COBOL-80

参考手册

《通信与计算机》编辑部

一九八五年九月

编 者 的 话

COBOL（Common Business Oriented Language）是一种面向商业和事务处理的通用程序设计语言。至今为止，几乎所有的微型计算机都配有 COBOL 语言。由此可见，COBOL不仅有着它严密的独特风格，还有着它生存和发展的广阔前景，目前已有好多微机能用COBOL语言进行汉字处理。

这本《Microsoft COBOL—80 参考手册》全书共分十章，后面有七个附录。其中，第一章介绍了COBOL的一些基本概念，第二至第四章着重介绍了COBOL语言成份和程序编制规则，第五至第十章分别介绍了程序间通信，表处理，索引文件，相对文件，DECLARATIVES 和USE子句，程序分段。当然一至四章是作为必备的知识，五至十章有着一定的深度和难度，可根据不同的情况选用。

本手册是美国著名的 Microsoft 软件公司于1980年为 APPLE Ⅱ 微机配制的 COBOL 语言。运行本 COBOL 语言需要有 CP/M2.2 版操作系统的支持，本 COBOL 语言不仅能在 APPLE Ⅱ 机上运行，还能在紫金Ⅱ、爱卡等同类微机上运行，同时只要稍加一些硬设备还能进行汉字处理。

本书由紫金信息工业公司电脑分公司软件所胡文权同志翻译，编审工作中龚延范、徐庭生、刘同令等同志参加了工作。

由于我们水平有限，时间仓促，在本手册当中必定会有不少的错误，务请广大的用户、读者提出宝贵的意见，并表示感谢！

编 者

八五年九月

目　录

引　言

Microsoft COBOL 是以美国国家的标准 X3.23—1974 为基础。COBOL语言元素被分配到十二个不同功能的处理“模块”。

标准COBOL的每个模块有两种非—零“级”，一级代表全集的一个子集，其性能和特征包含在二级中。

为了指出系统调用 COBOL，它必须在最小核心一级中提供表处理和顺序的 I—O 模块。

下面摘要说明一下关于 Microsoft COBOL 的内容标准。

模　块	Microsoft COBOL 的特点：
核　心	一级的全部加上二级的如下特性：
	条件：
	带有值系列或值域的88层条件名。
	在条件中使用逻辑运算符 AND/OR/NOT
	对于等式或不等式（，>，=）使用代数关系符
	在关系条件中，意味着从属于，或既从属又关系
	符号测试
	IF语句的嵌套；在条件中的括弧
	过程语句：
	作为ACCEPT和DISPLAY屏幕处理格式的扩充
	带DATE／DAY／TIME任选的ACCEPT 语句
	STRING语句和UNSTRING语句
	多结果域的COMPUTE 语句
	带VARYING…UNTIL短语的PERFORM语句
	标识符：
	助忆名作为ACCEPT或DISPLAY的设备
	过程名仅有数字组成
	名的限定（仅在过程部分语句中）
顺序，相对	一级的全部加二级的特性：
和索引 I／O	RESERVE子句
	在OPEN和CLOSE中的多种运算由每个文件个别选择
	VALUE OF FILE—ID是数据名
顺序 I／O	OPEN的EXTEND方式
	WRITE ADVANCINC 数据名行 LINAGE 短句和
	AT END—OF—PAGE 子句
相对和	DYNAMIC存取方式（用READ

引索 I/O	NEXT） START（用关系键EQUAL，GREATER，NOT LESS）
库	一级
程序间通信	一级
表处理	一级的全部，增加了SEARCH语句的二级格式
排错	对 ANSI—74 标准专门扩充规定的设备追踪一文本排错除非WITH DEBUGGING MODE在SOURCE—COMPUTER段中被给定，否则每行使用“在第7列中的D”被回避
程序分段	一级

第 一 章

COBOL 的基本概念

1.1 字符规定

COBOL源语言字符规定由下面的一些字符组成：

字母A到Z

空白或空格

数字0到9

专用符号：

+加符号

－减符号

*星　号

=等于符号

>关系符号（大于）

<关系符号（小于）

$美元符号

，逗　号

；分　号

。句号或小数点

“引　号

（左括号

）右括号

、省略号（交替引号）

／斜杠

根据上述规定，由下面的一些字符可作为词组使用：

0到9

A到Z

—（连字符）

下面的符号是用来作为加标点：

（左括号

）右括号

，逗　号

。句　号

；分　号

下面的关系符号用在简单条件中：

>
<
=

就非数值（引用）字面来说，如注解项，注解行，COBOL 字符的设置可扩大到计算机包含的全体的字符集。

1.2 标点符

在书写源程序的过程中下面的标点符是经常应用的：

1.例如标点符号，句号，分号，或逗号前可没有空格，但它的后面必须跟有一个空格。

2.在相继地出现词组和/或字面的两者之间必须要有一个空格。除了非数值字面中的以外，二个或更多的空格相继出现时仅当作一个空格来处理。

3.在关系符号的两边均要有一个空格出现。

4.当句号，逗号，加号或减号字符用在 PICTURE 子句时，他们作为记录项的规则被单独确定。

5.一个逗号可以作为语句中多个运算量之间或两个下标之间的分隔符。

6.一个分号或逗号可以用来分隔一连串的语句或子句。

1.3 字的形式

用户—自定义字和保留字由小于等于30个混合字符组成，其选择形式为下面的37个字符集：

0 到 9（数字）

A 到 Z（字母）

—（连字符）

所有的字最少必须包含一个字母或连字符，除了过程名之外，也许还会由全部数字组成。在一个字的开头或末尾不可使用连字符。一个字总是被一个空格或固有的标点符号结尾。一个字中含有插入的连字符可多于一个；也容许连字符的相邻插入。也不是那些预先有指定意义，或程序设计时提供名称的所有的字都为保留字。如果一个程序提供的名称不唯一，则必须以唯一标准的方法对它名字的使用给于限制，例如，TAX—RATE IN STATE—TABLE。首先，一个非保留字等同于一个数据项或域并称为一个数据名。另一样情况，非保留字是文件名，条件名，助忆名，和过程名。

1.4 格式表示法

通过这本书，介绍一下在书写语句本身时“一般格式”的规则，而这些规则适用于各种各样子句和语句的引导程序。他们是根据同一个系统所表示的符号，在下面各段中说明。

1.在所有的印刷字中保留字完全采用大写字母。这些是已经指定了意义的字。对于大写字母的字表示这些字在所有的格式中应如实出现。

2.除了格式包括的本身部分是自选的以外，所有的保留字要求带有下划线。这就是关键字。如果任何关键字被遗漏或者表示得不够正确，这在程序中被认为是一个错误。没有带下划线的保留字是包含还是省略由程序员选择。这些字是任选字；他们只是作为程序的可读性而来独自灵活使用。

3.字符< >=（虽然不带下划线）当需要使用时依照上述格式。

4.所有的标点符和别的专门字符表示实际存在这些字符。标点符是一种说明，它必不可

就是说对记录的细分到了其本身不能再可进一步细分时的数据项被称之为初等项。含有可细分的数据项通称为组项。当一个过程语句查阅有关组项时，作为被查阅的全部组项应有其保留区。所有的初等项必须用一个 PICTURE 或 USAGE IS INDEX 子描述。在工作存储节中相连的逻辑记录（01），对于任何由文件隐含表示的重定义其下属项规定了同一个区域，每个记录（01））也这样定义它自己的内存区。

包括组项在内被指定的数字越小则层号越高。在组项里面的层号并不需要连续的。所有的组项和初等项的描述包含在层号K之下，直到它遇到一个层号小于或等于K时为止。

作为每层的分隔符在书写时被插在源程序中。举个例子来说明层号和组项的关系，在上述的例子中，每周工时卡记录可由数据部分登录描述以下的层号，数据名和 PICTURE 定义。

```
01  TIME—CARD
    02  NAME
        03  LAST—NAME          PICTURE X(18).
        03  FIRST—INIT         PICTURE X.
        03  MIDDLE—INIT        PICTURE X.
    02  EMPLOYEE—NUN     PICTURE 99999
    02  WEEKS—END—DATE
        05  MONTH              PIC 99.
        05  DAY—NUMBER         PIC 99.
        05  YEAR               PIC 99.
    02  HOURS—WORKED     PICTURE 99V9.
```

数据名是由用户指定的字，它在程序中等同于一个数据项使用。一个数据名总被认为是一个数据区域，而不是一个特殊的值。项通常被假定为一个数，但在程序占有期间有不同的值。

数据名必须使用字母符号开头。数据名或关键字 FILLER 在每个记录描述项中必须紧跟在首字层号的后面，以下面的一般格式加以说明：

$$\text{层　号}\left\{\begin{array}{l}\text{数据名}\\ \underline{\text{FILLER}}\end{array}\right\}$$

这个数据名是项的定义名，并作为连续的数据区（含有数据项的值）。如果在一个记录中的某些字符在程序的处理步骤中不使用，然而，这些字符的数据描述不需要包含数据名。这时，可由 FILLER 写在层号之后代替一个数据名。

1.6 文件名

文件是数据记录的一种集合，例如，打印列表或软盘的一个区域，包括若干同种类型的记录或应用。文件名在数据部分的文件节中由一个 FD 项来定义。FD 是一个保留字，而它的后面是一个唯一的被称作为文件名的且由程序员填写的字。对于文件名字的组成规则同数据名一样（请见1.3节）。关于过程语句OPEN，CLOSE和READ中出现的文件名，应和在环境部分一样。注意：这里的文件名不能同在3.14.2节中用 ID 描述的文件混为一谈。

1.7 条件名

条件名在数据部分里面的88层项上被定义。它对被指定的名字一个专门的值，位或字段值，在那里完全可以假设数据项的值。关于名字的形成规则在 1.3 节中说明。表示条件名的说明和过程语句怎样使用他们这在数据和过程部分的某些章节中会专门给出。

1.8 助忆名

在环境部分中被指定的助忆名是为了在ACCEPT或DISPLAY语句中查阅。它指定一少。额外被插入的标点符，要根据节1.2中标点符的专门规定。一般情况下，因为本手册要求在格式中指定结尾的句号；分号和逗号因为他们是任选的所以通常不表示。对于所有的分隔符如在逗号，分号和句号的后面必须跟有一个空格（或空白）。

5.采用小写字母印刷的字在格式中表示发生项（例如，数据名），因为用户在源程序发生项上必须插入一个有用的项。

6.就是说无论是语句还是数据描述项在方括弧对中的任何部分是任选的。大括弧对（{ }）中的内容表示既要相互排拆又必须随意的选择一个。

7.在格式中由大写字母开头后面跟有“Clause”或“Statement”字样的都有明确的入口。这些指定的子句或语句在文本另外有关节的格式中说明。

8.为了做到参考容易，在文本解释性方面使用小写字体，有时他们的后面有一个连字符和一个数字或字母。改变了这个字的定义也不作语法上的检查。

9.另外的一些选择可分别情况加以说明，一个垂直笔划的选择表示互相排拆，例如：

AREA | AREAS是相当于 {AREA / AREAS}

10.省略符号（…）表示紧接以前的单位可以在以后出现一次或连续出现任意次。一个单位的意思是任何一个独立的小写体字，或一组小写体字或一个与更多个保留字括在方括弧对或大括弧对中。如果一个项出现在方括弧对或大括弧对中，当那个项被指定为重复时而这部分必须给予重复。

11.任选元素用方括弧表示而不是大括弧，这只要看格式的表示有无实际意义。

12.注解，限制和使用上的说明以及每种格式的意义均包含在本手册的各个章节里。

1.5 层号和数据名

对于处理的对象，文件的内容是内部逻辑记录，而逻辑记录的描述用层号01开始。其下属的数据项就构成逻辑记录的组项，从组项至以下各子项能使用的层号是2到49，但这些层号不一定是相连续的。另外还有，层号77在工作存储或连接节中等同于“Stand alone”项；就是说，它不象01层那样有下属的初等项。88层用到规定的条件名和复合条件。当层号小于10时可以按照一个数字来写。

对于与记录有关的数据的必要细分允许用层号作详细说明，它可以细分到较详细的数据标准为止。下例是每周的时间记录卡，利用图表的方式加以说明，内部分配的四个主要项目是：名字，雇员人数，日期和小时，再用较特殊的信息作为名字和日期出现。

```
                    ┌──LAST—NAME
                NAME┼──FIRST—INIT
                    └──MIDDLE—INIT
                EMPLOYEE—NUM
TIME—CARD                           ┌──MONTH
                WEEKS—END—DATE      ┼──DAY—NUMBER
                                    └──YEAR
                HOURS—WORKED
```

个用户定义名到一个执行选择名，如象PRINTER。一个助忆名的组成依照1.3节的规定。

1.9 字面

字面是一种常数，它在程序中与一个数据名不一样，但是它本身就有许多特殊的规定，一个字面不是非数值的就是数值的。

非数值字面

一个非数值字面必须受配对的引号或撇号限制，并可由 ASCII 表中的任何字符组成，当然要除去引号或撇号本身。由引号对括起来的所有空格作为字面的一部分。一个非数值字面的长度不能超120个字符。

下面是一些非数值字面的例子：

"ILLEGAL CONTROL CARD"

"CHARACTER—STRING"

"DO'S & DON'T' S"

一个非数值字面的每个字符（下面引入定义符）可以不同于任何字符的定义符。就是说，如果字面由撇号限制，然后引号（"）内可以是字面，反过来也是这样。一个非数值字面的长度应除去定义符；最小的长度是一个。

在一个字面里连续出现两个定义符，这如同在字面里单个定义符的出现一样解释。

非数值字面可以从这行"延续"到下一行。那时非数值字面的长度不包括上一行的编码表，下面的一些规定适用于编码的下一行（延续行）：

1.在延续行的第 7 列上设置一个连字符。

2.在延续区域B之前设置一个定义符。

3.在行末端以前的所有空格和在延续行中定义符后的所有空格及最后一个定义符以前的字面都被看作是字面的成份。

4.在任何一个延续行上，区域A应该是空白。

数值字面一个数值字面必须至少包含一位数同时又不能大于18位数字。一个数值字面可以包含字符 0 到 9（由可供选择的符号带头）和小数点。它可仅包含一个符号字符和小数点。符号是否出现，必须按照数值字面中最左边字符的情况而定。如果一个数值字面无符号，就假定它是正的。

除了最右边字符之外，一个小数点可以出现在数值字面的任何地方。如果一个数值字面不含有小数点，就认为它是一个整数。

下面是数值字面的例子：

72　+1011　3.14159　－6　－.333　0.5

通过环境部分使用 DECIMAL—POINT IS COMMA 的详细说明，字符句号和逗号的功能是互换的。使"EuroPean"实现内部表示。在这种情况下，"PI"当作一个数值字面的时候其值应是3.1416。

1.10 形象常数

形象常数是一种特殊类型的字面。它表示的标准数据名已分配了一个值。形象常数不需要被引号所括起来。

ZERO可以象数值字面那样用在一个程序的任何地方。别的形象常数对提供非数值数据是很有用的；这些保留字代表着如下不同的字符：

SPACE	空白字符由“八进制”40表示。
LOW—VALUE	它的“八进制”字符表示00。
HIGH---VALUE	它的“八进制”字符表示177。
QUOTE	引号，它的“八进制”表示42（在穿孔机中代码7—8）。
ALL字面	在一个或多个字面的情况下，那必须是一个非字数值字面字符或形象常数，在那些情况下ALL是多余的，但作为可读性保留。

这些形式的形象常数编译程序可以接收，但实际上是相同的。形象常数如同许多连字符的情况一样，应根据语句中上下文的需要而表示。

一个形象常数可以用于被称之为字面的任何一个地方，在“一般格式”中无论是除了字面对数据的限制，唯一容许的形象常数是ZERO。

1.11 一个程序的结构

每个COBOL源程序分成四个部分。每部分必须出现在它适当顺序的地方，而且每部分必须使用一个标头部分作为开始。

这四个部分排列的顺序和它们的作用是：

IDENTIFICATION DIVISION，有个程序名。

ENVIRONMENT DIVISION，在程序中指出计算机所使用的设备和特征。

DATA DIVISION，定义某些名字和对被处理数据的特征。

PROCEDURE DIVISION，那是由一些语句组成同时引导数据加工执行的次数。

如果是省略了标题部分或认为不重要的注解，那么想要看懂COBOL源程序代码是一件非常困难的事。在这种情况下，也许会产生不可预言的事件。

下面的轮廓体现了程序设计时规定程序成份的结构和顺序。

IDENTIFICATION DIVISION.

PROGRAM—ID. 程序名.
〔AUTHOR. 注解项…〕
〔INSTALLATION. 注解项…〕
〔DATE—WRITTEN .注解项…〕
〔DATE—COMPILED. 注解项…〕
〔SECURITY. 注解项 ..〕
ENVIRONMENT DIVISION.
〔CONFIGURATION SECTION.〕
〔SOURCE—COMPUTER. 项〕
〔OBJECT—COMPUTER. 项〕
〔SPECIAL—NAMES. 项〕
〔INPUT—OUTPUT SECTION.
FILE—CONTROL. 项…
〔I—O—CONTROL. 项…〕〕

DATA DIVISION.
〔FILE SECTION.
〔文件描述项
记录描述项…〕…〕
〔WORKING—STORGE SECTION.
〔数据项描述项…〕…〕
〔LINKAGE SECTION.
〔数据描述项…〕…〕
〔SCREEN SECTION.
〔屏幕描述项…〕…〕
PROCEDURE DIVISION 〔USING标识符1…〕.
〔DECLARATIVES.
〔节名 SECTION. USE句子.
〔段名.〔句子〕…〕…〕…
END DECLARATIVES.〕
〔〔节名 SECTION.〔程序段数〕〕
〔段名.〔句子〕…〕…〕…

1.12 程序编制规则

既然 Microsoft COBOL 是美国国家标准协会（ANSI）COBOL 的一个子集，程序设计可在标准的COBOL程序编制纸上书写，并且应按下面的规则进行。

1.每一行编码的1—6列上应该是一个6位的有序数，按照这样的穿孔卡片其序数是递增的，当然，在1—6列上也容许是空白。

2.作为部分，节和段头标使用的保留字必须在区域A（8～11列）中开始。过程名的出现也必须在区域A中（他们在什么地方有规定）。层号可以在区域A中出现。层号01，77和由“FD”表示的层必须在区域A中开始。

3.另外所有的元素应被限制在12～72列上，其它按语句强调的规定处理。

4.对73—80列上的东西编译程序且不管。通常，这些列只是对标识符的修饰。

5.说明性的注解可以被插入在一个源程序的任何行里面，但是它必须跟随在该行第7列上的星号之后。该行在源程序列表时能见到，但无其它的用途。在第7列上如果出现一个斜杠（/），你应连想到卡片，当编译程序列表源程序时，把他当作注解来处理，并且将被打印在新的一页的头上。

6.程序的任何元素可以“延续”到源程序的下一行上。对于一个非数值（“引用”）字面的延续规定已在1.9节中说明。任何其它的字或字面或别的程序元素由延续行的第7列位置上出现的连字符来延续。事实上被延续的连接字部分，除去延续行上第一个字和最后一个字之间的全部空格便是了。在一个延续行上，区域A必须是空白。

7.在一行中的任何制表字符除了第一个制表点应在第7列上之外，即需跳过前面的6位顺序数。以后的制表点好象每隔8列有个扩充制表点。其后面的制表点是17，25，33等等。这就已作为一般规则被确定。

1.13 名字的技巧

当一个数据名，条件名或段名不唯一时，以此可借助于对名的限制使用来达到唯一之目的。例如，有二个或更多的项名 YEAR，受限标准。

YEAR OF HIRE—DATE

也许在HIRE—DATE和 TERMINATION—DATE中区别于年当中的信息组。

受限由字OF或 IN 引导；连续的数据名或条件名的受限必须标明少量的层数，这样按照组合的标准包含了前面所有的名字，HIRE—DATE 必须是含有YEAR项的一个组项（或文件）。段名可由一个节名受限。

最大受限的数字对于段名是一个，对于数据名或条件名是五个。文件名和助忆名必须是唯一的。

一个受限名只可书写在屏幕节或过程部分中。对于段名增加一个规定，当被认为是相同节里面时不需要受限。

1.14 拷贝语句

拷贝语句用于在源代码输入过程中，Microsoft COBOL 编译程序磁盘文件而不是源文件的文本进行逻辑插入。COPY 语句的格式是：

COPY　　文本名

这个文本名是一个按照使用操作系统所需要格式的磁盘文件名。例如，假定 BDEF.—COB 一个文本文件，包含下面的源代码：

```
05  B.
    10  B1  PIC  X.
    10  B2  PIC  X.
```

然后，一个源文件包含

```
05  A.
    10  A1  PIC  9.
COPY BDEF.COB
05  C.
    10  C1  PIC  Z.
```

便精确地编译成如下的代码：

```
05  A.
    10  A1  PIC  9.
05  B.
    10  B1  PIC  X.
    10  B2  PIC  X.
05  C.
```

含有一个COPY 语句行的源程序部分必须从文本名的末端到该行的结尾都为空格。

第二章　标识和环境部分

2.1 标识部分

每个COBOL程序一开始就使用标题：

IDENTIFICATION DIVISION．这部分由预先指定的一些名被分配成为各个段：

PROGRAM—ID．　　程序名．
AUTHOR．　　注　释．
INSTALLATION．　　注　释．
DATE—WRITTEN．　　注　释．
DATE—COMPILED．　　注　释．
SECURITY．　　注　释．

只有PROGRAM—ID段是必需的，并且它必须是第一段。程序名是任何字母数字的字符串，开头的那个必须是字母。只有程序名的前6个字符被编译程序保留。程序名用来区别目标程序和包括编译列表时的标题。

另外一些段的内容就不太重要了，仅仅是作为文件的附注。

2.2 环境部分

环境部分规定了一种标准的表示方法，一个COBOL程序具有那些式样，这要依赖于一台专门计算机上面的物理特征。它在每个程序中都需要。

环境部分的一般格式是：

```
ENVIRONMENT DIVISION．
CONFIGURATION SECTION．
 SOURCE--COMPUTER．计算机名 [WITH DEBUGGING MODE ]．
OBJECT—COMPUTER．计算机名
    [MEMORY SIZE 整数 WORDS | CHARACTERS | MODULES ]
    [PROGRAM COLLATING SEQUENCE IS ASSCII] ．
SPECIAL—NAMES． [PRINTER IS 助忆名] ASCII

          IS { STANDARD—1 }
             { NATIVE     }

          [CURRENCY SIGN IS 字面]
          [DECIMAL—POINT IS COMMA]．
INPUT—OUTPUT SECTION．
FILE—CONTROL ．{文件控制项}…
 I—O—CONTROL ．
```