah file program tunggal, maka perintah **Set Procedure To** *NamafileProsedur* diberikan pada file program yang akan mempergunakannya.

2.5. Bahasa Pemrograman Pascal

2.5.1. Pengolahan Data Dengan File

Untuk keperluan yang lebih luas, maka bahasa pemrograman Pascal juga memberikan fasilitas untuk dapat melakukan pengolahan data dengan file. Pengolahan data dengan file diperlukan agar pengetikan pemasukkan data tidak dilakukan berulang kali, namun data tersebut dapat disimpan dalam media penyimpan. Setiap file tersusun dari satu atau beberapa item data yang disebut komponen. Setiap komponen memiliki nomor komponen, yang menyatakan urutan komponen dalam file. Dalam Turbo Pascal, komponen pertama mempunyai nomor komponen sama dengan 0, komponen berikutnya bernomor 1 dan seterusnya. Untuk keperluan pengaksesan komponen dari file, file dilengkapi dengan pointer file. Pointer file menunjuk ke komponen yang akan diakses. apabila pointer file sedang menunjuk ke suatu komponen, kemudian terjadi operasi pengaksesan, dengan sendirinya pointer file akan bergeser ke lokasi sesudah komponen yang telah ditunjuk. Pada Turbo Pascal, file dibagi dalam 3 golongan, yaitu File bertipe, File teks dan File tak

bertipe. Sesuai dengan penggunaan dalam tugas akhir ini, maka dalam hal ini hanya akan dibahas tentang File Teks saja.

2.5.1.1.File Teks

File teks merupakan file yang hanya bisa diakses secara sekuensial saja. Pengaksesan secara acak tidak dimungkinkan. Pada file teks, file tersusun atas sejumlah baris, dengan setiap baris diakhiri dengan suatu tanda akhir baris (disebut *eoln* atau end of line). File teks didefinisikan dengan menggunakan kata tercadang TEXT, dengan bentuk umum :

Variabel_file : TEXT

Pengolahan data dengan file mempunyai urutan proses sebagai berikut :

1. Menyebutkan/mengaitkan variabel file dengan nama file
2. Membuka file
3. Melakukan pengaksesan file
4. Menutup file

2.5.1.2. Beberapa Prosedur dan Fungsi Pustaka Untuk Pengolahan File

a. Assign

Sebelum melakukan pengolahan data dengan file, variabel file haruslah terlebih dahulu dikaitkan dengan nama file. Hal ini dilakukan dengan prosedur pustaka Assign.

Bentuk umum : **Assign** (*Bks, Nama*);

Catatan : Parameter *Bks* adalah nama perubah berkas yang telah dideklarasikan pada bagian type atau var dan bisa berupa sembarang tipe berkas. Parameter *nama* menunjukkan nama sebenarnya yang ada dalam media penyimpan.

Contoh : **Assign** (F, 'C:\DBASEWIN\SAMPLES\DATAGRAF.TXT);

b. Reset

Fungsi : Untuk membuka berkas yang ditunjuk oleh prosedur **assign**.

Bentuk umum : **Reset** (*Bks* [,Ukuran]);

Catatan : *Bks* adalah berkas yang telah ditunjuk. Ukuran adalah ungkapan bertipe word yang bisa digunakan hanya jika *bks* adalah file tak bertipe.

Contoh : **reset** (F);

c. Read

Fungsi : Untuk membaca satu atau lebih nilai data dari berkas text ke ke dalam satu perubah atau sejumlah perubah.

Bentuk umum : **Read** ([*Bks*,] P1 [,P2,P3,...,Pn]);

Catatan : *Bks*, jika ditentukan, adalah berkas bertipe text. Jika tidak ditentukan dianggap sebagai perubah berkas standar (input). Parameter berikutnya merupakan variabel yang disiapkan untuk menampung nilai dari isi berkas. Apabial dalam proses pengaksesan eof(*Bks*) atau eoln(*Bks*) bernilai true, **read** berikutnya akan dimulai dari tanda akhir baris yang mengakhirinya (pindah ke komponen berikutnya). Prosedur **Read**, setelah melakukan pembacaan tidak akan menyebabkan pindah baris.

d. Readln

Fungsi : Untuk mengerjakan prosedur read dan kemudian pindah ke baris berikutnya.

Bentuk umum : Readln ([Bks,] P1 [,P2, P3, ..., Pn]);

Catatan : Prosedur ini merupakan perluasan dari prosedure **read**. Setelah prosedur **read** dilaksanakan, maka dengan sendiringa pointer akan pindah ke baris berikutnya.

e. Close

Fungsi : Untuk menutup berkas yang digunakan.

Bentuk umum : **Close** (*Bks*);

```
Contoh :    Var   Bks  : Text;
                  Kata : String;
            Begin
                  Assign(Bks,'C:\DATA_MHS.DAT');
                  Reset(Bks);
                  Readln(Bks, Kata);
                  Close (Bks);
            end.
```

2.5.2. Pemrograman Grafik Dengan Turbo Pascal

Untuk memanfaatkan fasilitas grafik, haruslah tersedia file sebagai berikut :

- GRAPH.TPU { berupa unit grafik}
- *BGI { pengendali grafik sesuai dengan adpater komputer}
- *.CHR { file yang berisi data jenis huruf}

Mode grafik pada komputer memiliki sifat yang berbeda dengan mode teks. Pada mode grafik, elemen terkecil yang menyusun tampilan pada layar bukan lagi berupa karakter, melainkan berupa titik (biasa disebut piksel). Banyaknya piksel dala layar menentukan halus tidaknya citra gambar yang ditampilkan. Istilah yang menyatakan banyaknya piksel dalam layar adalah resolusi. Tinggi rendahnya resolusi bergantung dari jenis adapter komputer yang digunakan. Dalam menggunakan fasilitas grafik, diperlukan langkah-langkah sebagai berikut :

1. Membuka mode grafik
2. Menggunakan fasilitas yang ada pada mode grafik sesuai kebutuhan

3. Menutup mode grafik kembali ke mode teks.

2.5.2.1. Prosedur dan Fungsi Pustaka Penting Berkaitan Dengan Mode Grafik

a. Initgraph

Fungsi : Untuk melakukan inisialisasi terhadap mode grafik / membuka mode grafik.

Bentuk Umum : **InitGraph** (var *kendaligraf*, *modegrafik* : integer;
Pathkendali : string)

Pathkendali menyatakan letak dari file pengendali grafik (.BGI). Jika file pengendali grafik berada dalam direktori yang aktif, cukup diisi dengan string kosong. Parameter *kendaligarf* menunjukan kode pengendali grafik yang sesuai denagn adapter. Parameter *modegrafik* menentukan mode grafik yang dipilih sesuai dengan adapter dan keinginan user. Apabila *kendaligrafik* diisi dengan Detect, maka proses pemilihan pengendali grafik maupun mode grafik akan dillakukan secara otomatis.

b. GraphResult

Fungsi : Untuk memperoleh kode kesalahan terhadap operasi grafik.

c. Closegraph

Fungsi : Untuk menutup mode grafik dan kembali ke mode yang aktif sebelum mode grafik diaktifkan.

d. PutPixel (x,y :integer; warna :word)

Fungsi : Untuk membuat titik (piksel) pada koordinat (x,y) atau baris x, kolom y dengan warna piksel sesuai parameter warna.

e. Moveto(x,y :integer)

Fungsi : Untuk memindahkan penunjuk gambar (pointer) ke koordinat (x,y).

f. Moverel (Dx,Dy :integer)

Fungsi : Untuk memindahkan posisi pointer pada posisi dengan jarak Dx piksel ke kanan dan Dy piksel ke bawah.

f. Line(x1,y1,x2,y2 : integer)

Fungsi : Untuk membuat garis dari posisi (x1,y1) sampai posisi (x2,y2)

g. Lineto(x,y : integer)

Fungsi : Untuk membuat garis dari posisi pointer sampai posisi (x,y)

h. LineRel(Dx,Dy :integer)

Fungsi : Untuk membuat garis mulai letak pointer sampai posisi dengan jarak Dx piksel ke kanan dan Dy piksel ke bawah.

i. OutTextXY (x,y :integer ; Teks :string)

Fungsi : Untuk menampilakn teks pada posisi (x,y).