

BAB VII

INFORMIX-4GL

Pada Informix-4GL terdapat dua bagian, bahasa pemrograman dan lingkungan pendukung pemrograman. Bahasa pemrograman berfungsi menerjemahkan instruksi tingkat tinggi yang dapat dimengerti oleh programmer ke dalam perintah tingkat rendah yang dapat dilakukan komputer. Lingkungan pendukung pemrograman berfungsi membantu dalam penyusunan program 4GL.

BAHASA PEMROGRAMAN 4GL

Bahasa pemrograman 4GL berfungsi menerima file yang berisi urutan-urutan statement program dan menghasilkan program yang dapat dijalankan. Misalnya untuk membuat basisdata untuk pertama kali, kita dapat menulis program yang sangat sederhana seperti berikut

```
main  
create database music  
end main
```

Informix 4GL akan menerjemahkan ketiga statement di atas ke dalam operasi yang diperlukan untuk menghasilkan basisdata kosong yang bernama music.

Sebuah file yang bernama music.4gl dapat menyimpan ketiga perintah ini. Compiler Informix 4GL dapat mengambil file tersebut yang berisikan perintah dan kemudian menghasilkan suatu program yang dapat dijalankan yang disebut **music.4ge**. Dengan menjalankan program music .4ge akan menciptakan basisdata musik. Bagian berikut menunjukkan rincian dalam melakukan tahap-tahap tersebut.

LINGKUNGAN PENDUKUNG 4GL

Pada 4GL terdapat bahasa pemrograman dan lingkungan pendukung pemrograman. Kita dapat menulis program 4GL dengan editor teks yang kita pilih dan mengcompilanya dengan bahasa pemrograman.

Dengan adanya lingkungan 4GL tersebut maka lebih mudah bagi kita untuk menulis, mengkompile, menjalankan, dan membetulkan program. Kita dapat memilih perintah-perintah editing, compiling dan modifikasi dari menu yang terdapat pada lingkungan pemrograman.

INFORMIX-4GL DAN SQL

SQL merupakan akronim dari Structured Query Language. SQL adalah bahasa standard industri yang digunakan untuk mengolah dan mengakses basisdata. Lingkungan 4GL dapat secara langsung mengakses bahasa SQL dengan sebuah pilihan menu.

4GL juga menyajikan semua kemampuan SQL dalam bahasa 4GL. Operasi bahasa 4GL memiliki sintak yang berbeda dari SQL.

OPERASI BAHASA 4GL

Penjelasan berikut menunjukkan operasi-operasi yang ada bagi programmer 4GL. Informix-4GL memberikan perintah-

perintah untuk membuat, mengubah, dan mengakses basisdata. Pada Infomix 4GL terdapat tampilan layar window, menu, form, atau data dan perintah-perintah untuk membentuk dan menampilkan data dalam bentuk laporan yang tercetak.

4GL juga memberikan fungsi-fungsi yang bermanfaat lainnya. Misalnya pelayanan sistem akses perintah 4GL seperti waktu dan tanggal. 4GL juga menyediakan perintah-perintah untuk menjaga keamanan dan integritas data. Perintah-perintah tersebut meliputi perintah menjaga catatan transaksi dan audit.

Kita dapat memulai dengan menuliskan program 4GL yang berfungsi membuat basisdata. Kemudian kita dapat menulis program lain yang berfungsi meletakkan bentuk pada layar. Data dapat masuk ke dalam basisdata melalui bentuk ini. Kita juga dapat membuat program lain yang berfungsi untuk memilih data dari basisdata dan menuliskan laporan.

ORGANISASI PROGRAM

Program 4GL berisi urutan-urutan statement 4GL. Masing-masing pernyataan tersebut memiliki format khusus dan bertugas melaksanakan fungsi tertentu. Terdapat berbagai statement untuk melaksanakan operasi 4GL yang dijelaskan di atas. Berikut adalah contoh program pendek.

```
main
  define a_var integer
  let a_var = 2
  display "Hello, the variable = ", a_var
end main
```

Baris pertama berfungsi untuk memberitahukan kepada compiler 4GL bahwa modul utama dimulai. Baris kedua mendefinisikan **a_var integer** dan membuat variabel yang bernama **a_var**. **An_integer** adalah nama tempat untuk data tersebut, dalam hal ini nilai integer dapat disimpan. Baris ketiga menyimpan nilai 2 ke dalam variabel **a_var**. Baris keempat berfungsi menampilkan pesan **Hello, variabel =** dan nilai yang disimpan adalah integer 2. Baris terakhir berfungsi memberitahukan compiler 4GL bahwa program berakhir.

STATEMENT FORMAT

Statement 4GL memiliki bentuk bebas. Ruang dan baris yang kosong biasanya tidak ada pengaruhnya. Sebagai contoh, pernyataan-pernyataan berikut diperlakukan sama.

```
define an_integer
define
an_integer
```

COMMENT

Comment hendaknya selalu dipakai untuk menjelaskan apa saja yang kita anggap tidak jelas dalam program. Comment dapat muncul dimana pun dalam program 4GL. Comment dimulai dengan tanda # dan diakhiri pada satu baris yang sama. Berikut adalah contoh-contoh comment.

```
main # A comment can come anywhere
# after the pound sign.
# use comments wherever a program
```

```
# explanation is needed
define ax integer # A description could go here
end main
```

MODUL PROGRAM

Statement program dapat dikelompokkan ke dalam modul-modul. Program utama harus ada. Juga bisa terdapat fungsi dan report. Berikut adalah program 4GL dalam bentuk outline.

```
global data
main routine
one or more functions
one or more reports
```

Berikut adalah program 4GL yang pendek yang tidak berfungsi melakukan apapun. Program ini juga tidak dapat dicompile; program ini diberikan hanya untuk menunjukkan masing-masing jenis modul program.

```
main
  call function one ( )
  call function two ( )
  output to report one ( )
  output to report two ( )
end main
#
function one ( )
end function one
#
```

```
function two ( )
end function two
#
report one ( )
end report
#
report two ( )
end report
```

Seperti modul-modul lainnya, modul utama berisikan pernyataan 4GL. Modul utama harus ada. Statement 4GL yang terdapat pada modul utama mencakup statement yang dapat memanggil satu modul lain atau lebih.

Sebuah fungsi merupakan suatu urutan statement 4GL yang ditulis untuk melakukan kegiatan khusus. Statement ini dikombinasikan ke dalam suatu unit yang berbeda. Dalam sebuah program mungkin terdapat satu fungsi atau lebih, bukan suatu keharusan. Tidak harus terdapat beberapa fungsi dalam suatu program. Fungsi tidak boleh memasukkan fungsi lain. Dengan fungsi maka lebih mudah untuk menulis dan memperbaiki program. Dengan fungsi pula struktur program dapat menunjukkan kegunaan program. Fungsi yang ditulis untuk tujuan umum dapat digunakan dalam banyak tempat. Segmen program dapat digunakan kembali tanpa harus menulis ulang.

Report merupakan jenis fungsi khusus yang digunakan untuk mencetak data dalam format yang telah ditentukan. Report tidak boleh berisikan report atau fungsi lain.

Modul utama dapat memerintahkan fungsi dan report untuk melakukan pekerjaan. Control dapat kembali lagi ke program utama apabila report atau fungsi telah selesai. Fungsi dapat

memerintahkan fungsi lain untuk melakukan tugasnya. Fungsi dapat memulai report, sedangkan rutin report dapat memanggil fungsi.

Apabila telah diperintahkan, maka data dapat masuk ke dalam fungsi atau report. Apabila telah selesai data dapat dikembalikan dari sebuah fungsi.

KOMPONEN PROGRAM 4GL

Bagian berikut menjelaskan komponen bahasa pemrograman 4GL. Dan bab berikutnya berfungsi menjelaskan masing-masing fasilitas bahasa tersebut secara rinci.

DATA

Informix-4GL dapat berfungsi untuk menggabungkan dan membandingkan angka, karakter dan nilai Boolean (benar atau salah).

Angka

4GL dapat melakukan operasi matematika dengan angka meliputi penambahan, pengurangan, perkalian dan pembagian.

KARAKTER

String adalah sekelompok karakter, misalnya "abcd" atau "Ini adalah string." 4GL dapat menggabungkan karakter ke dalam string. Sebuah karakter dapat dibandingkan dengan karakter lainnya. String karakter dapat juga digabungkan dengan string karakter lainnya. Bagian dari suatu karakter string dapat dipilih.

Nilai (logikal) Boolean

Nilai Boolean adalah benar atau salah.

VARIABEL

Variabel program berfungsi menyimpan data yang sering kali berubah. Variabel dapat berisi angka atau karakter atau nilai Boolean.

```
a = 1  
b = "x"  
c = true
```

Variabel dapat menyimpan hasil dari pernyataan matematik, karakter atau Boolean.

```
a = b + c  
b = "c", "d"  
c = true
```

CAKUPAN DATA

Sebuah variabel dapat didefinisikan dalam suatu modul

```
main  
  define var_1 integer  
  call demo ( )
```

```
end main
function demo ( )
    define var_2 integer
end demo
```

Variabel hanya dapat diakses dari dalam modul tempat variabel tersebut didefinisikan. Dalam contoh ini, tiap statement 4GL dalam modul utama dapat menggunakan variabel var_1. Dalam contoh ini setiap statement dalam fungsi tersebut juga dapat mengakses data yang terdapat dalam variabel var_2. Sebagai contoh, program berikut memberikan nilai untuk masing-masing variabel.

```
main
    define var_1 integer
    let var_1 = 2
    call demo ( )
end main
function demo ( )
    define var_2 integer
    let var_2 = 2
end function
```

Inilah yang disebut dengan cakupan kontrol. cakupan kontrol dari variabel var_2 adalah fungsi demo. Cakupan kontrol dari variabel var_1 adalah modul utama. Sebagai contoh program berikut yang tidak dapat dicompile dan di-run.

```
main
    define var_1 integer
```

```
call demo ( )
end main
function demo ( )
  define var_2 integer
  let var_2 = 2
  let var_1 = 2
end function
```

Program ini tidak dapat berjalan karena fungsi demo tidak mengetahui tentang variabel var_1.

Tiap modul dapat berisi data yang hanya diakses oleh modul itu sendiri. Data yang hanya dapat diakses oleh sebuah modul disebut sebagai data lokal.

Daerah data global memuat data yang dapat digunakan oleh tiap modul dalam sebuah program. Sebuah program tidak memerlukan bagian data global. Berikut adalah contoh program dengan daerah data global.

```
globals
define
  var_1, var_2 integer,
end globals
main
  let Var_1 = 2
  call demo ( )
end main
function demo ( )
  let var_2 = 2
end function
```

Masing-masing modul dalam program ini mengetahui tentang variabel `var_1` dan `var_2`. Hal ini karena kedua variabel tersebut global bagi kedua rutin ini. Fungsi atau program lain yang terdapat dalam program ini akan mengetahui kedua variabel global tersebut.

OPERASI YANG TELAH DITENTUKAN SEBELUMNYA

Perintah yang ditentukan sebelumnya mencakup pengolahan string. Perintah ini juga mengakses waktu dalam hari. Sebagai contoh, tanggal hari ini terdapat pada operasi yang telah dibuat dari pabrik

```
let a_var = today
```

Fasilitas yang ditentukan sebelumnya berfungsi mengolah file help serta berfungsi untuk memberikan tanggapan terhadap situasi yang salah. Ini hanyalah salah satu dari beberapa variasi operasi yang ditentukan sebelumnya.

ALUR KONTROL

Sebuah operasi dapat dipilih dari beberapa operasi yang mungkin dalam sebuah program 4GL. Operasi atau sejumlah operasi dapat diulang-ulang beberapa kali. Statement yang berfungsi mengontrol alterasi, dan repetisi atau berfungsi untuk memanggil report atau fungsi akan menentukan alur kontrol suatu program. Sebagai contoh instruksi berikut akan melakukan sesuatu apabila nilai Boolean yang dikandung dalam variabel adalah betul (`true`), dan instruksi tersebut akan melakukan hal yang lain apabila nilai Boolean tersebut adalah salah (`false`).

```
if a=true then
  do this
else
  do this instead
end if
```

INTERAKSI LAYAR

Terdapat variasi statement 4GL yang berfungsi untuk menampilkan dan memasukkan data ke dalam layar. Perintah-perintah yang ada berfungsi menampilkan dan mengolah bentuk serta menu.

REPORT GENERATION

Pada 4GL terdapat statement yang berfungsi untuk memformat data dalam bentuk report yang tercetak. Sebuah program dapat mengumpulkan data dari basisdata atau dari pemakai serta dapat memerintahkan report untuk memformat data yang akan disajikan.

PENANGANAN KESALAHAN DAN KELAINAN

Apabila kesalahan atau kelainan terdeteksi, maka 4GL dapat mengontrol alur kontrol program. Statement memberikan kemampuan untuk menangkap kesalahan yang mungkin dapat menyebabkan program mengalami kesalahan. Apabila terjadi kesalahan, tindakan yang tepat akan dilakukan, tidak menghentikannya sama sekali.