COBOL - DATA LAYOUT

COBOL layout is the description of use of each field and the values present in it. Following are the data description entries used in COBOL:

- Redefines Clause
- Renames Clause
- Usage Clause
- Copybooks

Redefines Clause

Redeifnes clause is used to define a storage with different data description. If one or more data items are not used simultaneously, then the same storage can be utilized for another data item. So the same storage can be referred with different data items.

Syntax

Following is the syntax for Redefines clause:

```
01 WS-OLD PIC X(10).
01 WS-NEW1 REDEFINES WS-OLD PIC 9(8).
01 WS-NEW2 REDEFINES WS-OLD PIC A(10).
```

Following are the details of the used parameters:

- WS-OLD is Redefined Item
- WS-NEW1 and WS-NEW2 are Redefining Item

Level numbers of redefined item and redefining item must be the same and it cannot be 66 or 88 level number. Do not use VALUE clause with a redefining item. In File Section, do not use a redefines clause with 01 level number. Redefines definition must be the next data description you want to redefine. A redefining item will always have the same value as a redefined item.

Example

```
IDENTIFICATION DIVISION.
PROGRAM-ID. HELLO.

DATA DIVISION.
   WORKING-STORAGE SECTION.
   01 WS-DESCRIPTION.
   05 WS-DATE1 VALUE '20140831'.
   10 WS-YEAR PIC X(4).
   10 WS-MONTH PIC X(2).
   10 WS-DATE PIC X(2).
   05 WS-DATE2 REDEFINES WS-DATE1 PIC 9(8).

PROCEDURE DIVISION.
   DISPLAY "WS-DATE1 : "WS-DATE1.
   DISPLAY "WS-DATE2 : "WS-DATE2.

STOP RUN.
```

JCL to execute the above COBOL program.

```
//SAMPLE JOB(TESTJCL,XXXXXX),CLASS=A,MSGCLASS=C
//STEP1 EXEC PGM=HELLO
```

When you compile and execute the above program it produces the following result:

```
WS-DATE1 : 20140831
WS-DATE2 : 20140831
```

Renames Clause

Renames clause is used to give different names to existing data items. It is used to re-group the data names and give a new name to them. The new data names can rename across groups or elementary items. Level number 66 is reserved for renames.

Syntax

Following is the syntax for Renames clause:

```
01 WS-OLD.
10 WS-A PIC 9(12).
10 WS-B PIC X(20).
10 WS-C PIC A(25).
10 WS-D PIC X(12).
66 WS-NEW RENAMES WS-A THRU WS-C.
```

Renaming is possible at same level only. In the above example ,WS-A, WS-B, and WS-C are at the same level. Renames definition must be the next data description you want to rename. Do not use Renames with 01,77, or 66 level number. The data names used for renames must come in sequence. Data items with occur clause cannot be renamed.

Example

```
IDENTIFICATION DIVISION.
PROGRAM-ID. HELLO.

DATA DIVISION.
   WORKING-STORAGE SECTION.
   01 WS-DESCRIPTION.
   05 WS-NUM.
   10 WS-NUM1 PIC 9(2) VALUE 20.
   10 WS-NUM2 PIC 9(2) VALUE 56.
   05 WS-CHAR.
   10 WS-CHAR1 PIC X(2) VALUE 'AA'.
   10 WS-CHAR2 PIC X(2) VALUE 'BB'.
   66 WS-RENAME RENAMES WS-NUM2 THRU WS-CHAR2.

PROCEDURE DIVISION.
   DISPLAY "WS-RENAME : " WS-RENAME.

STOP RUN.
```

JCL to execute the above COBOL program.

```
//SAMPLE JOB(TESTJCL,XXXXXX),CLASS=A,MSGCLASS=C
//STEP1 EXEC PGM=HELLO
```

When you compile and execute the above program, it produces the following result:

```
WS-RENAME : 56AABB
```

Usage Clause

Usage clause specifies the operating system in which the format data is stored. It can not be used with level numbers 66 or 88. If usage clause is specified on a group, then all the elementary items will have the same usage clause.The different options available with Usage clause are as follows:

Display

Data item is stored in ASCII format and each character will take 1 byte. It is default usage.

The following example to calculates the number of bytes required:

```
01 WS-NUM PIC S9(5)V9(3) USAGE IS DISPLAY.
It requires 8 bytes as sign and decimal doesn't require any byte.

01 WS-NUM PIC 9(5) USAGE IS DISPLAY.
It requires 5 bytes as sign.
```

COMPUTATIONAL / COMP

Data item is stored in binary format. Here data items must be integer.

The following example calculates the number of bytes required:

```
01 WS-NUM PIC S9(n) USAGE IS COMP.

If 'n' = 1 to 4, it takes 2 bytes.
If 'n' = 5 to 9, it takes 4 bytes.
If 'n' = 10 to 18, it takes 8 bytes.
```

COMP-1

Data item is similar to Real or Float and is represented as a single precision floating point number. Internally data is stored in hexadecimal format. COMP-1 does not accept PIC clause. Here 1 word is equal to 4 bytes.

COMP-2

Data item is similar to Long or Double and is represented as double precision floating point number. Internally data is stored in hexadecimal format. COMP-2 does not specify PIC clause. Here 2 word is equal to 8 bytes.

COMP-3

Data item is stores in pack decimal format. Each digit occupies half a byte $1nibble$ and the sign is stored at the right most nibble.

The following example calculates the number of bytes required:

```
01 WS-NUM PIC 9(n) USAGE IS COMP.
Number of bytes = n/2 (If n is even)
Number of bytes = n/2 + 1(If n is odd, consider only integer part)

01 WS-NUM PIC 9(4) USAGE IS COMP-3 VALUE 21.
It requires 2 bytes of storage as each digit occupies half a byte.

01 WS-NUM PIC 9(5) USAGE IS COMP-3 VALUE 21.
It requires 3 bytes of storage as each digit occupies half a byte.
```

Copybooks

A COBOL copybook is a selection of code that defines data structures. If a particular data structure is used in many programs then instead of writing the same data structure again, we can use copybooks. We use the COPY statement to include a copybook in a program. COPY statement is used in the Working-Storage Section.

the following example to include copybook inside COBOL program:

```
DATA DIVISION.
WORKING-STORAGE SECTION.
COPY ABC.
```

Here ABC is the copybook name. The following data items in ABC copybook can be used inside a program.

```
01 WS-DESCRIPTION.
   05 WS-NUM.
      10 WS-NUM1 PIC 9(2) VALUE 20.
      10 WS-NUM2 PIC 9(2) VALUE 56.
   05 WS-CHAR.
      10 WS-CHAR1 PIC X(2) VALUE 'AA'.
      10 WS-CHAR2 PIC X(2) VALUE 'BB'.
```