SORT DAN MERGE PADA COBOL

OBJEKTIF :

1. Mahasiswa Mengetahui Pengetahuan Dasar Tentang Pernyataan Sort dan Merge pada COBOL.
2. Mahasiswa Mampu Membuat Program Sort dan Merge pada Aplikasi OpenCobolIDE.
3. Mahasiswa Mampu Memahami Struktur Pernyataan Sort dan Merge

---

12.1 PERNYATAAN SORT

Dalam program COBOL, pernyataan SORT biasanya digunakan untuk mengurutkan file berurutan. Beberapa pemrogram mengklaim bahwa pernyataan SORT tidak diperlukan. Tetapi satu keuntungan utama menggunakan pernyataan SORT adalah meningkatkan portabilitas program COBOL.

Karena pernyataan SORT tersedia di setiap COBOL compiler, ketika program yang menggunakan SORT harus dipindahkan ke sistem komputer lain, program tersebut dapat melakukan transisi tanpa memerlukan perubahan pada SORT. SORT sederhana bekerja dengan mengambil record dari file USING, menyortirnya, dan kemudian menulisnya ke file GIVING.

File berikut digunakan dalam pernyataan SORT:

1. Input File.

   File ini digunakan untuk mengurutkan dalam urutan ascending atau descending.

2. Work File.

   File ini menyimpan record saat proses penyortiran sedang berlangsung. Dalam proses penyortiran, input file record dipindahkan ke Work file. File ini didefinisikan di File-Section di bawah entri SD.

3. Output File.

   File ini adalah hasil akhir dari pernyataan SORT. Ini diperoleh setelah proses penyortiran.

Bentuk umum kata kerja SORT sederhana :

```
SORT SDWorkFileName
    { ON   { ASCENDING  }  }    KEY    SortKeyIdentifier
           { DESCENDING }  }
      [ WITH DUPLICATES IN ORDER ]
      [ COLLATING SEQUENCE IS Alphabet ]
      USING  { InFileName }
      GIVING { OutFileName }
```

Keterangan :

- **SDWorkFileName** mengidentifikasi file kerja sementara yang digunakan proses SORT untuk pengurutan. Ini didefinisikan dalam FILE SECTION menggunakan entri SD (Stream/Sort Description). Meskipun file kerja adalah file sementara, SDWorkFileName masih harus memiliki klausa SELECT dan ASSIGN terkait di ENVIRONMENT DIVISION.
- **SDWorkFileName** adalah file Sequential dengan organisasi RECORD SEQUENTIAL. Karena ini adalah organisasi default, maka biasanya dihilangkan.
- **SortKeyIdentifie**r mengidentifikasi bidang dalam rekaman file kerja. File yang diurutkan akan berurutan pada key field ini. Jika lebih dari satu SortKeyIdentifier ditentukan, signifikansi kunci berkurang dari kiri ke kanan (kunci paling kiri paling signifikan, paling kanan paling signifikan).
- **InFileName** dan **OutFileName**, masing-masing adalah nama file input dan output.
- Jika klausa **DUPLICATES** digunakan, maka ketika file telah diurutkan, urutan akhir rekaman dengan kunci duplikat sama dengan yang ada di file yang tidak diurutkan. Jika tidak ada klausa DUPLICATES yang digunakan, urutan rekaman dengan kunci duplikat tidak ditentukan.
- **AlphabetName** adalah nama-alfabet yang ditentukan dalam paragraf SPECIAL-NAMES dari ENVIRONMENT DIVISION. Klausa ini digunakan untuk memilih kumpulan karakter yang digunakan kata kerja SORT untuk menyusun catatan dalam file. Rangkaian karakter mungkin ASCII (8 atau 7 bit), EBCDIC, atau ditentukan pengguna.
- SORT dapat digunakan dimanapun di PROCEDURE DIVISION kecuali di INPUT PROCEDURE, PROSEDUR OUTPUT atau di DECLARATIVE SECTION.
- Record yang dijelaskan untuk file input (**USING**) harus sesuai dengan record yang dijelaskan untuk SDWorkFileName.
- Record yang dijelaskan untuk SDWorkFileName harus sesuai dengan record yang dijelaskan untuk file keluaran (**GIVING**).
- Deskripsi **WorkSortKey** tidak dapat berisi klausa OCCURS (tidak dapat berupa tabel).

Arah operasi penggabungan tergantung pada spesifikasi kata kunci ASCENDING atau DESCENDING sebagai berikut:

- Jika ASCENDING, maka urutannya adalah dari nilai kunci terendah hingga nilai kunci tertinggi.
- Jika DESCENDING, maka urutannya adalah dari nilai kunci tertinggi hingga nilai kunci terendah.

SORT menggunakan PROCEDURE

Versi sederhana SORT mengambil record dari InFileName, mengurutkannya, dan kemudian mengeluarkannya ke OutFileName. Terkadang, bagaimanapun, tidak semua record dalam file yang tidak diurutkan diperlukan dalam file yang diurutkan, atau tidak semua item data dalam record file yang tidak disortir diperlukan dalam record file yang diurutkan. Maka dari itu, dapat digunakan PROCEDURE dalam pernyataan SORT.

Procedure yang dapat digunakan dalam kata kerja SORT yaitu Input Procedure dan Output Procedure. Ketika INPUT PROCEDURE digunakan, tidak ada file USING sehingga proses pengurutan harus mendapatkan record dari INPUT PROCEDURE. INPUT PROCEDURE menggunakan kata kerja RELEASE untuk menyediakan record ke file kerja SORT, satu per satu waktu.

SORT menggunakan INPUT PROCEDURE

INPUT PROCEDURE memasok record ke proses pengurutan dengan menuliskannya ke file kerja yang dideklarasikan dalam entri SD SORT. Tetapi untuk menulis record ke file pekerjaan, menggunakan special verb yaitu RELEASE.

Sintaks dari kata kerja RELEASE :

```
RELEASE SDRecordName [FROM Identifier].
```

Pada sintaks tersebut, SDRecordName adalah nama rekaman yang dideklarasikan dalam entri SD file kerja.

Template operasional untuk INPUT PROCEDURE, yang mendapatkan record dari file input dan RELEASE ke file kerja, ditunjukkan pada sintaks di bawah ini.

```
OPEN INPUT InFileName.
READ InFileName.
PERFORM UNTIL TerminatingCondition.
   Process input record.
   RELEASE SDWorkRec.
   READ InFileName.
END-PERFORM.
CLOSE InFileName.
```

SORT menggunakan OUTPUT PROCEDURE

OUTPUT PROCEDURE digunakan untuk mengambil yang record dari file kerja menggunakan kata kerja RETURN. OUTPUT PROCEDURE hanya dijalankan setelah file disortir.

Sintaks dari kata kerja RETURN :

```
RETURN SDFileName RECORD [INTO Identifier].
         AT END StatementBlock.
END-RETURN.
```

Pada sintaks tersebut, SDFileName adalah nama file yang dideklarasikan dalam entri SD.

Template operasional untuk OUTPUT PROCEDURE, yang mendapatkan record dari file kerja dan menulisnya ke file Output. Perhatikan bahwa file kerja tidak dibuka oleh kode dalam OUTPUT PROCEDURE. File pekerjaan secara otomatis dibuka oleh SORT.

Berikut ini merupakan template file output :

```
OPEN OUTPUT OutFile.
RETURN SDWorkFile RECORD.
PERFORM UNTIL TerminatingCondition.
   Setup OutRec.
   WRITE OutRec.
   RETURN SDWorkFile RECORD.
END-PERFORM.
CLOSE OutFile.
```

Bentuk penulisan pernyataan SORT menggunakan PROCEDURE :

```
SORT SDWorkFileName
     { ON { ASCENDING  } }    KEY    SortKeyIdentifier
          { DESCENDING }
       [ WITH DUPLICATES IN ORDER ]
       [ COLLATING SEQUENCE IS Alphabet ]
     { INPUT PROCEDURE IS ProcName { { THRU     } ProcName } }
                                   { { THROUGH  }           }
       USING  { InFileName }
     { OUTPUT PROCEDURE IS ProcName { { THRU     } ProcName } }
                                    { { THROUGH  }           }

       GIVING { OutFileName }
```

Keterangan :

- Kata kerja SORT .. GIVING tidak dapat digunakan jika OUTPUT PROCEDURE digunakan.

Contoh Program menggunakan pernyataan SORT :

```
       IDENTIFICATION DIVISION.
       PROGRAM-ID. SORT-PROGRAM.
       ENVIRONMENT DIVISION.
       INPUT-OUTPUT SECTION.
       FILE-CONTROL.
                SELECT INPUT1 ASSIGN TO IN1.
                SELECT INPUT2 ASSIGN TO IN2.
                SELECT OUTPUT1 ASSIGN TO OUT.
                SELECT WORK ASSIGN TO WRK.

       DATA DIVISION.
       FILE SECTION.
       FD INPUT1.
       01 INPUT1-STUDENT.
             05 STUDENT-ID-I1 PIC 9(5).
             05 STUDENT-NAME-I1 PIC A(25).
```

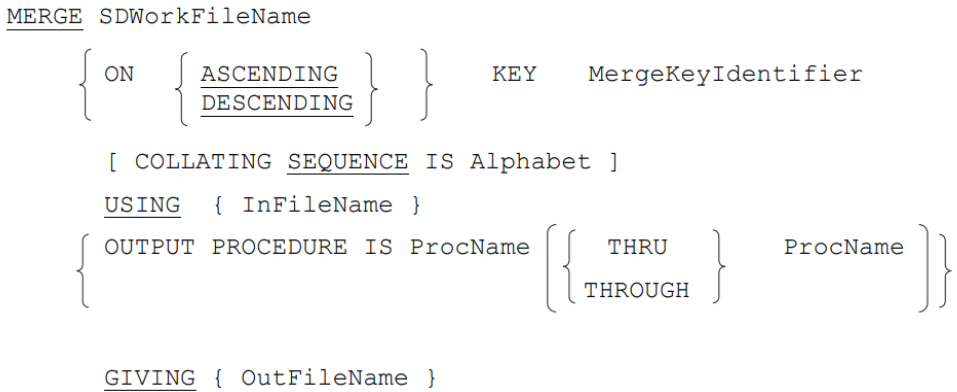
```
FD INPUT2.
01 INPUT2-STUDENT.
      05 STUDENT-ID-I2 PIC 9(5).
      05 STUDENT-NAME-I2 PIC A(25).
FD OUTPUT1.
01 OUTPUT-STUDENT.
      05 STUDENT-ID-O PIC 9(5).
      05 STUDENT-NAME-O PIC A(25).

SD WORK.
01 WORK-STUDENT.
      05 STUDENT-ID-W PIC 9(5).
      05 STUDENT-NAME-W PIC A(25).

PROCEDURE DIVISION.
     SORT WORK ON ASCENDING KEY STUDENT-ID-O
       USING INPUT1, INPUT2 GIVING OUTPUT1.
       DISPLAY 'SORT SUCCESSFULL'.
STOP RUN.
```

Pada program di atas, terdapat 2 buah file input bernama INPUT1 dan INPUT2. Kedua file tersebut akan digunakan sebagai file input yang akan diurutkan. Di dalam file tersebut berisi record student-id dan student-name. SDWorkFile pada program tersebut yaitu bernama WORK. File ini berfungsi untuk memproses pernyortiran kedua file input yang digunakan. OUTPUT1 menunjukkan file input yang akan menampung hasil sortir dari program tersebut. Apabila proses ini berhasil dijalankan, maka program akan menampilkan output 'SORT SUCCESSFULL'.

12.2 PERNYATAAN MERGE

Pernyataan MERGE merupakan pernyataan yang digunakan untuk menggabungkan dua atau lebih file yang diurutkan secara identik pada satu set kunci yang ditentukan, dan selama proses membuat record tersedia, dalam urutan yang digabungkan ke Output Procedure atau ke file Output.

Bentuk umum pernyataan MERGE :



Keterangan :

- Hasil dari kata kerja MERGE dapat diprediksi hanya jika record dalam file input diurutkan seperti yang dijelaskan dalam klausa KEY yang terkait dengan pernyataan MERGE. Misalnya, jika pernyataan MERGE memiliki klausa ON DESCENDING KEY dengan nama key identifier StudentId, maka semua file USING harus diurutkan pada StudentId secara menurun.
- Seperti pada SORT, SDWorkFileName adalah nama file sementara, dengan entri SD di FILE SECTION, entri SELECT dan ASSIGN di INPUT-OUTPUT SECTION, dan organisasi RECORD SEQUENTIAL.
- InFileName dan OutFileName, masing-masing adalah nama file input dan output.
- MERGE dapat menggunakan OUTPUT PROCEDURE dan kata kerja RETURN untuk mendapatkan record gabungan dari SDWorkFileName.
- OUTPUT PROCEDURE hanya dijalankan setelah file digabungkan dan harus berisi setidaknya satu pernyataan RETURN untuk mendapatkan record dari SortFile.

Contoh program menggunakan pernyataan Merge :

```
        IDENTIFICATION DIVISION.
        PROGRAM-ID. SORT-PROGRAM.
        ENVIRONMENT DIVISION.
        INPUT-OUTPUT SECTION.
        FILE-CONTROL.
                SELECT INPUT1 ASSIGN TO IN1.
                SELECT INPUT2 ASSIGN TO IN2.
                SELECT OUTPUT1 ASSIGN TO OUT.
                SELECT WORK ASSIGN TO WRK.

        DATA DIVISION.
        FILE SECTION.
        FD INPUT1.
        01 INPUT1-STUDENT.
             05 STUDENT-ID-I1 PIC 9(5).
             05 STUDENT-NAME-I1 PIC A(25).
        FD INPUT2.
        01 INPUT2-STUDENT.
             05 STUDENT-ID-I2 PIC 9(5).
             05 STUDENT-NAME-I2 PIC A(25).
        FD OUTPUT1.
        01 OUTPUT-STUDENT.
             05 STUDENT-ID-O PIC 9(5).
             05 STUDENT-NAME-O PIC A(25).
        SD WORK.
        01 WORK-STUDENT.
             05 STUDENT-ID-W PIC 9(5).
             05 STUDENT-NAME-W PIC A(25).

        PROCEDURE DIVISION.
        MERGE WORK ON ASCENDING KEY STUDENT-ID-O
               USING INPUT1, INPUT2 GIVING OUTPUT1.
                   DISPLAY 'MERGE SUCCESSFULL'.
        STOP RUN.
```

Pada program di atas, terdapat 2 buah file input bernama INPUT1 dan INPUT2. Kedua file tersebut akan digunakan sebagai file input yang akan diurutkan. Di dalam file tersebut berisi record student-id dan student-name. SDWorkFile pada program tersebut yaitu bernama WORK. File ini berfungsi untuk memproses penggabungan kedua file input yang digunakan. OUTPUT1

menunjukkan file input yang akan menampung hasil merge dari program tersebut. Apabila proses ini berhasil dijalankan, maka program akan menampilkan output 'MERGE SUCCESSFULL'.