# PERNYATAAN DASAR COBOL

**OBJEKTIF :**

1. Mahasiswa Mengetahui Pengetahuan Pernyataan Dasar Bahasa COBOL.
2. Mahasiswa Mampu Membuat dan Menjalankan Pernyataan Dasar Bahasa COBOL.
3. Mahasiswa Mampu Memahami Struktur Pernyataan Dasar Bahasa COBOL.

---

## 5.1 PERNYATAAN INPUT/OUTPUT

Pernyataan input dan ouput digunakan untuk mendapatkan data dari user dan menampilkan output dari program cobol. Pada cobol terdapat ACCEPT dan DISPLAY verbs yang digunakan untuk read dan write dari keyboard ke layar. Tujuan sebenarnya dari penggunaan commands ini bukan untuk berkomunikasi dengan end user tetapi digunakan dalam batch-programming environment untuk memungkinkan terjadinya interaksi dengan computer operator.

**ACCEPT VERB**

Terdapat dua format dalam pernyataan Accept yaitu :

- Mendapatkan data dari keyboard atau *peripheral device* dari user.
- Memungkinkan untuk mengakses tanggal dan waktu dari system.

Pada umumnya data tersebut dalam jumlah relatif kecil dan tidak didefinisikan sebagai sebuah file.

Berikut bentuk umum penggunaan pernyataan Accept :

```
ACCEPT nama-data.
```

Saat mendapatkan data dari sistem operasi sertakan optional FROM, seperti contoh berikut :

```
ACCEPT DATA-NILAI.
ACCEPT TANGGAL FROM DATE.
```

**DISPLAY VERB**

Display statement digunakan untuk menampilkan output dari program COBOL.

Contoh :

```
DISPLAY NAMA_MHS.
DISPLAY "Tanggal Lahir  : " TANGGAL.
DISPLAY “LAPORAN”
```

Pada contoh diatas pernyataan Display dapat digunakan untuk menampilkan nilai dari suatu data name seperti data name NAMA_MHS dan TANGGAL, dan juga dapat langsung menampilkan suatu pernyataan atau paragraf dengan meletakkannya diantara tanda dua kutip seperti kata LAPORAN.

## 5.2 PERNYATAAN INITIALIZE

Penyataan ini digunakan untuk menginisialisasikan group item atau elementary item. Data name yang disertai dengan RENAME clause tidak dapat diinisialisasi. Dengan menggunakan statement ini anda dapat menginisialisasi banyak jenis data item seperti alfanumerik, alfabetik, dan numeric-edited. Data numerik akan diganti dengan ZEROES. Sedangkan alfanumerik dan alfabetik akan diganti dengan SPACES (blank). Sebuah statement initialize secara fungsional setara dengan satu atau lebih penggunaan statement MOVE.

Contoh penggunaan pernyataan initialize :

- Initializing data alfanumerik :

```
01  HURUF-1    PIC X    VALUE "y".
01  HURUF-3    PIC X(1) VALUE "A".
. . .
   INITIALIZE HURUF-1
      REPLACING ALPHANUMERIC DATA BY HURUF-3
```

Pada contoh pernyataan diatas dilakukan initialize pada data name HURUF-1 berdasarkan data name HURUF-3 sehingga terjadi perubahan nilai. Perubahan nilai pada data name tersebut seperti pada tabel berikut :

| HURUF-3 | HURUF-1 sebelum initialize | HURUF-1 setelah initialize |
|---|---|---|
| A | y | A |

- Initializing data numerik :

```
01  ANGKA-1       PIC 9(8)     VALUE 98765432.
01  ANGKA-3       PIC 9(7)       VALUE 1234567.
. . .
   INITIALIZE ANGKA-1
      REPLACING NUMERIC DATA BY ANGKA-3
```

Pada contoh pernyataan diatas dilakukan initialize terhadap data berjenis numerik, dimana data name ANGKA-1 diinitialize dengan data name ANGKA-3, sehingga nilai yang ada pada data name tersebut akan berubah menjadi :

| ANGKA-3 | ANGKA-1 sebelum diinitialize | ANGKA-1 setelah diinitialize |
|---|---|---|
| 1234567 | 98765432l | 01234567 |

Dengan disertakannya clause REPLACING maka data item bisa diinisialisasi ke nilai yang digantikan.

## 5.2 PERNYATAAN MOVE & STOP

## PERNYATAAN MOVE

Termasuk kedalam pernyataan untuk manipulasi data, dimana pernyataan ini digunakan untuk memindahkan data dari suatu field ke lokasi field lainnya sehingga input data dapat dimanipulasi untuk menghasilkan output. Statement ini dapat digunakan untuk *group item* dan juga *elementary item*. Untuk memindahkan data string, dapat menggunakan **MOVE(x:l)** dimana x merupakan posisi awal dan l adalah panjangnya.

Bentuk umum 1 :

```
MOVE {Identifier-1 Literal} TO Identifier-2 {Identifier-3}...
```

Atau

```
MOVE nama-data1 TO nama-data2 [,nama-data-3]...
```

Identifier-1 atau literal-1 sering disebut dengan sending area, karena nilai nilainya akan ditransfer ke area lainnya. Sedangkan identifier-2, identifier-3 dan seterusnya disebut dengan receiving area. Dalam bentuk umum 1 sending area bisa berupa suatu group item, alfabetik, alfanumerik, alfanumerik edited, numerik, numerik edited atau figurative constants. Sedangkan receving areanya harus merupakan area tersendiri, jadi dalam hal ini tidak diperkenankan untuk memakai literal.

Bentuk umum 2 :

```
MOVE CORRESPONDING nama-data1 TO nama-data2
         CORR
```

Untuk group item, digunakan bentuk khusus dari move yaitu MOVE CORRESPONDING yang berguna untuk memindahkan data dari group data item ke group lainnya.

### PERNYATAAN STOP

Pernyataan ini digunakan untuk menghentikan program baik secara permanen maupun secara sementara.

Bentuk umum :

```
STOP {literal}
     {RUN}
```

Stop literal menyebabkan proses program terhenti sementara dan literal akan ditampilkan dilayar. Literal yang dimaksud dapat *berupa numeric literal, non numeric literal* atau *figurative constant*. Untuk *numeric literal* harus berbentuk bilangan bulat yang tidak bertanda. Jika operator menekan sembarang tombol maka program akan dilanjutkan mulai statement setelah STOP litertal tersebut. Sedangkan untuk STOP RUN, program akan berhenti secara permanen.

### 5.4 PERNYATAAN ADD

Pernyataan ADD digunakan untuk menambahkan dua atau lebih operand numerik dan menyimpan hasilnya.

Bentuk umum secara sederhana dari pernyataan ADD yaitu :

```
ADD A B TO C D
ADD A B C TO D GIVING E
ADD CORR WS-GROUP1 TO WS-GROUP2
```

Penjelasan :

- Pada sintaks yang pertama, A, B, C ditambahkan dan hasilnya disimpan dalam C **(C = A + B + C)**. A, B, D ditambahkan dan hasilnya disimpan dalam D **(D = A + B + D)**.
- Pada sinyaks yang kedua, A, B, C, D ditambahkan dan hasilnya disimpan dalam E **(E = A + B + C + D)**.
- Pada sintaks yang ketiga, terdapat sub-grup item yaitu WS-GROUP1 dan WS-GROUP2 yang ditambahkan dan hasilnya akan disimpan dalam WS-GROUP2. (** WS-GROUP = Working Storage Group).

Untuk mempermudah perhitungan perhitungan yang telah disediakan beberapa option yaitu :

1. CORRESPONDING OPTION

   Dipakai untuk melaksanakan proses perhitungan atas beberapa data name yang dilakukan sekaligus. Dengan persyaratan bahwa data name harus merupakan bagian dari suatu group item tertentu dan harus merupakan elementary item dengan nama yang sama. Dalam penulisannya cukup disebutkan nama dari group itemnya saja.

2. GIVING OPTION

   Data name yang dituliskan setelah option ini merupakan tempat untuk menampung hasil perhitungan yang telah dilaksanakan. Bila data name tersebut tidak dipakai lagi untuk perhitungan selanjutnya, maka boleh berupa suatu *numeric edited item.*

3. ROUNDED OPTION

   Dipakai untuk membulatkan hasil perhitungan yang sudah dilaksanakan. Sebagai contoh bilangan suatu hasil perhitungan adalah 123,4 sedangkan tempat untuk menampung tidak disediakan tempat untuk angka pecahan. Maka dengan menggunakan option ini, hasil yang ditransfer adalah 123. Jika hasil perhitungan tersebut adalah 123,55 maka yang ditransfer adalah 124.

4. SIZE ERROR OPTION

   Jika area untuk menampung hasil perhitungan ternyata tidak mencakupi untuk menampung hasil tersebut, maka *imperative statement* sesudah option ini akan dilaksanakan. Misalnya, suatu angka yang dibagi dengan nol, seharusnya hasil perhitungan adalha tak terhingga, maka tidak akan ada area yang sanggup menampung hasil tersebut. Dan jika option ini dipakai maka proses akan dipindahkan ke *imperative statement* yang mengikutinya.

## 5.5 PERNYATAAN SUBSTRACT

Pernyataan substract digunakan untuk operasi pengurangan suatu nilai data numerik.

Bentuk umum :

```
SUBTRACT A B FROM C D
SUBTRACT A B C FROM D GIVING E
SUBTRACT CORR WS-GROUP1 TO WS-GROUP2
```

Penjelasan :

- Pada sintaks yang pertama data name A dan B ditambahkan dan dilakukan pengurangan dengan data name C. Dan hasil operasinya disimpan didalam data name C maka dapat

disimbolkan dengan **C = C - (A+B)**. Begitu juga dengan data name D dimana data name A dan B ditambahkan dan dikurangi dengan data name D, dan hasilnya disimpan dalam data name D maka pengoperasian tersebut dapat disimbolkan dengan **D = D - (A+B).**

- Pada sintaks kedua, data name A, B dan C ditambahkan dan dikurangi dengan data name D, hasilnya disimpan didalam data name E, maka dapat disimbolkan dengan **E= D - (A + B +C)**.
- Pada sintaks ketiga, item sub-group dari WS-GROUP1 dan WS-GROUP2 dikurangi dan hasilnya disimpan didalam WS-GROUP2.

Pada pendeklarasian pernyataan substract terdapat beberapa optional seperti corresponding, giving, rounded dan size error yang penjelasannya sama dengan yang sudah dijelaskan pada sub bab sebelumnya.

## 5.6 PERNYATAAN MULTIPLY

Pernyataan multiply digunakan untuk mengkalikan dua nilai numerik dan menyimpan hasilnya kedalam suatu data name.

Bentuk umum :

```
MULTIPLY A BY B C
MULTIPLY A BY B GIVING E
```

Pada sintaks pertama, data name A dan B dikalikan dan hasilnya akan disimpan didalam data name B **( B = A * B).** Juga data name A dan C dikalikan dan hasilnya disimpan dalam data name C **(C = A * C)**. Sedangkan dalam sintaks kedua data name A dan B dikalikan dan hasilnya disimpan dalam data name E **(E = A * B)**.

Pada pendeklarasian pernyataan multiply terdapat beberapa optional seperti giving, rounded dan size error yang penjelasannya sama dengan yang sudah dijelaskan pada sub bab sebelumnya.

## 5.7 PERNYATAAN DIVIDE

Pernyataan divide digunakan untuk melakukan operasi pembagian antar dua nilai.

Bentuk umum :

```
DIVIDE A INTO B.
DIVIDE A BY B GIVING C REMAINDER R.
```

Jika menggunakan optional INTO, maka data name B dibagi oleh data name A. Untuk penggunaan optional BY, maka yang terjadi adalah sebaliknya yaitu data name A dibagi data name B. Dan hasilnya disimpan dalam data name C dan sisanya disimpan dalam data name R.

Pada sintaks pertama, data name B dibagi dengan data name A dan hasilnya akan disimpan dalam data name B **(B = B / A)**. Sedangkan pada sintaks kedua, data name A dibagi dengan data name B dan hasilnya akan disimpan dalam data name C **(C = A / B)** dan sisanya disimpan dalam data-name R.

## 5.8 PERNYATAAN COMPUTE

Pernyataan ini digunakan untuk operasi yang lebih rumit, yaitu untuk menyederhanakan 4 *arithmatic verb* sebelumnya. Jadi pernyataan ini digunakan untuk menulis *arithmatica-expression* dalam cobol, yaitu sebagai pengganti untuk statement Add, Substract, Multiply dan Divide.

Contoh penggunaannya dalam menyelesaikan perhitungan rumit berikut :

```
Y = A + B - C / D * E
```

Pada COBOL untuk menyelesaikan perhitungan tersebut dapat menggunakan statement berikut :

```
ADD A TO B
DIVIDE D INTO C
MULTIPLY C BY E
SUBSTRACT E FROM B GIVING Y
```

Dengan menggunakan pernyataan COMPUTE pernyataan diatas akan menjadi lebih sederhana menjadi :

```
COMPUTE Y = A + B - C/D * E
```