UNIT 6

Library Services

COPY Statement

The COPY statement is a library statement that places prewritten text in a COBOL program

Each COPY must be terminated by a period

If 'library-name' is omitted, then SYSLIB is assumed

Format:

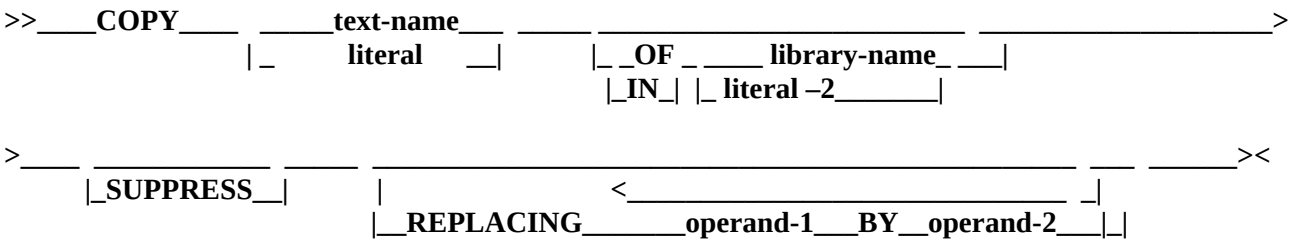

Figure 6-1 COPY Statement

Notes:

SUPRRESS means that the imbedded text will not be printed in the source program listing

COPY requires the LIB compiler option to be in effect

The COPY statement is a library statement that places prewritten text in a COBOL program.

Prewritten source program entries can be included in a source program at compile time. Thus, an installation can use standard file descriptions, record descriptions, or procedures without recording them. These entries and procedures can then be saved in user-created libraries; they can then be included in the source program by means of the COPY statement.

Compilation of the source program containing COPY statements is logically equivalent to processing all COPY statements before processing the resulting source program.
The effect of processing a COPY statement is that the library text associated with text-name is copied into the source program, logically replacing the entire COPY statement, beginning with the word COPY and ending with the period, inclusive. When the REPLACING phrase is not specified, the library text is copied unchanged.

Each COPY statement must be preceded by a space and ended with a separator period.

Debugging lines are permitted within library text and pseudo-text.

Comment lines or blank lines can occur in library text. Comment lines or blank lines appearing in library text are copied into the resultant source program unchanged with the following exception: a command line or blank line in library text is not copied if that comment line or blank line appears within the sequence of text words that match operand-1

A COPY statement can appear in the source program anywhere a character string or a separator can appear; however, a COPY statement must not be specified within a COPY statement. The resulting copied text must not contain a COPY statement.

Nested COPY

COPY FILEA.

FILEA contains:

```
01 FILEA.
      05  NAME     PIC  X(40).
      COPY ADDRESS.
       05  DATA     PIC   X(100).
```

ADDRESS contains:

```
05  STREET  PIC  X(40).
05   CITY      PIC  X(20).
05    STATE   PIC  X(02).
05     ZIP       PIC  X(09).
```

Compile produces:

```
01   FILEA.
     05      NAME       PIC   X(40).
     05     STREET     PIC   X(40).
     05      CITY        PIC  X(20).
     05      STATE      PIC   X(02).
     05      ZIP          PIC  X(09).
     05      DATA       PIC  X(100).
```

Figure 6-2: Nested COPY

Notes:

COBOL allows nested COPY statements.
Nested COPY statements cannot contain the REPLACING phrase.
A COPY statement can appear in the source program anywhere a character string or a separator can appear. As an IBM extension, COPY statements can be nested. However, nested COPY statements cannot contain the REPLACING phrase, and a COPY statement with the REPLACING phrase cannot contain nested COPY statements.

A COPY statement cannot cause recursion. That is, a COPY member can be named only once in a set of nested COPY statements until the end-of-file for that COPY member is reached.

COPY REPLACING

To change some, or all, of the names in the library (COPY'ed) text, the programmer can use the REPLACING option.

The text in the library is unchanged.

```
COPY PAYLIB REPLACING
      FLDA   BY  PAY-RECORD
       FLDA  BY  HRLY-RATE
       FLDA  BY  HRS-WORKD.
```

```
LIBRARY TEXT                    SOURCE PROGRAM
01  FLDA.                       01  PAY-RECORD.
    02  FLDB  PIC  999V99.          02  HRLY-RATE  PIC  999V99.
    02  FLDC  PIC  999V99.          02  HRS-WORKD PIC 999V99.
```

Figure 6-3 COPY REPLACING

In the discussion that follows, each operand can consist of one of the following:

- Pseudo-text
- An identifier
- A literal
- A COBOL word
- Function identifier

COPY Pseudo-Text

- **To change only part of the data-name(s) in the library text, the programmer can use the REPLACING option with the standard pseudo-text delimiters (==)**

```
COPY PAYLIB REPLACING
            = = : PFFX: = =      BY    = = PAY = =.
```

```
LIBRAR TEXT                      SOURCE PROGRAM

01  :PFFX:.                     01  PAY.
    02  :PFFX:-RTE                  02  PAY-RTE
            PIC  999V99.                 PIC  999V99.
    02  :PFFX: - HRS                02  PAY-HRS
            PIC 999V99.                  PIC  999V99.
```

Figure 6-4 COPY pseudo-Text

Notes :

Pseudo-text A sequence of character-strings and/or separators bounded by, but not including, pseudo-text-1 delimiters (= =). Both characters of each pseudo-text-1 delimiter must appear on one line; however, character-strings within pseudo-text-1 can be continued.

Any individual character-string within pseudo-text-1 can be up to 322 characters long.

Pseudo-text-1 cannot be null, nor can it consist solely of the space character, separator comma, separator semicolon, and/or of comment lines. Beginning and ending blanks are not included in the text comparison process. Embedded blanks are used in the text comparison process to indicate multiple text words.

Pseudo-text must not contain the word COPY.

REPLACE Pseudo-test

- Replace can be applied to the entire program, including text introduced through COPY members

- Replace action starts at a the REPLACE statement and continues until:
  - Another REPLACE statement
  - REPLACE OFF statement
  - End of source program

- REPLACE statements are processed by the compiler after any COPY statements are processed

Figure 6-5 REPLACE Pseudo-Text