Condor Series 20/DBMS

Data Base Management System

User's Operation Manual

CONDOR COMPUTER CORPORATION

3989 RESEARCH PARK DRIVE
ANN ARBOR, MICHIGAN 48104

SEPTEMBER 1980

Table of Contents

Table of Contents (continued)

What is Condor Series 20/DBMS?

CONDOR SERIES 20/DBMS is the product of Condor Computer Corporation's years of research, development, and field testing of data base management system applications for micro computers.

CONDOR SERIES 20/DBMS is the data management system that simplifies informatio processing for inventory control, accounting functions, personnel reporting, and many other applications.

CONDOR SERIES 20/DBMS uses the relational data base concept. Data bases consist of many information files that work with one another to organize data according to the user's wishes. Once the data is organized, simple command procedures allow unrelated information to become dynamically related; a property of a relational data base system.

Advantages of Condor Series 20/DBMS

SPECIAL FEATURES/COMMANDS OF CONDOR SERIES 20/DBMS NOT USUALLY FOUND IN OTHER DATA BASE MANAGEMENT SYSTEMS

1. CONDOR SERIES 20/DBMS is a relationally structured data base management system. It is much easier to understand by the novice user than a network or hierchical system.

2. CONDOR SERIES 20/DBMS has a self contained data language. It does not require a host language. A professional computer programmer is not required to develop data bases.

3. CONDOR SERIES 20/DBMS's parent version, DBM-I, has been field tested for three years on Z-80 microcomputers. CONDOR SERIES 20/DBMS data structures are upward compatible with DBM-I's data structures.

4. Transaction processing oriented. Permits an efficient method for posting transactions to a master file. An audit trail (Result set) is generated each time a post is run. This can be used for keeping an ongoing record of master before transactions are posted.

5. Data Dictionary system - Provides the ability for different sub-schemas or views of the data base.

6. Data entry and Updating provides user with CRT screen format and setable editing criteria. For example, user must meet minimum and maximum values.

7. DEFINE command - permits creation of a new data base in minutes.

8. COMPUTE command - permits up to 32 term equation.

9. SORT command - fast and efficient using up to 32 keys.

10. SELECT command - selects records by specifying up to 32 logical conditions. Boolean operations AND/OR permitted.

11. Complete batch operation is permitted.

12. Extensive error checking and messages are provided.

Glossary of Terms

Alphabetic Data	Characters a to z, A to Z, "-", ".", "'"
Boot-up	Procedure, specific to a hardware configuration, for loading the operating system. In this manual CP/M is the operating system.
Data Base	A collection of disk files describing the screen format, data items, data item definitions, and data's allocated disk space.
Database1	A term used in CONDOR SERIES 20/DBMS command line syntax, indicating the "destination" data base designated in the COMPARE and POST commands. Example: POST Database1 Database2 MATCHING Key...
Database2	A term used in CONDOR SERIES 20/DBMS command line syntax, indicating the "source" data base designated in the COMPARE and POST commands. Example: POST Database1 Database2 MATCHING Key....
Data Definitions	Attributes of a data item, (i.e. Type of data, length in bytes, minimum and maximum values, and default values). Data Definitions are stored in a disk file and are associated with a data base.
Data Dictionary	A CONDOR SERIES 20/DBMS data base file containing records which describe the files associated with each data base defined.
Data Directory	The header record of ".DAT" files. The header contains the number of data items, bytes per record, and number of records in the data base.
Data Item (name)	The smallest unit of "moved" information. Also called a field. Examples: city, age, name, etc. A name designating a fixed amount of space allocated within a record for storing data. Item names are illustrated on the CRT screen format and are described to the computer by data definitions.
Data Size	Number of bytes of disk storage allocated to a data item.
Data Type	Specification in a data item definition indicating whether the data to be stored in a data field is Alphabetic (A type), Alphabetic-Numeric (AN Type), Numeric (N Type), Dollar ($ Type), or Date (J Type, where J indicates a Julian Date).
Data Value	Specific alphabetic characters, numbers, or special characters ("+", "-", ".", etc.) stored in the allocated disk space of a data item. Examples: New York, 59.56, John Doe, etc.

efault Value	Predefined data values, which will be entered in a data field while entering data if no other value is entered.
Edit Criteria	Predefined restrictions for entering data such as minimum/maximum values, default values, and data types. CONDOR SERIES 20/DBMS prohibits entry of data not meeting edit requirements and signals the user if errors occur.
Field	A specific amount of space allocated within a record.
File	Allocated space on a disk to be used for storage of data, data definitions, formats, batch commands, etc. A data base file is a collection of data base records.
Format	The process of "painting" a screen form on the CRT terminal for later data entry use.
Input	Refers to the process of entering data into a data base, entering data in a file, or describes a set of data which is to be or has been entered into the computer.
Julian	A numeric representation of a date where 01/01/00 (January 1, 1900) is 1 Julian. The date is entered in the form mm/dd/yy, mm-dd-yy, or mmddyy and output in the form mm/dd/yy. Within the computer, the Julian representation is used for processing and storage in a 3 byte data field.
Master/Transaction	Disk file qualifiers (master file, transaction file) used within this manual to clarify transaction processing. The master file is updated by a transaction file. For example, journal transactions update a general ledger.
Matching Field	Synonymous with "Matching Data Item". Refers to data item naming and data item definition requirements. A data item in one file be defined with the same name and data definition as in another file.
Record	A collection of data items. The contents of a record refers to the collection of data values stored in a record.
Schema	The primary structure of a data base. Describes inter-relationships of data items, including the association of the input screen format and item data definitions.
Sub-schema	Alternate views of the schema of data items, input screen format, and data definitions. For example, an application may have one format for data entry and a different format for output.

Chapter 1 Introduction

This manual is designed especially for first-time users of small computer systems. By following step-by-step procedures outlined in this manual, users can develop CONDOR SERIES 20/DBMS data base management applications using an English-like command language without assistance from a programmer.

The chapters in this manual use application examples to illustrate the features of CONDOR SERIES 20/DBMS commands. Before you begin, it is necessary to become familiar with CONDOR SERIES 20/DBMS command line syntax and other operating features which provide essential background in systems development.

CONDOR SERIES 20/DBMS COMMAND LINE DEFINITION

Instructions are given to the computer processor in the form of CONDOR SERIES 20/DBMS command lines. The command line generally consists of a verb-like command, data base for file name, and, in some commands, additional qualifying information. The qualifying information describes data base or file data items and logical/arithmetic operators for adding, comparing, assigning items, etc. The syntax for the command line is as follows:

```
Syntax:    COMMAND      (Dr:)Database/File    Qualifying Information

           ( ) indicate optional; / indicates alternate choice

Example:   UPDATE       B:EMPLOYEE           WHERE NAME IS W.B.SMITH
```

COMMAND:	One of the CONDOR SERIES 20/DBMS instructions described in Appendix A of this manual.
(Dr:)	Indicates Optional drive designation. In the example, data base EMPLOYEE resides on drive B.
Database/File	The *data base name* or *file name.ext*. (.ext is a three letter file name extension used to describe the type of file.)
	File name.ext describes a single disk file. The name should not exceed eight characters in length and the .ext three characters. The extensions of .DAT, .FRM, .DEF, .DIC, .DIR, and .HLP are standardized and will be described in other sections of this manual.
	Data base refers to a CONDOR SERIES 20/DBMS data base which is composed of a .FRM file (form definition), a .DEF file (item definitions), and .DAT file (data). Data base names should not exceed eight characters.
Qualifying Information	A user specified list of item names and operands which varies according to the CONDOR SERIES 20/DBMS COMMAND used. Options are described in Appendix A within the COMMAND

description. CONDOR SERIES 20/DBMS wild characters are extremely helpful in expressions. The wild character "*" allows you to shorten an item name such as ADDRESS to ADDR* in a command line. The wild character "?" instructs the computer to accept any value in the data item's corresponding position. If a data item is always stored with a person's first and second initials followed by a last name (For example: W.B.SMITH), the insruction ".....WHERE NAME IS ????SMI*" instructs the computer to search for all last names beginning with SMI.

APPLICATION DEVELOPMENT AND OPERATION

Steps in developing a data base using CONDOR SERIES 20/DBMS command lines include:

1. Designing data input format screen
2. Defining data characteristics of each data item in the screen format.
3. Defining the data file on disk.
4. Adding the data base to the dictionary.

The above steps are carried out in most applications by the FORMAT command where steps three and four are automatic. For sub-schemas: i.e., alternate input formats, data definitions, or data files are needed, the DEFINE command is required.

After development of the data base is completed, your file is ready for operation; that is, the input of data and the processing of reports.

The step-by-step development and operation procedures are as follows:

Startup..	Chapter 2
Data Base Development..................................	Chapter 3
Data Base Input...	Chapter 4
Other CONDOR SERIES 20/DBMS Features..........	Chapter 5

DESIGN CONSIDERATIONS

CONDOR SERIES 20/DBMS is a programming language and, as with other programming languages, there are rules within the language that must be followed. Prior to programming, the design of the system should be described in enough detail, and within CONDOR SERIES 20/DBMS rules, to ensure data bases and files are named correctly, items within the data bases are named and defined to meet CONDOR SERIES 20/DBMS command requirements, and procedures are designed which can meet the requirements of both CONDOR SERIES 20/DBMS and the new application.

CONDOR SERIES 20/DBMS SPECIFICATIONS

Number of records	32,767 records maximum per file
Record Size	2 to 1024 bytes per record. In each record the first byte is automatically assigned as a status byte.
Fields	1 to 127 fields per record
A or AN Data	1 to 127 bytes per field
Date or J Data	3 bytes per field
N or $ Data	1 to 10 digits (stored as a 1 to 4 byte integer)
CRT Screen Format	Must fit on 80 X 24 line video screen

CONDOR SERIES 20/DBMS DATA TYPES

AN - Alpha Numeric	a to z, A to Z, 0 to 9, symbols
A - Alpha	a to z, A to Z, space, " ' ", ".", "-"
N - Numeric	+ or -2148373647, integer
$ - Dollar	+ or -$21,483,736.47, 2 decimal places
J - Julian Date	Format mm/dd/yy, mm-dd-yy, or mmddyy for input Format mm/dd/yy for output
R - Required Entry	Used with any of the above data types

CONDOR SERIES 20/DBMS SYSTEM REQUIREMENTS

Z-80 Microcomputer with at least 48K RAM

CRT Terminal with cursor addressing and screen erase (e.g., ADDS, Beehive, Hazletine, Infoton, Lear-Seigler, Perkin-Elmer, Soroc, TRS 80-Model II)

CP/M operating system (version 1.4 or greater)

40K addressable space

Floppy and/or Hard Disk Drives

Line or character printer with formfeed (Forms length control is desirable)

Important naming conventions, reserved words, and matching fields are described below.

A. Naming Guidelines and Data Item Definitions for CONDOR SERIES 20/DBMS

1. File Name

Limitations: Eight Characters or Less in Length and required three-character extension.

Example: EMPLOYEE.FRM

In the example, EMPLOYEE.FRM file is stored on the diskette. The extension, .FRM, instructs the computer that it is a form file. Special extensions for CONDOR SERIES 20/DBMS are: .FRM (Form File), .DEF (Data Definitions File), .DAT (Data File), .DIC (Dictionary File), .DIR (Directory File), and .HLP (Help File). File names automatically default to certain extensions. For example, command FORMAT EMPLOYEE will automatically create a file called EMPLOYEE.FRM, and, if the option is to define a new data base, will automatically create the EMPLOYEE.DEF and EMPLOYEE.DAT files.

2. Data Base Names

Limitations: Eight Characters or Less.

Example: PURORDR for purchase order data base

A CONDOR SERIES 20/DBMS Database Name is not a file reference and, therefore, never has an extension. The data base name is used in CONDOR SERIES 20/DBMS commands to describe the data base files which consist of a format, definition, and data files for processing. Chapter Three addresses creating a data base.

It is a good practice to choose names that are easily remembered. Examples of easy-to-remember names:

PURCHASE - Purchase Order Data Base

GLEDGER - General Ledger Data Base

Because users also address *data item names* frequently, choose them with care. (Data item names are described in the following section.) Examples of CONDOR SERIES 20/DBMS commands highlighting data base and data item names are:

```
SELECT EMPLOYEE WHERE DATEHIRE IS GT 01/01/80
        (GT means Greater Than)

LIST GLEDGER BY ACCT DATE DEBIT CREDIT
```

3. Data Item Names

Limitations:	Fifteen Characters or Less.
Example:	PROMDATE for promised date
Definitions:	Each data item has a name, data definition, and reserved storage space within a data base. (The reserved storage space is called a "field") It is illustrated on the input screen and described by the data definition file. Examples of screen illustrations and data definitions required for each data item in the Systems Design are:

Screen Illustration: [ADDRESS]________________
Data Definition:
16 Characters underscored
AN Alphabetic/numeric type
16 Bytes allocated storage
0 minimum characters
16 maximum characters
No default value

Screen Illustration: [NAME]___________________
Data Definition:
19 Characters Underscored
A Alphabetic/numeric type
19 Bytes allocated storage
0 minimum characters
19 maximum characters
"New York" default value, example

Screen Illustration: [COST]_________
Data Definition:
9 Numbers underscored (2 Decimals)
$ Dollar Type
Entry must include decimal.
4 Bytes allocated storage
$0.00 minimum value
$999999.99 maximum value
No default value

Screen Illustration: [PROMDATE]________
Data Definition:
8 Underscores required
J Date (mm/dd/yy) Type
01/01/00 minimum date
12/31/99 maximum date
3 Bytes allocated storage
$TODAY assigns date entered in DATE command as default

Screen Illustration: [QUANTITY]______
Data Definition:
6 Numbers underscored
N Numeric Type
3 Bytes allocated storage
-99999 minimum value
999999 maximum value
No default

Note: The brackets ([]) are inserted around data item names when the input screen is being developed. When the screen is displayed for data entry or data listing the brackets are not shown.

4. Reserved Words

In the CONDOR SERIES 20/DBMS command line, certain words have special meaning. For example, the word ADD in the POST command instructs the computer to add. If the word ADD is used as an item name, it is highly probable the computer will misinterpret the name and perform an erroneous calculation. For this reason, words having special meaning to CONDOR SERIES 20/DBMS are designated as Reserved Words and their use is restricted, that is, data item names should not be Reserved Words. CONDOR SERIES 20/DBMS Reserved Words are:

ADD
AND
BY
EQ (Equal)
EQUAL
GE (Greater Than or Equal)
GT (Greater Than)
IS
LE (Less Than or Equal)
LT (Less Than)
MATCHING
NE (Not Equal)
NOT
OR
REPLACE
REP (Replace)
ST (Such That)
SUBTRACT
SUB (Subtract)
SUM
USING
WHERE
WHOSE

Any item name containing "*", "?", "#", "<", ">", or "=" may cause command interpretation problems.

Numeric or $ item names containing "*", "/", "+", or "-", may cause command interpretation problems.

Data item names should not contain blanks such as NEW DATE. The preferred data item name would be NEW.DATE.

5. Matching Fields:

There are two commands that reference two data bases in the command line. These are COMPARE and POST commands. In these commands, data items are compared and/or changed. Care must be taken in the design to ensure items to be compared and posted have the same name and identical data definitions.

For example, in a purchase order file there may be an item for quantity received. This item could be named RECEIVED and have a three byte numeric data definition. Another file may have been created to collect daily purchase order receipts where, once a day, they are summarized and posted to the purchase order file. For this to be possible, the item RECEIVED, with a three byte numeric data definition, must also exist in the receipts file for recording each receipts quantity.

CRT CONTROL KEYS

The control key, with one of the following letter keys depressed, allows special operations:

CONTROL C:	Aborts current operation where permitted.
CONTROL E:	In FORMAT, will save form. In UPDATE, will end revise mode. No movement of the cursor is required to end the record.
CONTROL O:	In data entry, overrides the edit criteria which checks for the minimum and maximum range. If the data entered does not agree with the declared field type, CONTROL O will not override the edit procedure.
CONTROL P:	Starts or stops output to the system printer and duplicates the CRT display on the printer. CONTROL P should not be used during data entry or update.
CONTROL Q:	Resumes output previously suspended by CONTROL S.
CONTROL R:	In form design mode, restores screen if clear button was accidently pressed. In data entry mode, restores screen format if accidently erased.
CONTROL S:	Temporarily suspends output to the CRT screen or printer until a CONTROL Q is entered.
RETURN:	(Carriage return, abbr. <C/R>) Terminates a command line.

CURSOR MOVEMENT KEYS IN FORMAT, DATA ENTRY, AND UPDATE MODE

RETURN:	Carriage return, line feed. When entering data, ends item's data entry and moves cursor to the beginning postion of next item.
CONTROL L:	Move cursor *forward*. When entering data, moves cursor to next data item.
CONTROL H:	Move cursor *back*. When entering data, moves cursor to previous data item.
CONTROL K:	Move cursor *up*. When entering data, moves cursor to previous data item.
CONTROL J:	Move cursor *down*. When entering data, moves cursor to next data item.

Chapter 2 Start-up

Initial start-up for CONDOR SERIES 20/DBMS requires transfering all CONDOR SERIES 20/DBMS commands and files into a diskette containing the CP/M operating system. After transfer, bootup, and initiate CP/M batch command SUBMIT START. All necessary procedures are described below.

CONDOR SERIES 20/DBMS COMMANDS USED IN THIS CHAPTER

DATE	Allows for entering current date
DBMS	Loads CONDOR SERIES 20/DBMS system
TERM	Computer requests video terminal type identification

CREATE A WORKING MASTER

At this time, turn on the computer, video terminal, and printer.

The first step is to backup your CONDOR SERIES 20/DBMS diskette. Save the CONDOR SERIES 20/DBMS diskette as your file master and use the *copy* for your *working* master.

The second step is to transfer CONDOR SERIES 20/DBMS commands and files into your CP/M diskette. The procedure for transfer is described below.

After this, it is not necessary to repeat the transfer procedure until another copy of CONDOR SERIES 20/DBMS is required.

The CONDOR SERIES 20/DBMS commands and files are now on a CP/M diskette and the CP/M diskette is ready for development of a CONDOR SERIES 20/DBMS application. The procedure for starting CONDOR SERIES 20/DBMS for applications development and operation is described below.

TRANSFERING CONDOR SERIES 20/DBMS COMMANDS TO CP/M DISKETTE

```
=================================== == .=======================================

      ACTION          RESPONSE  AND  COMMENTS
==============================================================================
```

1. Turn on power switch to the computer, disk drives, and video console.

2. Insert your CP/M diskette in drive A.
   a. Make sure your CP/M system is 48K or greater. If not, use MOVCPM command. (See your CP/M reference manual.)

3. Insert CONDOR SERIES 20/DBMS Diskette in Drive B.
    a. This diskette should be your working master (copy) of the CONDOR SERIES 20/DBMS diskette.

4. Bootup.
    a. Follow specific computer's procedure for bootup.

    b. System will respond with prompt (A).

5. Type PIP <C/R>
    a. Disk transfer program will be loaded into memory.

    b. System will respond with prompt (*).

6. Type A:=B:*.* <C/R>
    a. All the CONDOR SERIES 20/DBMS commands are transferred to the CP/M diskette. System will respond with prompt (*).

7. Type Carriage Return <C/R>
    a. PIP program is complete. Single <C/R> stops PIP.

    b. System will respond with prompt (A>).

8. Remove CONDOR SERIES 20/DBMS diskette from drive and store for safe keeping.

9. Transfer completed.
    a. CP/M diskette with CONDOR SERIES 20/DBMS commands is ready for starting the development of a CONDOR SERIES 20/DBMS application.

==

STARTING CONDOR SERIES 20/DBMS

==

ACTION   RESPONSE AND COMMENTS

==

1. Turn on power switch to the computer disk drives, video console, and printer.

2. Insert CONDOR SERIES 20/DBMS diskette in drive A and bootup.

3. Type SUBMIT START<C/R>
    a. Computer responds with version number, licensing information, and a READY message.

    b. You will be requested to enter "today's date".

4. Type date in mm/dd/yy format<C/R>
    a. Computer will request terminal ID.

5. Type one of following:
ADDS I BEEH I HAZL I INFO
LEAR I PERK I SORC I TRS2
MISC <C/R>

   a. Computer will respond with a II READY II message which must be centered on the screen. If it is not centered, the terminal is not correctly referenced. See TERM command for description for video terminal types.

==

START COMMANDS WITHOUT BATCH COMMAND

The above start procedure was performed in batch mode. The following commands demonstrate start-up procedures without batch mode. This procedure is slower and intended to *demonstrate only* the commands used in start.

==

ACTION RESPONSE AND COMMENTS

==

1. Insert CONDOR SERIES 20/DBMS disk in drive A and bootup.

2. Type DBMS<C/R>
   a. System displays start-up message: Licensing: CONDOR SERIES 20/DBMS vendor's address.

3. Type DATE<C/R>
   a. System displays date, and message to enter new date into system.

4. Type mm/dd/yy <C/R>
   a. mm/dd/yy is entered

5. Type TERM<C/R>
   a. Computer will request terminal ID.

6. Type one of following:
ADDS I BEEH I HAZL I INFO
LEAR I PERK I SORC I TRS2
MISC <C/R>

   a. Computer will respond with a II READY II message which must be centered on the screen. If it is not centered, the terminal is not correctly referenced. See TERM command for description of video terminal types.

==

WARM-UP EXERCISE FOR CONDOR SERIES 20/DBMS

After transfer and start procedures are performed, you are ready to create a CONDOR SERIES 20/DBMS data base.

The following warm-up exercise is intended to familiarize first-time users with the simplicity involved in creating a data base information file and the powerful performance in CONDOR SERIES 20/DBMS data base language. Follow the steps outlined below to create a data base containing names and birthdays.

Step 1: Creating Data Base.

Type FORMAT BIRTHDAY<C/R>
A blank screen is displayed for creating the format.

Type[NAME]________________[BIRTHDATE]________
Type 16 underscores (Caution!! Not hyphens.) for NAME, and 8 for BIRTHDATE.

Type <Control-E>
<Control-E> ends form creation and stores the BIRTHDAY.FRM file on disk. The system asks if the data base is to be defined.

Type Y
After a few moments, system will display 1:NAME, with cursor positioned after the comma. Enter the data definition.

Type AN<C/R>
You have instructed the computer that NAME is alpha numeric. Next, the system will display 2:BIRTHDATE, with cursor positioned after the comma. Enter the data definition.

Type J<C/R>
You have instructed the computer that BIRTHDATE is a Julian date item. Next, the system requests you to review data definitions and asks for confirmation.

Type Y
System will indicate "Busy" while it creates the data base. Attributes for the data base, number of items and record length, are printed. A message is printed indicating the data base is created, and a prompt is displayed. You are now ready to enter data.

Step 2: Entering Data

Type ENTER BIRTHDAY<C/R >
Format will be displayed. Type any name and a birthday in the format mm/dd/yy. Type C to enter another name and birthday. When you are finished entering names and dates, type E to end data entry.

Step 3: Trying Some CONDOR SERIES 20/DBMS Commands

Type SORT BIRTHDAY BY BIRTHDATE<C/R>

The computer will print messages indicating progress of the sort routine. Upon sort completion, the data base will be in ascending date sequence.

Type LIST BIRTHDAY BY NAME BIRTHDATE<C/R>

The computer will display the sorted data base on the video terminal.

Type PRINT BIRTHDAY BY NAME BIRTHDATE<C/R>

The computer will print a hard copy of the sorted data base. At this time, you should have a better understanding of CONDOR SERIES 20/DBMS screen formatting and input procedures required in data base creation. The following chapters describe in detail all the features of the language.

Chapter 3 Developing a Condor Series 20/DBMS Data Base

This chapter outlines development of files associated with CONDOR SERIES 20/DBMS data basecreation:

Screen Format File	(.FRM)	Describes data items and provides format for data input and output.
Data Item Definition File	(.DEF)	Contains data definitions for data items illustrated in format.
Data File	(.DAT)	Contains user data.
Data Dictionary	(.DIC)	Contains data base name and defines the .FRM, .DEF, and .DAT files.

Defining the above four files allows subsequent data entry, data storage, and data access applications. A brief description of data base file access will help you understand the development. The LIST command provides an example.

LIST EMPLOYEE BY NAME DEPARTMENT SENIORITY

In the above command line, the command refers to a data base name, EMPLOYEE, which is recorded in a record in the data dictionary (DATA.DIC). This record contains names of the data base files, i.e., form (.FRM), data definitions (.DEF), and data (.DAT). The LIST command combines these files, then proceeds with execution.

The data base is defined by two steps: .

1. Creating the screen format and data items using the FORMAT command.
2. Creating data item definitions file (.DEF), creating the data file (.DAT), and adding the data base name to the dictionary (.DIC). This is accomplished by the FORMAT or DEFINE command.

CONDOR SERIES 20/DBMS COMMANDS USED IN THIS CHAPTER

FORMAT	Allows you to define the CRT screen format (.FRM), and optionally create data item definitions (.DEF) and the data file (.DAT).
DEFINE	Allows you to create data item definitions (.DEF), allocate disk space for the user data file (.DAT), and update the data dictionary (.DIC).
LIST	Allows data base records to be displayed on the CRT screen.
PRINT	Allows data base records to be printed on the system printer.

DEFINING A CONDOR SERIES 20/DBMS DATA BASE

```
=================================================================
     ACTION        RESPONSE  AND  COMMENTS
=================================================================
```

1. Determine data base name

   a) The name may be equal to or less than eight alpha numeric characters. No extension entry is allowed.

   b) A simple general ledger accounting system will be used to describe CONDOR SERIES 20/DBMS command features. This application will consist of a general ledger data base, general journal data base, and a journal audit trail data base. Following are the names to be assigned to the three data bases.

      1. GLEDGER - General Ledger Data Base

      2. JOURNAL - General Journal Data Base

      3. JOURAUDT - Journal Audit Trail Data Base

   This chapter will describe the creation of these data bases. The following chapters will describe entering data into the data bases and accessing the data bases for accounting reports.

2. Determine drive on which to locate the system.

   a) Drive A contains the operating system and CONDOR SERIES 20/DBMS commands. As a result, space is limited and it is best to locate a system on another drive such as drive B. The command format for specifying a drive option will be Dr: .

   Standard: COMMAND Dr: data base name

   Example: FORMAT B:GLEDGER
   Note: There is no space following B:.

   b) In the general ledger accounting system example, the data base is to be located on drive B. This is accomplished by preceeding the data base name with the drive name, i.e., B:.

```
******************************************
****  DEVELOP GLEDGER DATA BASE  ****
******************************************
```

3. Type FORMAT B:GLEDGER<C/R>

   a) The system will respond "New File" unless you are updating an existing file.

   b) The system will display a blank screen for developing the format. If updating, the existing file will be displayed.

Refer to GLEDGER screen example below for developing the format.

```
****************************************************
*        GENERAL LEDGER ACCOUNTING SYSTEM          *
****************************************************
*                                                  *
*             GENERAL LEDGER RECORD                *
*                                                  *
*                                                  *
*[ACCOUNT]:   _____       [ASSET]:_  [LIABILITY]:_ *
*                                                  *
*[NAME]:      ________________                     *
*                                                  *
*[DEBIT]:     ________   [CREDIT]: ________       *
*                                                  *
*[YTDAMT]:    ______     (Year-To-Date Amount)     *
*                                                  *
*[MONTH]:     ______     (Last Accounting Month    *
*                        Posted to General Ledger) *
****************************************************
```

4. Type the format

   a) The cursor can be moved any place and in any direction on the screen. You may position titles, headings, and "data items" as desired. The "data items" must be enclosed in brackets ( ) and followed by the number of underscores needed to store the data item's data. (Remember, dates require eight underscores.) Example: [NAME]__________________

   b) Ensure all data items are formatted with correct number of underscores. All underscores for a given data item name must be consecutive. For example, [AMOUNT]___, ___, ___. ___ is not permissible but it is permissible to format [AMOUNT] $_________.

5. Type < CONTROL-E >

   a) This action will automatically enter contents of the screen format into the data base .FRM file.

      (If <CONTROL-C> is typed, the system will not write out the format. This is useful only if you do not wish to save the information or, if in the revise mode, no change is made.)

   b) The system will inquire if a new Data Base is to be defined. (If the format was being revised, this inquiry would not be made. In the example, GLEDGER is a new data base format.)

6. Type Y

   a) The system will automatically prompt you for definitions of data items inserted in brackets on the format. Items will appear in the order they appear on the screen format; from left to right, top to bottom.

7. Type appropriate data type <C/R> as shown in table below.

a) The data type abbreviations are as follows:

AN....Alphabetic-Numeric
A......Alphabetic
J.......Date
N......Numeric
$.......Dollar
R......Required

Table of Number of Bytes for N and $ data types	
1 BYTES	+/- 127 Max
2 BYTES	+/-32,767 Max
3 BYTES	+/-8,388,607 Max
4 BYTES	+/-2,147,483,647 Max

b) The system will assign values to data size, minimum, maximum. To enter default value or change the assigned min or max, perform steps 7.1 -7.3 below.

7.1 Type<C/R>

The system will respond with a " " prompt.

7.2 Type name of data item to be changed.

The system will re-display the name. The values assigned to the same name in step 7 should be visible for easy copying. Change max, min, and/or default as required.

7.3 Type new values

The system will respond with the next data name.

c) Upon completion of all names, the system asks for confirmation.

8. Type Y if the data item definitions are correct. Type N if a data item is incorrect.

a) If Y is typed the system will respond Busy and automatically allocate the data base name .DAT and add the data base name to the directory. If at this point, you wish to change the data base, the FORMAT and DEFINE command must be used. The screen is altered by FORMAT and the Data Definitions are altered by DEFINE in *redefine mode* mode.

b) The system will display the record size and number of items defined.

c) If N is typed, the system will respond with a " " prompt allowing you to correct the data item. Type the data item name needing correction, and correct its values.

9. Type ENTER B:GLEDGER <C/R> to enter test data.

   a) The format developed will be displayed on the screen. You may test data making sure to enter incorrect data; that is, alpha characters where numeric is required. The system will prompt you if errors occur.

   b) When you have entered all the test data, type E as prompted.

10. Type LIST B:GLEDGER<C/R >

   a) You will see the data added to the user data file. The system will display the format and data.

11. Type PRINT B:GLEDGER.FRM <C/R>

   a) The computer will print out the format.

12. Type PRINT B:GLEDGER.DEF BY F* T* S* MAX MIN D*

   a) F* = FIELD, T* = TYPE, S* = SIZE, and D* = DEFAULT.

   b) The computer will print out the data definitions.

   c) File the format and data definition printouts in system's manual.

```
*******************************************
**** DEVELOP JOURNAL DATA BASE *****
*******************************************
```

13. Type FORMAT B:JOURNAL<C/R >

   a) The JOURNAL Data Base is created by following previous actions 3 through 8, used in creating the GLEDGER data base. Change GLEDGER references to JOURNAL. Refer to JOURNAL screen example below for developing the JOURNAL data base.

   b) To test and document, follow actions 9 through 12 above. Change GLEDGER references to JOURNAL.

```
*************************************************************
*          GENERAL LEDGER ACCOUNTING SYSTEM                 *
*************************************************************
*                                                           *
*                   JOURNAL RECORD                          *
*                                                           *
*                                                           *
*   [ACCOUNT]:    _____                                     *
*                                                           *
*   [NAME]:       __________________                        *
*                                                           *
*   [DEBIT]:      ________  [CREDIT]: ________              *
*                                                           *
*   [MONTH]:      _______      (Accounting Month)           *
*                                                           *
*   [DATE]:       _______      (Date of Journal Entry)      *
*************************************************************
```

```
*******************************************
**** DEVELOP JOURAUDT DATA BASE ****
*******************************************
```

14. Type DEFINE B:JOURAUDT < C/R >

    a) The JOURAUDT Data Base is to be created with the same format and data definitions as the JOURNAL data base.

    b) System will indicate a .FRM file is not available and another .FRM file must be substituted.

15. Type B:JOURNAL <C/R>

    a) The system will inquire if an existing definitions file (.DEF) is to be used.

16. Type Y<C/R>

    a) The system will instruct the definition file name to be entered.

17. Type B:JOURNAL< C/R>

    a) The JOURAUDT data base is created. System responds with prompt.

    b) To test and document, follow actions 9 through 12 above. Change GLEDGER references to JOURAUDT.

The general ledger accounting system data bases are complete. The procedures in the next chapter describe:

1. Entering chart of accounts into the GLEDGER data base.
2. Entering journal entries into the JOURNAL data base.
3. Posting journal entries to the general ledger.
4. Accumulating journal entries for audit trail.

Chapter 4 DataBase Input Guidelines

In this chapter, three data input procedures are described. A two fold objective is served: 1. learning CONDOR SERIES 20/DBMS commands while 2. developing expertise in computer operating procedures.

Input can be defined as taking information external to the computer and, through some process, storing the information in a computer data base. For example, in Chapter Three, information about a general ledger account (account number, description, etc.) was typed into the computer through a video terminal and stored in a data base file on a magn diskette. (Please note: Input includes *input of new* data as well as *input of revisio* existing data.) The input process may have *restrictions.* For example, a payroll sys may be designed to allow only one payroll record per employee and, if another record is input it is to be eliminated.

The following outlines input guidelines for adding unique or matching records and posting records to data bases.

I. Entering "Unique Records" (Generally *master* records)

When entering new records to an existing data base, duplicate records may not be allowed. For example, in general ledger account records, two records having identical account numbers are not allowed. (This restriction is imposed by the particular application, and is not a general limitation of CONDOR SERIES 20/DBMS)

II. Entering "Matching Records" (Generally *transaction* records)

When entering new records to a "transaction" data base, there must be a "matching record" in the "master" base. For example, a payment from a client would be entered in accounts receivable data base with an account number which must match the account number of a record in the general ledger data base.

III. Posting (Generally transactions are posted to master records)

New records that have been entered into a data base are posted to records (update records) in another data base. The data base posted to (master data base) has unique records, and the data base with posting data (transaction data base) may have multiple records, though it is restricted to matching records in the master data base. One example is a purchase order receipt transaction that updates the purchase order master data base.

CONDOR SERIES 20/DBMS COMMANDS USED IN THIS CHAPTER

APPEND	Appends records of one data base to records of another data base.

COMPARE	Compares records of one data base file (Database1) and records of another data base file (Database2) by specified item(s) and creates a "result data base" of Database1 records.
COMPUTE	Computes specified items in a data base record and enters result in another specified field in the data base record.
EMPTY	Erases all data records of a data base.
ENTER	Enters new records into a data base.
POST	Matches records of one data base file (Database1) and records of another data base file (Database2) by specified item(s) and updates Database1 records from Database2 records for specified matching data items. Creates a "result data base" of Database1 records that matched Database2 records.
PRINT	Prints data base on the system printer according to specified items.
SELECT	Selects data base records meeting specified conditions and creates "result data base" of selected records.
SORT	Sorts data base according to specified data items.
STAX	Calculates statistics for specified data items.
TITLE	Prints report headings.
UPDATE	Changes existing records in a data base/file.

RESULT DATA BASES

Commands, COMPARE, POST and SELECT, create "result data base" files on the current drive. That is, if the current drive is B, result files $$.FRM, $$.DEF, and $$.DAT would be created on drive B. To retrieve these files, a data base entitled RESULT should be added only once to DATA.DIC on drive B, as follows:

Type ENTER B:DATA.DIC <C/R>

DATA.DIC input format will be displayed.

Type RESULT for title name, $$.DAT for data file, $$.FRM for form file, and $$.DEF for definition file. Type Control E and E to end.

Current drive is reassigned by typing B: <C/R>.

COMPARE	Compares records of one data base file (Database1) and records of another data base file (Database2) by specified item(s) and creates a "result data base" of Database1 records.
COMPUTE	Computes specified items in a data base record and enters result in another specified field in the data base record.
EMPTY	Erases all data records of a data base.
ENTER	Enters new records into a data base.
POST	Matches records of one data base file (Database1) and records of another data base file (Database2) by specified item(s) and updates Database1 records from Database2 records for specified matching data items. Creates a "result data base" of Database1 records that matched Database2 records.
PRINT	Prints data base on the system printer according to specified items.
SELECT	Selects data base records meeting specified conditions and creates "result data base" of selected records.
SORT	Sorts data base according to specified data items.
STAX	Calculates statistics for specified data items.
TITLE	Prints report headings.
UPDATE	Changes existing records in a data base/file.

RESULT DATA BASES

Commands, COMPARE, POST and SELECT, create "result data base" files on the current drive. That is, if the current drive is B, result files $$.FRM, $$.DEF, and $$.DAT would be created on drive B. To retrieve these files, a data base entitled RESULT should be added only once to DATA.DIC on drive B, as follows:

Type ENTER B:DATA.DIC <C/R>

DATA.DIC input format will be displayed.

Type RESULT for title name, $$.DAT for data file, $$.FRM for form file, and $$.DEF for definition file. Type Control E and E to end.

Current drive is reassigned by typing B: <C/R>.

If the number of characters entered for each item is equal to th number of underscores allocated in that data item, the cursor automatically move to next item position. If number of chara does not fill the number of underscores allocated, depress carrig return. This moves cursor to next item.

To enter default values, depress carriage return at the beginning of data item. Default value will be entered if defined, and the curso will move to the next item.

When all item information for a record has been entered, options ar displayed allowing for Revise (R), Continue (C), End (E), Print (P) Delete (D), or Abort (A).

Type C to enter another account. System will respond with a new GLEDGER.FRM input format. Type in item data as before.

Type E to end account number data entry. System will respond with a prompt.

UPDATE

UPDATE is needed only if an account number record which already exists in the master file, GLEDGER, requires changing. If such changes are required proceed as follows:

Type A:UPDATE GLEDGER<C/R>

System will request a search key.

Type ACCOUNT IS xxxxxx<C/R>
Where xxxxxx represents account number.

If record is found it will be displayed on the screen. Move cursor to desired position, type changes, and depress E to end. System will repeat the search condition again.

To end UPDATE procedure, respond to the search request with a carriage return. This ends update.

COMPARE

Type A:COMPARE GLTEMP GLEDGER NOT MATCHING ACCOUNT <C/R>

GLTEMP will be compared to GLEDGER for matching ACCOUNT numbers. GLTEMP records that do not match will be written to a "result data base" on drive B. Format and data item names in the result file are the same as GLEDGER.FRM.

TITLE

Type A:TITLE 'GENERAL LEDGER REPORTS',L,L, 'NEW ACCOUNT NUMBER ADDITIONS FOR ',DATE,L,L<C/R>

Above headings (Clauses enclosed in quotes) and date will be printed on every page of the NEW ACCOUNT NUMBER ADDITIONS report. The heading will be printed in two lines, and the L's will cause lines to be skipped.

PRINT

Type A:PRINT RESULT BY ACCOUNT NAME DEBIT CREDIT <C/R>

"Unique" new account numbers (those without matching account numbers in the master file) will be printed on the system's printer. NAME is the data item describing account name.

APPEND

Type A:APPEND GLEDGER.DAT $$.DAT<C/R>

New and unique general ledger account numbers will be appended t master (GLEDGER) file. The APPEND command attaches $$.DAT file to the GLEDGER.DAT file.

SORT

Type A:SORT GLEDGER BY ACCOUNT <C/R>

The GLEDGER data base is sorted in account number sequence.

TITLE

Type A:TITLE 'GENERAL LEDGER REPORTS',L,L, 'GENERAL LEDGER CHART OF ACCOUNTS DATE',DATE,L,L<C/R>

Above headings and date will be printed on every page of the GENERAL LEDGER CHART OF ACCOUNTS report.

PRINT

Type A:PRINT GLTEMP BY ACCOUNT NAME DEBIT CREDIT <C/R>

The GENERAL LEDGER CHART OF ACCOUNTS will be printed on the system's printer.

EMPTY

Type A:EMPTY GLTEMP<C/R>

Records in the GLTEMP data base will be eliminated.

Change Drive

Type A:<C/R>

Current drive is changed to A.

The above procedure completes the example for entering unique records. The next section of this chapter discusses entering matching records.

II. ENTERING MATCHING RECORD

Example: General Journal Input

In the general journal data base (JOURNAL) developed in Chapter Three, journal entries into the JOURNAL data base must match by account number records in the general ledger data base (GLEDGER). If a record entered into the JOURNAL data base does not match a record in the GLEDGER data base by account number, it is an error.

To ensure matching records, records in the journal data base are compared to records in the general ledger data base. Those records that do not match are corrected before the journal is posted to the general ledger.

A standard procedure for entering matching records using CONDOR SERIES 20/DBMS commands is illustrated below.

COMMAND	ACTIONS AND COMMENTS
Change Drive	Type B: <C/R> Current drive will be changed to B.
ENTER	Type A:ENTER JOURNAL <C/R> When entering journal transactions, the system will respond with JOURNAL.FRM input format. Type in new journal transactions.
SORT	Usage strategy: Sorting is not required nor done automatically by CONDOR SERIES 20/DBMS in order to use CONDOR SERIES 20/ DBMS commands. However, if files are sorted response time will improve by as much as 90% if the files are greatly out of sequence. Type A:SORT JOURNAL BY ACCOUNT <C/R> System will sort JOURNAL data base into account number sequence. Type A:SORT GLEDGER BY ACCOUNT <C/R> System will sort GLEDGER data base into account number sequence.
COMPARE	Type A:COMPARE JOURNAL GLEDGER NOT MATCHING ACCOUNT <C/R> JOURNAL records will be compared to GLEDGER records by matching ACCOUNT. JOURNAL records that do not match will be output to the "result" file. Format and data items will be same as JOURNAL.FRM.
TITLE	Type A:TITLE 'GENERAL JOURNAL REPORTS',L,L, 'JOURNAL EXCEPTIONS FOR ',DATE,L,L Above headings and date will be printed on every page of the JOURNAL EXCEPTIONS report.

PRINT	Type A:PRINT RESULT BY ACCOUNT NAME DEBIT CREDIT <C/R>
	Journal records that do not match a record in the GLEDGER data base will be printed. Determine the problem and update either GLEDGER or JOURNAL data base as required. If updating is required, it is necessary to restart with SORT commands above.
	Adding matching records will be complete when printout prints zero records; that is, all JOURNAL records match the GLEDGER data base. The next step is POSTING the JOURNAL to GLEDGER data base.
Change Drive	Type A:<C/R> Current drive is changed to A.

III. POSTING

Example: Posting Journal Entries to the General Ledger.

In the general ledger accounting system example, journal entries should be collected for a monthly period as predetermined by accounting periods. At the end of a month, journal entries are posted to the general ledger. After posting, journal entries are accumulated in an audit trail data base and the JOURNAL data base is emptied in preparation for the next month's entries.

A standard procedure for posting records using CONDOR SERIES 20/DBMS commands is illustrated below.

COMMAND	ACTIONS AND COMMENTS
Change drive	Type B:<C/R>
	Current drive is changed to B.
STAX	Type A:STAX JOURNAL BY DEBIT,CREDIT<C/R>
	Statistics are calculated for DEBIT and CREDIT which include the total amounts for DEBIT and CREDIT in JOURNAL data base. For a "trial balance" the total credit must equal the total debit. If these totals are not equal the error must be found and the journal file updated by typing UPDATE JOURNAL and entering the search condition... WHERE ACCOUNT IS xxxxxx <C/R>. (Refer to the UPDATE GLEDGER previously described.)
POST	Type A:POST GLEDGER JOURNAL BY ACCOUNT AND ADD DEBIT,CREDIT REPLACE MONTH<C/R>
	For each matching ACCOUNT, DEBIT and CREDIT amounts in JOURNAL data base are added to DEBIT and CREDIT amounts in GLEDGER data base. MONTH in GLEDGER record is replaced by MONTH in JOURNAL record.

COMPUTE	Type A:COMPUTE GLEDGER ST YTDAMT = DEBIT - CREDIT <C/R> Year-to-date amount (YTDAMT) is calculated by subtracting CREDITS from DEBITS.
SELECT	Type A:SELECT GLEDGER ST ASSET IS Y<C/R > General ledger records that are asset accounts will be written to the "result data base". Format will be same as GLEDGER.FRM.
TITLE	Type A:TITLE''GENERAL LEDGER MONTHLY REPORTS',L,L, 'ASSET ACCOUNTS REPORT FOR ',DATE,L,L Above headings and date will be printed on every page of the ASSETS ACCOUNTS REPORT.
PRINT	Type A:PRINT RESULT BY ACCOUNT NAME DEBIT CREDIT YTDAMT <C/R > Asset general ledger records will be printed on the printer along with their descriptions, debit and credit amounts, and the year-to-date amount.
STAX	Type A:STAX B:RESULT BY YTDAMT<C/R > Statistics are calculated for YTDAMT data item in RESULT data base. Since the RESULT data base contains asset records at this moment, STAX will calculate and print the total YTDAMT amount for asset account records. This amounts is to be compared to the liability STAX amount calculated below.
SELECT	Type A:SELECT GLEDGER ST LIABILITY IS Y<C/R> General ledger records that are liability accounts will be written to the "result data base". Format will be same as GLEDGER.FRM.
TITLE	Type A:TITLE 'GENERAL LEDGER MONTHLY REPORTS',L,L, 'LIABILITIES ACCOUNTS REPORT FOR ',DATE,L,L Above headings and date will be printed on every page of the LIABILITIES ACCOUNTS REPORT.
PRINT	Type A:PRINT RESULT BY ACCOUNT NAME DEBIT CREDIT YTDAMT<C/R> Liability general ledger records will be printed on the printer along with their descriptions, debit and credit amounts, and the year-to-date amount.

STAX	Type A:STAX B:RESULT BY YTDAMT <C/R> Statistics are calculated for YTDAMT data item in RESULT data base. Since the RESULT data base contains liability records at this moment, STAX will calculate and print the total YTDAMT amount for liability account records. This amount is to be compared to the asset STAX amount calculated above. If year-to-date amounts are not equal, the general ledger data base is in error. Correcting entries must be entered into the JOURNAL data base and the JOURNAL posted to the GLEDGER data base as described above. Prior to any adjusting entries ensure that the JOURNAL data base is emptied as described below.
APPEND	Type A:APPEND JOURAUDT.DAT JOURNAL.DAT <C/R> APPEND accumulates JOURNAL entries for audit trail.
EMPTY	Type A:EMPTY JOURNAL<C/R> EMPTY deletes all records in a data base and sets record count at zero. JOURNAL data base is reset and ready for another input cycle.
Change drive	Type A:<C/R> Current drive is set to A.

Chapter 5 Other Condor Series 20/DBMS Features

Chapter Five will demonstrate the HELP Command and batch command procedures, using the General Ledger Accounting System as an example. By the end of the chapter, all CONDOR SERIES 20/DBMS commands will have been demonstrated.

CONDOR SERIES 20/DBMS COMMANDS USED IN THIS CHAPTER

HELP	Causes computer to display an instruction and/or message menu on video terminal, allowing the user to select a procedure or answer a question.

HELP SCREENS AND BATCH COMMANDS

HELP screens and batch commands simplify computer operations by easing the task of remembering procedures and typing commands. The first time through a procedure it is good policy to proceed step-by-step through commands, as was done in Chapter Four; but, after debugging, it is a good practice to create HELP and batch files. Procedure errors can be greatly reduced.

HELP COMMANDS

HELP is a CONDOR SERIES 20/DBMS command to display on the screen of the video terminal a menu of options which describes programs and procedures to select when operating a computer system.

The menu is easily displayed by typing HELP helpname. The computer will retrieve a file name helpname.HLP and display the contents on the screen. The operator chooses a number and types in the number followed by a carriage return. The computer will execute the associated program or procedure. HELP files can be created with system editor or by the CONDOR SERIES 20/DBMS FORMAT command.

BATCH COMMAND PROCEDURES

Batch Command Procedures allow many commands to be grouped and stored in a "batch" file in a diskette. At a later time, they may be retrieved and executed by typing the "batch command" or by selecting an option in a HELP file.

Batch commands are given to the computer by submitting the batch file to the batch command processor, e.g., in CP/M by typing SUBMIT batchname<C/R>. The computer will retrieve a batch file named batchname.SUB and will execute the commands that have been previously typed into the file.

HELP AND BATCH COMMAND EXAMPLES

The General Ledger Accounting System example used in Chapter Four will also provide examples for this chapter. For example, Users, when operating the General Ledger Accounting System, would be greatly aided if the following video screen was displayed by typing HELP RESTART C/R . (Please note the name RESTART was used since it is very easy to remember, though any name of eight or less alphabetic/numeric letters may have been used.)

```
*********************************************************
*        GENERAL LEDGER ACCOUNTING SYSTEM               *
*********************************************************
*                                                       *
*          SELECT ONE OF FOLLOWING OPTIONS              *
*                                                       *
*     1.    INPUT CHART OF ACCOUNTS                     *
*                                                       *
*     2.    UPDATE GENERAL LEDGER                       *
*                                                       *
*     3.    ENTER JOURNAL ENTRIES                       *
*                                                       *
*     4.    JOURNAL TRIAL BALANCE                       *
*                                                       *
*     5.    POST GENERAL LEDGER                         *
*                                                       *
*********************************************************
```

From the above menu, one of five options can be selected by typing a number followed by a carriage return. The system will execute the HELP or BATCH command that is associated with the number typed.

The HELP or Batch command associated with a number is defined when the helpfile is created. Defining the associated command is accomplished by formatting the command in brackets ([]) following the number. Note, in the above video display, the brackets do not appear on the video screen. In the following example, when a video screen is being formatted to create a helpfile, you will see that bracketed commands follow each number. The HELP command automatically removes bracketed commands from the video display at time of execution.

The next section describes creating a help file. As you will see this file is created approximately the same way as a .FRM file.

CREATING HELP MENU

The help file is created by a system editor. The file that is created must have an extension of .HLP.

EDIT	Type ED B:RESTART.HLP< C/R>
	Type I<C/R>. The cursor may be positioned to center the following headings and lines on the screen.

```
*************************************************************
*            GENERAL LEDGER ACCOUNTING SYSTEM               *
*************************************************************
*                                                           *
*            SELECT ONE OF FOLLOWING OPTIONS                *
*                                                           *
*    1.    INPUT CHART OF ACCOUNTS[SUBMIT B:ACCNTS]         *
*                                                           *
*    2.    UPDATE GENERAL LEDGER[SUBMIT B:UPGLEDG]          *
*                                                           *
*    3.    ENTER JOURNAL ENTRIES [SUBMIT B:JOURNIN]         *
*                                                           *
*    4.    JOURNAL TRIAL BALANCE [SUBMIT B:TRIALB]          *
*                                                           *
*    5.    POST GENERAL LEDGER[SUBMIT B:POSTGL]             *
*                                                           *
*************************************************************
```

Type<CONTROL -Z> and E<C/R> to end formatting of RESTART menu and creates RESTART.HLP file.

System will write out RESTART.HLP file to diskette B.

System will end ED command and display prompt.

<u>CREATING BATCH COMMAND FILES</u>

The batch command file is created by the system editor. The file that is created must have an extension of .SUB. To illustrate batch command file creation, batch command files referenced in above helpfile (RESTART) will be developed in this section. Batch command files to be created are:

1.	ACCNTS.SUB	Entering Chart of Accounts
2.	UPGLEDG.SUB	Updating General Ledger
3.	JOURNIN.SUB	Entering General Journal
4.	TRIALB.SUB	General Journal Trial Balance
5.	POSTGL.SUB	Posting Journal to General Ledger

Beginning with ACCNTS.SUB, perform the following procedure to create batch command file.

CREATING ACCNTS.SUB BATCH COMMAND FILE

EDIT Type ED B:ACCNTS.SUB<C/R>

Type the following:

```
ENTER B:GLTEMP<C/R>
COMPARE B:GLTEMP B:GLEDGER NOT MATCHING ACCOUNT <C/R
TITLE 'GENERAL LEDGER REPORTS'L,L, 'NEW ACCCOUNT NUMBER
ADDITIONS DATE ',DATE,L,L <C/R>
PRINT RESULT BY ACCOUNT NAME DEBIT CREDIT<C/R>
APPEND B:GLEDGER.DAT $$$$.DAT<C/R>
COMPARE B:GLTEMP RESULT NOT MATCHING ACCOUNT <C/R>
TITLE 'GENERAL LEDGER REPORTS',L,L,'DUPLICATE ACCOUNT
NUMBERS DATE ',DATE,L,L <C/R>
PRINT RESULT BY ACCOUNT NAME DEBIT CREDIT < C/R>
SORT B:GLEDGER BY ACCOUNT<C/R >
TITLE 'GENERAL LEDGER REPORTS',L,L, 'GENERAL LEDGER
CHART OF ACCOUNTS DATE ',DATE,L,L<C/R>
PRINT B:GLEDGER BY ACCOUNT NAME DEBIT CREDIT<C/R>
EMPTY B:GLTEMP< C/R>
HELP B:RESTART<C/R >
```

The file ACCNTS.SUB will be written to the B diskette. To execute type SUBMIT B:ACCNTS.SUB<C/R>.

Note: $$.DAT is always typed as $$$$.DAT in CP/M batch files.

By the same process all procedures in the General Ledger Accounting System may be batched as illustrated below.

CREATING UPGLEDG.SUB BATCH COMMAND FILE

EDIT Type ED B:UPGLEDG.SUB<C/R>

Type the following:

```
UPDATE B:GLEDGER <C/R>
SORT B:GLEDGER BY ACCOUNT<C/R>
TITLE 'GENERAL LEDGER REPORTS',L,L, 'GENERAL LEDGER
CHART OF ACCOUNTS DATE ',DATE,L,L<C/R>
PRINT B:GLEDGER BY ACCOUNT NAME DEBIT CREDIT <C/R>
HELP B:RESTART <C/R>
```

The file UPGLEDG.SUB will be written to the B diskette. To execute type SUBMIT B:UPGLEDG<C/R>.

CREATING JOURNIN.SUB BATCH COMMAND FILE

EDIT

Type ED B:JOURNIN.SUB<C/R>

Type the following:

```
ENTER B:JOURNAL <C/R>
SORT B:JOURNAL BY ACCOUNT <C/R>
SORT B:GLEDGER BY ACCOUNT <C/R>
COMPARE B:JOURNAL B:GLEDGER NOT MATCHING ACCOUNT <C/
TITLE 'GENERAL LEDGER REPORTS',L,L, 'JOURNAL EXCEPTIONS
FOR DATE ' ,DATE,L,L<C/R >
PRINT RESULT BY ACCOUNT NAME DEBIT CREDIT< C/R>
HELP B:RESTART <C/R>
```

The file JOURNIN.SUB will be written to the A diskette. To execute type SUBMIT B:JOURNIN <C/R>.

CREATING TRIALB.SUB BATCH COMMAND FILE

EDIT

Type ED B:TRIALB.SUB <C/R>

Type the following:

```
TITLE 'GENERAL LEDGER REPORTS',L,L,'JOURNAL TRIAL
BALANCE FOR DATE ',DATE,L,L< C/R >
STAX B:JOURNAL BY DEBIT CREDIT [P] <C/R >
TITLE 'GENERAL-LEDGER REPORTS',L,L,'GENERAL LEDGER
TRIAL BALANCE FOR DATE ',DATE,L,L <C/R>
STAX B:GLEDGER BY DEBIT CREDIT [P] <C/R>
HELP B:RESTART < C/R>
```

The file TRIALB.SUB will be written to the A diskette. To execute type SUBMIT B:TRIALB<C/R>.

CREATING POSTGL.SUB BATCH COMMAND FILE

EDIT

Type ED POSTGL.SUB<C/R>

Type the following:

```
POST B:GLEDGER B:JOURNAL BY ACCOUNT AND ADD DEBIT,
CREDIT AND REPLACE MONTH<C/R>
COMPUTE B:GLEDGER ST YTDAMT = DEBIT - CREDIT<C/R>
SELECT B:GLEDGER ST ASSET IS Y<C/R>
TITLE 'GENERAL LEDGER REPORTS',L,L,'ASSETS ACCOUNT
REPORT FOR DATE ',DATE,L,L<C/R>
PRINT RESULT BY ACCOUNT NAME DEBIT CREDIT YTDAMT <C/R>
STAX RESULT BY YTDAMT [P] <C/R>
SELECT B:GLEDGER ST LIABILITY IS Y<C/R>
TITLE 'GENERAL LEDGER REPORTS',L,L,'LIABILITIES ACCOUNT
REPORT FOR DATE ',DATE,L,L<C/R>
PRINT RESULT BY ACCOUNT NAME DEBIT CREDIT YTDAMT <C/R>
STAX RESULT BY YTDAMT[P] <C/R>
APPEND B:JOURAUDT.DAT B:JOURNAL.DAT <C/R>
EMPTY B:JOURNAL <C/R>
HELP B:RESTART <C/R>
```

The file POSTGL.SUB will be written to the B diskette. To execute type SUBMIT B:POSTGL<C/R>.

The HELP and Batch commands for the general ledger accounting system are completed. Test the system by typing HELP B:RESTART and selecting each option.

This Chapter completes the examples of CONDOR SERIES 20/DBMS commands. CONDOR SERIES 20/DBMS has many features, therefore Appendix A, CONDOR SERIES 20/DBMS Commands, should be studied carefully. With experience, CONDOR SERIES 20/DBMS can be the programming language for many of your applications.

APPENDIX A

CONDOR SERIES 20/DBMS COMMAND DEFINITIONS

TABLE OF CONTENTS

COMMAND DESCRIPTION AND USAGE

THE DATA BASE MANAGER COMMANDS REQUIRE THAT THE DATA BASE OPERATING SYSTEM INTERFACE (DBOS) BE IN MEMORY. DBOS CAN BE LOADED BY TYPING "DBMS". THE DATA BASE OPERATING SYSTEM RESIDES IN A DIFFERENT AREA OF MEMORY THAN THE DISK OPERATING SYSTEM.

THE USER SHOULD ALWAYS RECORD THE CURRENT DATE AT STARTUP TIME. (SEE DATE COMMAND.) THE OPERATOR IS ALWAYS PROMPTED ON THE CONSOLE SCREEN WITH THE CHARACTERS "X>", WHERE X IS A DRIVE LETTER (A- D) ON WHICH THE USER IS CURRENTLY LOGGED ON TO. FOR EXAMPLE, IF THE PROMPT IS "A>", THE USER IS LOGGED ON TO DRIVE "A". TO LOG ON TO ANOTHER DRIVE TYPE THE DRIVE LETTER, A COLON, AND CARRIAGE RETURN, E.G., "B:<C/R>". TYPICALLY THE USER MAY LOG ON TO ANOTHER DRIVE WHERE MORE DISK SPACE IS AVAILABLE.

IN THE FOLLOWING COMMANDS THE GENERAL SYNTACTICAL FORM, DESCRIPTION, AND EXAMPLES ARE GIVEN.

1. WHEN BRACKETS ([]) ARE USED THEY MEAN THAT THE CHARACTER OR WORD IS OPTIONAL. WHEN PARENTHESES () FOLLOW A COMMAND, THE LETTERS ARE NOT PART OF THE COMMAND WORD AND ONLY GIVE THE READER THE DERIVATION THEREOF. TYPE BRACKETS ONLY IF ENCLOSED IN BRACES ({}).

2. WHERE APPLICABLE, COMMANDS CAN ADDRESS UP TO 32 OPERANDS. OPERANDS ARE DATA ITEM NAMES (FORM LABELS), AND CONSTANTS. DELIMITERS, LOGICAL OR ARITHMETIC OPERATORS, COMMAND NAMES, OR FILENAMES DO NOT COUNT AS OPERANDS. FOR EXAMPLE, A 32 KEY SORT CAN BE INITIATED BY THE COMMAND "SORT CLIENTS BY NAME,RACE,...KEY 32".

3. LABELS MUST BE REFERENCED AS THEY APPEAR IN THE BRACKETS OF THE FORM FILE (.FRM). THEY MAY BE ABBREVIATED BY THE * OR ? SYMBOLS. HOWEVER WHEN USING THE "COMPUTE" COMMAND, THEY MUST BE SPELLED OUT IN FULL.

4. OPERANDS MUST BE LESS THAN 16 CHARACTERS IN LENGTH WHEN DEFINED IN THE COMMAND LINE. OPERANDS WITH EMBEDDED BLANKS (E.G., JOHN SMITH) MUST BE ENCLOSED IN SINGLE OR DOUBLE QUOTES IN ORDER TO BE INTERPRETED AS ONE OPERAND (E.G., 'JOHN SMITH'). THIS RULE DOES NOT APPLY TO DATA ENTERED ON A FORM. DATA-BASE TITLES MUST BE REFERENCED BY NO MORE THAN 8 CHARACTERS.

5. ALL COMMAND LINES ARE LIMITED TO 128 CHARACTERS

6. UPPER AND LOWER CASE LETTERS ARE ALWAYS TRANSLATED TO UPPER CASE FOR SEARCH AND COMPARE OPERATIONS. HENCE, THEY MAY BE ENTERED EITHER WAY. HOWEVER, THEY ARE DISPLAYED AS ENTERED.

7. WORK OR RESULT SETS ARE CREATED UNDER THE NAME "$$.EXT". THEY WILL ALWAYS BE CREATED ON THE DRIVE THAT THE USER IS CURRENTLY LOGGED ONTO. A BLANK DISK CAN BE INSERTED INTO AN UNUSED DRIVE AND SELECTED AS THE CURRENT DRIVE IF MORE WORK SPACE IS DESIRED. THESE FILES CAN BE RENAMED IF THEY ARE TO BE SAVED FOR LATER USE. IN CP/M BATCH PROCEDURES, FILENAMES OF THE FORMAT "$$.EXT" MUST BE TYPED AS "$$$$.EXT" SO THAT THE "$" SYMBOL WILL NOT BE INTERPRETED AS A BATCH PARAMETER.

8. THE SYSTEM FILES, DEF.DIC, DEF.FRM, DIC.FRM, ETC. ALWAYS LIE ON DRIVE A. THE SYSTEM WILL ONLY SEARCH DRIVE A FOR THESE FILES REGARDLESS OF THE CURRENT DRIVE.

9. CONDOR SERIES 20/DBMS WILL NOT SEARCH OTHER DICTIONARIES ON OTHER DRIVES IN THE SYSTEM WHENEVER THE TITLE IS NOT FOUND IN THE DATA DICTIONARY OF THE CURRENT DRIVE.

SUMMARY OF COMMANDS

COMMAND	DESCRIPTION
APPEND	Appends two data bases
COMPARE	Compares two data bases on selected fields and creates result file
COMPUTE	Computes and stores result in a data base
DATE	Displays or records date
DBMS	Loads CONDOR SERIES 20/DBMS system
DEFINE	Creates a new data base
EMPTY	Deletes all data records and resets directory counter to zero
ENTER	Enter new records into file
FORMAT	Formats video screen
HELP	Assist user with procedures
LIST	Display records of file in sequential order
POST	Updates database1 with values from database2
PRINT	Print records of file in sequential order
SELECT	Selects data base records meeting logical condition
SORT	Sort and rewrite file
STAX	Provides statistics on min,max,average and total
TERM	Defines system terminal
TITLE	Prints headings on every page
UPDATE	Change values in file meeting logical condition

CONDOR SERIES 20/DBMS COMMAND SYNTAX

```
-command ----------syntax ----------------------------------------------

APPEND   :APPEND database1 database2

COMPARE  :COMPARE database1 database2 [not] [matching] field1
          field2...field32

COMPUTE  :COMPUTE database st fr = f1 opr f2 opr ...f31

DATE     :DATE [mm/dd/yy]

DBMS     :DBMS

DEFINE   :DEFINE database [*]

EMPTY    :EMPTY database [OK]

ENTER    :ENTER database

FORMAT   :FORMAT formfile[.frm]

HELP     :HELP file[.hlp]

LIST     :LIST database [by field1 field2 ...field32]

POST     :POST database1 database2 [matching] key-fields OPR fields

PRINT    :PRINT database [by field1 field2 ...  field32]

SELECT   :SELECT database [where condition]

SORT     :SORT database by field1 field2 ....field32

STAX     :STAX database by field1 field2 ....field32

TERM     :TERM type ('type' is a four letter terminal ID)

TITLE    :TITLE 'text enclosed in literals or quotes',T,L,R,DATE

UPDATE   :UPDATE database [where condition]
```

[ ] means optional, | | means choice of one

APPEND

PURPOSE: TO APPEND ONE DATA BASE'S RECORDS TO ANOTHER DATA BASE'S RECORDS.

SYNTAX: APPEND DATAFILE1 DATAFILE2

WHERE DATAFILE2 RECORDS ARE APPENDED TO DATAFILE1 RECORDS

EXAMPLES:

```
APPEND B:PARENT C:DAILYINT
APPEND B:DEFSET.DEF B:PARTDEF.DEF
APPEND B:CLIENTS.DAT $$.DAT
```

USAGE: IN MOST APPLICATIONS, DAILY INTAKE RECORDS WOULD BE APPENDED TO THE PARENT FILE. NOTE THAT THIS COMMAND REQUIRES THAT BOTH FILES HAVE IDENTICAL RECORD SIZES AND ITEM NUMBERS AS SHOWN IN THEIR RESPECTIVE FILE DIRECTORIES. OTHERWISE, THE APPEND WILL BE ABORTED. EITHER DATA BASE NAMES OR DATA FILE NAMES MAY BE SPECIFIED IN THIS COMMAND. ALSO, FILES WITH EXTENSIONS OF .DEF AND .DIC MAY BE APPENDED TO EACH OTHER. WHEN THE APPENDED RECORDS CANNOT FIT ON ONE DISKETTE (I.E., NO SPACE LEFT), AN ERROR MESSAGE WILL BE ISSUED. THE APPEND WILL TERMINATE, RESULTING IN NO RECORDS APPENDED TO THE PARENT FILE. THE DRIVE MUST ALWAYS BE SPECIFED IF NOT THE CURRENT DRIVE.

COMPARE

PURPOSE: COMPARES TWO DATA BASES ON SELECTED FIELDS AND CREATES A RESULT SET

SYNTAX: COMPARE DATABASE1 DATABASE2 [NOT] [MATCHING] F1,F2,...,F32

DELIMITER SYNONYMS: "USING", "BY","MATCHING"

RESULTANT SET: $$.DAT, $$.DEF, $$.FRM

EXAMPLES:

```
COMPARE ORDERS TRANSACT NOT MATCHING ACCOUNT
COMPARE ORDERS SHIPPING MATCHING ACCOUNT AND SHIPPER
COMPARE ORDERS OLDORDER USING ACCOUNT
COMPARE ORDERS OLDORDER ACCOUNT
```

USAGE: THE COMPARE COMMAND COMPARES FIELDS WHICH ARE DEFINED IN THE TWO FILES HAVING THE SAME FIELD TYPE AND SIZE. DEPENDING ON THE SYNTAX, MATCHING OR NON-MATCHING RECORDS FROM DATABASE1 ARE WRITTEN TO $$.DAT.

FOR EXAMPLE, THE COMPARE COMMAND WILL ALLOW THE USER TO TAKE A MASTER FILE SUCH AS A CUSTOMER LIST AND COMPARE IT WITH RECORDS IN A DETAIL FILE, SUCH AS TRANSACTIONS, AND FIND ALL RECORDS IN THE MASTER FILE WHICH DO NOT HAVE TRANSACTIONS ASSOCIATED WITH THEM (E.G., "COMPARE ACCOUNT TRANSACT NOT MATCHING ACCTNO"). THE NON MATCHING ACCOUNT RECORDS ARE WRITTEN TO A DATA FILE, $$.DAT. THE OUTPUT FILES ARE CREATED ON THE CURRENT DISK DRIVE.

NOTE: 'COMPARE ACCOUNTS TRANSACT USING ACCTNO' IS NOT THE SAME AS 'COMPARE TRANSACT ACCOUNTS USING ACCTNO'. THE FIRST DATA BASE IS BEING RESTRICTED BY THE SECOND DATA BASE. HENCE THE RESULT SETS WILL NOT NECESSARILY BE THE SAME.

COMPUTE

~~R~~POSE: CALCULATES A QUANTITY AND STORES NUMERIC RESULT INTO A ~~TA~~ BASE.

YNTAX: COMPUTE DATABASE ST R = A OPR B OPR C...
OR: COMPUTE DATABASE ST A OPR B OPR C... =R

HERE A,B,C... ARE OPERANDS WHICH CAN BE INTEGERS (CONSTANTS) OR TEM NAMES (VARIABLES) AND R IS THE RESULTANT ITEM NAME IN WHICH RESULT IS STORED. THESE MAY BE UP TO 32 TERMS INCLUDING THE T ITEM.

PR ARE THE FOLLOWING ARITHMETIC OPERATORS:

+ ADDITION
- SUBTRACTION
* MULTIPLICATION
/ DIVISION

EXAMPLES:

```
COMPUTE ORDERS ST SUBTOTAL=QUANTITY*PRICE
COMPUTE ORDERS ST PROFIT=PRICE-COST
COMPUTE ORDERS ST DISCOUNT=PRICE/4+12.00
```

SAGE: THE COMPUTE COMMAND IS USED TO CALCULATE A QUANTITY AND TORE THE RESULT IN A NUMERIC ITEM OF EACH RECORD IN THE DATA SE. USING THIS COMMAND PROPERLY REQUIRES A CAREFUL DERSTANDING OF THE SYNTAX; OTHERWISE, UNEXPECTED RESULTS MAY BE BTAINED.

HE RESULT DATA ITEM MUST BE ALREADY DEFINED IN THE DATA BASE ECORD. THE TERMS IN THE EXPRESSION ARE EVALUATED FROM LEFT TO IGHT AS THEY APPEAR IN THE EQUATION. THERE IS NO PRIORITY GIVEN O OPERANDS OR ITEMS IN PARENTHESES. PARENTHESES AND SPACES ARE PTIONAL AND IGNORED, HOWEVER THEY MAY BE USED FOR MAKING THE XPRESSION MORE READABLE. FOR EXAMPLE, THE EXPRESSIONS BELOW ARE DENTICAL:

```
PRICE=7-4*AMMOUNT+2              (1)
PRICE=(7-4)*AMOUNT+2             (2)
PRICE=7-(4*AMOUNT)+2             (3)
PRICE = 7 - 4*AMOUNT +2          (4)
PRICE = 7 - 4 * (AMOUNT+2)       (5)
```

THE ABOVE EXPRESSIONS ARE ALL EVALUATED IDENTICALLY AS ILLUSTRATED IN (2), I.E., FROM LEFT TO RIGHT.

THE TERMS IN THE EXPRESSION ARE IDENTIFIED BY THEIR FIRST CHARACTER. IF IT IS A DIGIT (0-9), THE TERM IS CONSIDERED A CONSTANT AND WILL BE INTERPRETED AS AN INTEGER. IF THE FIRST CHARACTER IS A LETTER (A-Z), THE TERM IS CONSIDERED A VARIABLE AND THE CONTENTS OF THE FILE INTEPRETED AS A INTEGER. SINCE THE MULTIPLICATION SIGN (*) IS ALSO A WILD CHARACTER SYMBOL, ITEM NAMES ARE REQUIRED TO BE SPELLED OUT AS THEY APPEAR IN THE FORM OF THE DATA BASE. IF THE ITEM NAME HAS EMBEDDED BLANKS OR ANY OF THE CHARACTERS (+,=,/), THE NAME MUST BE ENCLOSED IN QUOTATION MARKS OR LITERALS SO IT CAN BE TREATED AS A SINGLE NAME. FOR EXAMPLE THE ITEM NAME ENROLLED-DATE MUST BE ENTERED AS 'ENROLLED-DATE' OR "ENROLLED-DATE", OTHERWISE IT WOULD BE TREATED AS TWO TERMS, ENROLLED MINUS DATE. A MINIMUM IF THREE OPERANDS AND TWO OPERATORS INCLUDING THE EQUALS SIGN IS CONSIDERED A VALID EQUATION.

THE VARIABLE TERMS REFERENCED IN THE EXPRESSION MUST ONLY BE NUMERIC, DOLLAR, OR DATE DATA TYPES. IF A CONSTANT IS TO BE USED IN A COMPUTATION INVOLVING THE ADDITION OR SUBTRACTION OF A DOLLAR DATA TYPE, BOTH DECIMAL POSITIONS MUST BE USED. DATES ARE STORED IN JULIAN, SO FOR EXAMPLE, ENTER BIRTHDATE-30 NOT BIRTHDATE-1/1/80, NOTE: THE LATTER WOULD BE COMPUTED AS 1 DIVIDED BY 1 DIVIDED BY 80. TWO VARIABLE DATES CAN BE SUBTRACTED SINCE THEY ARE ALREADY IN JULIAN. FOR EXAMPLE "DATEIN -DATEOUT" WOULD GIVE THE NUMBER OF DAYS DIFFERANCE IN JULIAN.

IF THE RESULT OF THE COMPUTATION IS A NUMBER GREATER THAN FOUR BYTES OR THE DEFINED RESULT ITEM SIZE, AN OVERFLOW ERROR MESSAGE WILL BE LISTED, AND THE RESULT WILL NOT BE STORED. ARITHMETIC IS PERFORMED AS INTEGER ARITHMETIC AND FRACTIONS FROM DIVISION ARE TRUNCATED.

ALL OPERANDS ARE NORMALIZED TO A FOUR BYTE INTEGER VALUE AND THEN OPERATED UPON BY THE OPERATORS. THIS MEANS THAT TWO DIFFERENT ITEM SIZES MAY BE USED IN THE ARITHMETIC. FOR EXAMPLE A NUMBER IN A TWO BYTE ITEM MAY MULTIPLY A NUMBER IN A 1 BYTE ITEM. HOWEVER, THERE ARE RESTRICTIONS IN THE MULTIPLICATION AND DIVISION OPERATIONS. THESE OPERATIONS MUST NOT PRODUCE INTERMEDIATE RESULTS GREATER THAN FOUR BYTES. FOR EXAMPLE IN THE EXPRESSION:

A*B/C=D

A*B MUST NOT EXCEED FOUR BYTES, EVEN THOUGH AFTER DIVIDING BY C IT MAY PRODUCE A RESULT THAT FITS IN FOUR BYTES. IF THESE RESTRICTIONS ARE VIOLATED, AN OVERFLOW ERROR WILL OCCUR AND THAT COMPUTATION WILL NOT BE PERFORMED.

THE COMPUTE COMMAND CAN ALSO BE USED IN AN INTERACTIVE MODE BY ISSUING THE COMMAND "COMPUTE DATABASE". THE EQUATION CAN BE ENTERED AFTER THE PROMPT, THE COMPUTATIONS WILL BE PERFORMED, AND THE RESULTS STORED. WHEN COMPLETED, THE OPERATOR WILL BE PROMPTED FOR ANOTHER EQUATION.

THE USER SHOULD REMEMBER THAT DOLLAR ITEMS ARE STORED AS 100 TIMES THE DOLLAR VALUE I.E, IN CENTS. HENCE, THESE OPERANDS MUST BE IN THE RANGE OF $-21,474,836.48 TO $21,474,836.47.

CAUTION MUST BE EXERCISED IN USING THIS COMMAND BECAUSE OF ITS GLOBAL EFFECTS.

DATE

PURPOSE: THIS COMMAND ALLOWS THE USER TO DISPLAY AND/OR CHANGE THE CURRENT SIGN-ON DATE.

SYNTAX: DATE [CURRENT-DATE]

```
EXAMPLES:         DATE
                  DATE 4/1/79
```

USAGE: FOLLOWING THE COMMAND DATE, THE CURRENT DATE IS DISPLAYED AND THEN A PROMPT FOR A NEW DATE WILL OCCUR. A C/R WILL LEAVE THE DATE UNCHANGED. IF THE CURRENT DATE IS SPECIFIED IN THE COMMAND LINE THEN IT IS RECORDED AS THE CURRENT DATE WITH NO PROMPTING. DATES MAY BE ENTERED IN THE FORMATS mm/dd/yy, mm-dd-yy, OR mmddyy AND WILL BE STORED IN JULIAN AS A 2 BYTE BINARY NUMBER. NEW FILES CREATED DURING THE SESSION WILL HAVE THE CURRENT DATE STORED IN THEIR DIRECTORY. THE DATE OF CREATION OF A DATA FILE CAN BE DISPLAYED BY LISTING THE DIRECTORY OF THE FILE, E.G., "LIST CLIENTS.DIR"

DBMS

PURPOSE: LOADS THE DATA BASE MANAGER OPERATING SYSTEM (DBOS) INTO MEMORY.

SYNTAX: DBMS

EXAMPLE: DBMS

USAGE: THE DBMS O/S (DBOS) MUST BE IN MEMORY IN ORDER FOR ALL THE DBMS COMMANDS TO BE DISPATCHED. EXECUTING DBMS.COM LOADS DBOS. LOADING DBOS CLEARS DATE AND SHOULD BE RESET.

NORMALLY THE DBMS COMMAND IS EXECUTED AUTOMATICALLY AT START UP, AND THE OPERATOR SHOULD NOT REPEAT THE COMMMAND. THE USER SHOULD LOG ON AND RECORD THE CURRENT DATE AFTER DBMS HAS BEEN LOADED.

NOTE: THAT A 48K OR MORE DOS IS REQUIRED FOR DBOS. ONCE DBOS IS IN MEMORY THE USER HAS 32K OF USER SPACE IN WHICH TO LOAD PROGRAMS AND DATA. DBOS CAN NEVER BE WRITTEN OVER BY ANOTHER PROGRAM RUNNING UNDER DOS AS IT DEFINES THE APPARENT USER SPACE AS 32K. HENCE, THE USER CAN EXECUTE FORTRAN, BASIC, AND COBOL PROGRAMS IN BATCH AND THEN RETURN TO A DBMS COMMAND.(THIS ASSUMES THAT THE HIGHER LEVEL LANGUAGE PROGRAMS DO NOT REQUIRE MORE SPACE THAN 32K. IF THEY DO, AN ERROR MESSAGE WILL BE LISTED AND THE PROCEDURE ABORTED. THE PROGRAMMER SHOULD REDUCE THE SIZE OF THE PROGRAM SO THAT IT RUNS IN A SMALLER SPACE.)

DEFINE

PURPOSE: CREATES NEW DATA BASE, RE-DEFINES A DATA BASE, ERASES A DATA BASE

SYNTAX: DEFINE DATABASE {*}

EXAMPLES:

```
DEFINE CLIENTS
DEFINE INVENTRY
DEFINE QUICKDB *
```

USAGE: THE DEFINE COMMAND CREATES A DATA BASE BY CREATING A DATA DEFINITION FILE (.DEF), A DATA FILE (.DAT) AND WRITING A NEW RECORD TO THE DATA DICTIONARY (.DIC). THE DEFINE PROGRAM USES AS ITS INPUT A PRE-DEFINED FORM (SEE FORMAT COMMAND). IT'S FROM THIS FORM, THE ACTUAL DATA BASE IS CREATED. THE FORM NAME EXPECTED IS THE SAME AS THE DATA BASE NAME. IF IT IS NOT FOUND, THE USER IS PROMPTED FOR ANOTHER FORM NAME. FOR EXAMPLE, IF THE COMMAND "DEFINE CLIENTS" IS ISSUED, THE DEFINE PROGRAM LOOKS FIRST FOR A FILE CALLED "CLIENTS.FRM". IF IT IS NOT FOUND THE USER CAN ENTER ANOTHER FORM NAME. WHENEVER THE FORM NAME IS PROMPTED, ANOTHER OPTION IS ALSO PROMPTED. THIS OPTION ASKS IF AN EXISTING DEFINITION FILE IS TO BE USED, THEREBY NOT REQUIRING A NEW SET OF DATA ITEM DEFINITIONS. IF THE USER SPECIFYS "N", A NEW DEFINITION FILE IS CREATED AND DATA ITEM DEFINITIONS MUST BE SUPPLIED BY THE OPERATOR. IF THE USER SPECIFYS A "Y", THE DEFINITION FILE NAME MUST BE ENTERED. THE PROGRAM WILL REJECT ANY DEFINITION FILE WHOSE FIELD COUNT IS NOT EQUAL TO THE NUMBER OF FORM LABELS.

IF THE "*" IS USED IN THE COMMAND, ALL DATA ITEMS WILL BE DEFINED AS ALPHANUMERIC(AN). THE USER CAN CHANGE THESE TO OTHER DATA TYPES IF DESIRED BEFORE WRITING THE DEFINITION FILE.

IF THE DATA BASE NAME EXISTS ALREADY, THEN THE USER CAN EITHER RE-DEFINE THE DATA BASE OR ERASE IT COMPLETELY. IF THE USER ELECTS TO RE-DEFINE, THEN NO RECORDS CAN BE IN THE DATA FILE. THE USER IS PROMPTED TO EMPTY THE DATA FILE IF IT HAS RECORDS. ONCE A DATA BASE IS ERASED, IT CAN BE DEFINED AS A NEW DATA BASE.

ONCE A DATA BASE HAS BEEN DEFINED, THE USER CAN UPDATE THE EDIT CRITERIA AT ANY TIME. THIS SHOULD BE DONE BY UPDATING THE DEFINITION FILE ONLY. (E.G., UPDATE CLIENTS.DEF WHERE FIELD IS 3 4 5).

IF THE USER WISHES A HARD COPY OF THE LABELS AND DEFINITIONS, DO A CONTROL P BEFORE ENTERING THE COMMAND. THIS WILL SEND A DUPLICATE OF THE CONSOLE SCREEN TO THE PRINTER.

EMPTY

PURPOSE: DELETES ALL RECORDS OF A DATA BASE AND RESETS DIRECTORY RECORD COUNT TO ZERO.

SYNTAX: EMPTY DATABASE [OK]

EXAMPLES:

```
EMPTY TRANSACT
EMPTY B:CLIENTS
EMPTY $$.DAT
EMPTY $$.DEF
```

USAGE: THE EMPTY COMMAND WILL DELETE ALL RECORDS OF THE DATA BASE AND RESET THE DIRECTORY RECORD COUNT TO ZERO. NOTE: THE RECORDS ARE NOT RECOVERABLE. EITHER DATA BASE NAMES OR DATA FILE NAMES MAY BE SPECIFIED IN THIS COMMAND. THE DATA FILENAME AND DRIVE MUST BE EXPLICITLY SPECIFIED IN THIS COMMAND, IF NOT THE CURRENT DRIVE.

THE USER IS REQUESTED TO GIVE A CONFIRMATION BEFORE THE FILE RECORDS ARE DELETED BY TYPING "OK" THEN A C/R. IF ONLY A C/R IS ENTERED, THE RECORDS ARE PRESERVED AND THE PROCEDURE IS ABORTED. THE COMMAND CAN ALSO BE GIVEN IN A BATCH PROCEDURE BY GIVING THE CONFIRMATION OF "OK" FOLLOWING THE DATA BASE. A SPACE MUST DELIMIT THE DATA BASE FROM THE CONFIRMATION. THIS COMMAND IS USEFUL FOR TRANSACTION PROCESSING. FILES WITH EXTENSIONS .DEF AND .DIC MAY ALSO BE EMPTIED.

ENTER

PURPOSE: THIS COMMAND ALLOWS THE USER TO ENTER NEW RECORDS INTO A FILE.

SYNTAX: ENTER DATABASE

```
EXAMPLES:           ENTER CLIENTS
                    ENTER CLIENTS.DEF
                    ENTER DATA.DIC
```

USAGE: THE ENTER COMMAND PUTS THE SYSTEM INTO DATA ENTRY AND THE SYSTEM CHECKS FOR CONSISTENCY BETWEEN THE ENTERED DATA AND THE EDIT CRITERIA DEFINED IN THE DEFINITION FILES. SHOULD AN INCONSISTENCY OCCUR, THE CONSOLE WILL BEEP AND THE ERROR INDICATED AT THE BOTTOM OF THE SCREEN. THE CURSOR WILL AUTOMATICALLY RESET TO THE BEGINING OF THE CURRENT ITEM. THE USER MUST CORRECT THE ERROR TO MEET THE EDIT CRITERIA IN ORDER TO PROCEED.

DATA ENTERED WHOSE LENGTH FILLS THE ITEM WILL AUTOMATICALLY BE POSITIONED TO THE NEXT ITEM, OTHERWISE A C/R WILL POSITION THE CURSOR TO THE NEXT ITEM. ENTERING A CONTROL H WILL MOVE THE CURSOR BACKWARDS ONE SPACE OR ENTERING A CONTROL L WILL MOVE THE CURSOR FORWARD ONE SPACE.

IF THE USER WISHES TO ABORT THE DATA ENTRY MODE WHILE FILLING IN THE FORM, A CONTROL C MAY BE TYPED AND THE CURRENT RECORD WILL NOT BE SAVED. ANY PREVIOUS RECORDS ENTERED THAT SESSION WILL BE SAVED. NORMALLY, THE USER WILL EXIT FROM THE ENTER ROUTINE USING THE "SAVE-END (E)" OPTION.

IF AFTER ENTERING A NEW RECORD, THE USER HAS BEEN SIGNALLED THAT THERE IS NO SPACE LEFT ON THE DISKETTE OR THE FILE IS FULL, A NEW FILE SHOULD BE STARTED ON A NEW DISKETTE. THE NEW DISKETTE SHOULD BE INSERTED INTO THE DRIVE AND LOADED BY TYPING "CONTROL C". CHOOSE A DIFFERENT FILENAME FOR THE NEW FILE, (E.G. TRANS-2.DAT). THE ITEM COUNTS AND RECORD SIZE MINUS ONE SHOULD BE SET INTO THE NEW DIRECTORY USING THE NUMBERS FROM THE PREVIOUS FILE'S DIRECTORY NOTE: WHEN GIVEN A "NO SPACE LEFT ON DISKETTE" MESSAGE, THE LAST RECORD ENTERED MUST BE RE-ENTERED ON THE NEW DISKETTE.

FILES MAY BE CONTINUED UNTIL THEY REACH 32,767 RECORDS, AT WHICH TIME A NEW FILE WITH ANOTHER NAME MUST BE CREATED.

FORMAT

PURPOSE: CREATES OR REVISES A FORM

SYNTAX: FORMAT [DRV:]FILENAME[.FRM]

```
EXAMPLES:          FORMAT NEWFORM.FRM
                   FORMAT NEWFORM
                   FORMAT C:OLDFORM.FRM
                   FORMAT OLDFORM
```

USAGE: THE USER IS DISPLAYED "New File" IF A NEW FILE IS BEING CREATED, OTHERWISE THE FILE WILL BE EDITED. ONLY FILES WITH AN EXTENSION OF ".FRM" CAN BE FORMATTED. USE THE "UPDATE" COMMAND TO MAKE CHANGES TO DATA FILES. THE USER CAN DECIDE AT ANYTIME DURING REVISION PROCESS TO KEEP THE ORIGINAL AND NOT USE THE CURRENT REVISION. HOWEVER, IF IT IS DESIRED TO KEEP THE ORIGINAL AS WELL, BEFORE REVISING, MAKE A COPY OF THE ORIGINAL USING A NEW NAME AND FORMAT THE COPY. IF THE FORM IS ON ANOTHER DRIVE THAN THE CURRENT DRIVE, THE DRIVE MUST BE SPECIFIED. NB

IF A NEW FILE IS BEING FORMATTED, THE USER WILL BE PROMPTED WITH AN OPTION TO DEFINE A NEW DATA BASE. IF THE "Y" IS PRESSED, THE "DEFINE" PROGRAM IS AUTOMATICALLY LOADED AND RUN (SEE DEFINE COMMAND).

NOTE: UNLESS EXPLICITLY SPECIFIED, THE DRIVE THE NEW FILE WILL RESIDE ON WILL BE THE CURRENT DRIVE.

HELP

PURPOSE: TO ASSIST USER WITH PROGRAM OPERATIONS

SYNTAX: HELP [HELP-FILE][.HLP]

EXAMPLE:
```
HELP
HELP BILLING
```

USAGE: THE HELP COMMAND ASSISTS THE USER IN INITIATING PROGRAMS BY SIMPLY TYPING THE NUMBER OF THE DESIRED FUNCTION DISPLAYED ON THE CONSOLE SCREEN. THE COMMAND IS QUITE USEFUL WHERE THE OPERATOR DOES NOT KNOW OR CARE TO KNOW THE NAMES OF THE DATA BASES INVOLVED IN THE OPERATIONS, THE SYNTAX, OR ITEM NAMES.

FUNCTIONALLY, THE HELP COMMAND READS A "HELP" FILE WHOSE FORMAT IS DESCRIBED BELOW AND DISPLAYS THIS FILE ON THE CONSOLE. THE NAME "SYSTEM.HLP" IS USED AS THE DEFAULT NAME OF THE HELP FILE WHEN NONE HAS BEEN SPECIFIED.

THE HELP FILE IS CREATED WITH THE SYSTEMS TEXT EDITOR OR BY USING "FORMAT XX" . IF USING THE FORMAT PROGRAM, RENAME THE FILE TO AN ".HLP" EXTENSION. A HELP FILE HAS THE FOLLOWING FORMAT:

```
DESCRIPTION 1 [COMMAND LINE 1]
DESCRIPTION 2 [COMMAND LINE 2]
      .
      .
DESCRIPTION N [COMMAND LINE N]
```

NOTE THE COMMAND LINES ARE ENCLOSED IN THE SQUARE BRACKETS. ONLY THE COMMAND DESCRIPTIONS ARE DISPLAYED AFTER CALLING THE "HELP" COMMAND. A NUMBER THAT IS ENTERED, CORRESPONDING TO THE nTh BRACKETED STATEMENT, WILL INITIATE THAT COMMAND STATEMENT ACCORDING TO THE ACTUAL SYNTAX BETWEEN THE BRACKETS. NUMBERS BEGIN WITH 1 AND CAN BE AS LARGE AS 127. ONE HELP MAY ALSO CALL ANOTHER HELP, OR A BATCH PROCEDURE. ALL PROGRAM FILES WITH AN EXTENSION OF ".COM" MAY BE CALLED. COMMANDS WHICH ARE INTRINSIC TO THE OPERATING SYSTEM CANNOT BE CALLED (e.g. 'ERA','TYPE','DIR').

AN EXAMPLE OF A HELP FILE WHICH HANDLES FINANCIAL ACCOUNTING UNDER THE NAME "ACCOUNTS.HLP" MAY BE CONSTRUCTED AS FOLLOWS:

```
              HELP ACCOUNTS

 1 REVIEW OF  ACCOUNTS ON FILE [PRINT  ACCOUNT BY NAME,ADDRESS]
 2 ACCOUNTS RECEIVABLE [HELP ACCREC]
 3 ACCOUNTS PAYABLE [HELP PAYABLES]
```

NOTE THAT THIS HELP CAN CALL THE OTHER HELP FILES,"ACCREC.HLP" AND "PAYABLES.HLP",

LIST|PRINT

PURPOSE: TO LIST OR PRINT A DATA BASE IN SEQUENTIAL ORDER

SYNTAX: LIST|PRINT DATABASE [BY FIELD1,FIELD2,...FIELD32]

EXAMPLES:

```
LIST CLIENTS
PRINT CLIENTS
PRINT CLIENTS BY NAME, RACE AND SEX
LIST CLIENTS BY NAME AND ADDRESS
```

USAGE:BY SPECIFYING FIELD LABELS (USING THE "BY" CLAUSE), THE FILE MAY BE LISTED IN COLUMNAR FORMAT. DELIMITERS MAY BE A COMMA, ONE OR MORE SPACES OR THE CONJUNCTION "AND" (DELIMITED BY A SPACE ON BOTH SIDES). IF NO FIELD LABELS ARE SPECIFIED, THEN THE ENTIRE RECORD WILL BE LISTED USING THE FORM DEFINED IN THE FORM FILE(.FRM).

A FILE MAY BE PRINTED RECORD AFUER RECORD IN SEQUENTIAL ORDER BY USING PRINT. IN WHICH CASE, THE ENTIRE FILE WILL BE PRINTED. IF THE FILE IS BEING PRINTED IN COLUMNAR FORMAT A CARRIAGE RETURN-LINE FEED PAIR IS ISSUED TO THE PRINTER AFTER EACH RECORD. WHEN PRINTING THE ENTIRE RECORD, A FORM FEED (ASCII 0CH) IS ISSUED FOLLOWING EACH RECORD. THE SYSTEM PRINTER FORMS LENGTH CAN BE SET SO THAT SEVERAL RECORDS CAN FIT ON A PAGE AND NONE PRINTED OVER THE PAPER PERFORATION. THIS IS ESPECIALLY USEFUL FOR PRINTING MAILING LABELS OR ON PRE-PRINTED FORMS. PRINTING CAN BE ABORTED BY ISSUING A CONTROL S FOLLOWED BY A CONTROL C. WHEN USING LIST, THE OPERATOR WILL BE PROMPTED WITH LIST OPTIONS AFTER EACH RECORD IS DISPLAYED, ONE OF WHICH IS TO PRINT A SINGLE RECORD BY TYPING P.

LISTING THE ENTIRE RECORD WITH ITS FORM WILL LEFT JUSTIFY FIELDS DECLARED NUMERIC, DOLLAR, OR DATE. IF THERE IS NOT ENOUGH PRINTING SPACES ALLOCATED ON THE FORM, AN ERROR MESSAGE WILL BE ISSUED TO INDICATE THAT FIELD xx DOES NOT HAVE ENOUGH UNDERSCORES. THE FORM SHOULD BE REFORMATTED IN THIS CASE AND ADDITIONAL UNDERSCORES ADDED. THE USER SHOULD ALLOCATE THE PRINTING SPACES IN ACCORDANCE WITH THE DECLARED FIELD TYPE AND THE FIELD SIZE.

WHEN LISTING THE FILE IN COLUMNAR FORMAT, EACH FIELD IS ALLOCATED A MINIMUM OF 12 PRINTING SPACES, WITH AN UPPER LIMIT DETERMINED BY THE DECLARED FIELD SIZE AND TYPE. SINCE NO NUMERIC FIELD CAN BE GREATER THAN 4 BYTES, NUMBERS WILL ALWAYS BE GIVEN 12 PRINTING SPACES. NUMERIC, DOLLAR, AND DATE FIELDS ARE RIGHT JUSTIFIED. ALPHABETIC AND ALPHANUMERIC FIELD TYPES ARE ALSO LEFT JUSTIFIED. LABELS ARE ALLOCATED A MINIMUM OF 12 PRINTING SPACES, WITH AN UPPER LIMIT SET BY THE DECLARED FIELD SIZE. THE LABEL IS LEFT OR RIGHT JUSTIFIED IN ACCORDANCE WITH THE FIELD TYPE EXPLAINED ABOVE. NOTE WHILE THE DATA WILL ALWAYS BE PRINTED IN FULL, THE LABEL MAY GET CHOPPED OFF WHEN THE DECLARED FIELD SIZE IS LESS THAN 13 AND THE LABEL EXCEEDS 12 CHARACTERS.

POST

PURPOSE: UPDATE A MASTER DATEBASE FROM A DETAIL DATA BASE

SYNTAX:
```
        POST DATABASE1 DATABASE2 [MATCHING] KEY1,KEY2,...
[OPR FIELD3,FIELD4,... OPR FIELD5,FIELD6,...,FIELD32]
```

WHERE OPR CAN BE ANY OF THE FOLLOWING OPERATORS:

REP, REPLACE, ADD, SUM, SUB, SUBTRACT

RESULT SET: $$.DAT, $$.DEF, $$.FRM

EXAMPLES:
```
POST INVENTRY TRANSACT MATCHING PARTNO AND ADD
 QUANTITY AND REPLACE DATE
POST ACCOUNTS RECEIPTS USING ACCTNO,ADD AMOUNT,QUAN
POST ACCOUNTS RECEIPTS ACCTNO ADD ORDERS SUB STOCK
 REP DATE
```

USAGE: THE USER CAN ADDRESS UP TO 32 FIELDS OR KEYS (IN COMBINATION) PROVIDING ALL THE FIELDS NAMED IN THE COMMAND LINE EXIST IN BOTH DATA BASES. ALSO REQUIRED, IS THAT BOTH DATA BASES HAVE IDENTICAL FIELD DEFINITIONS FOR THOSE FIELDS ADDRESSED. EACH OPERATOR ENCOUNTERED IN THE COMMAND LINE PERFORMS THE SAME UPDATE OPERATION ON EVERY FIELD IN THE LIST UNTIL A NEW OPERATOR IS ENCOUNTERED. FOR EXAMPLE,

```
POST ACCOUNTS RECEIPTS BY PARTNO AND REPLACE DATE,PAYMENT
```

IS THE SAME AS,

```
POST ACCOUNTS RECEIPTS BY PARTNO AND REPLACE DATE, REPLACE
PAYMENT
```

THE USE OF THE WORD "AND" AND USE OF COMMAS ARE OPTIONAL. COMMAS AND SPACES ARE INTERCHANGABLE.

ANY FIELD BELONGING TO BOTH DATA BASES MAY BE SPECIFIED IN THE COMMAND LINE. HOWEVER, ONLY DATA ITEMS DECLARED AS NUMERIC (N) OR DOLLAR($) MAY BE PRECEDED BY AN ARITHMETIC OPERATOR (E.G. ADD). A REPLACE MAY BE DONE ON ANY FIELD TYPE. IF AN ERROR CONDITION EXISTS BECAUSE OF INVALID OPERATION ON A NON-NUMERIC FIELD, THE PROGRAM WILL ABORT AND NO UPDATES WILL BE MADE.

THE UPDATING SEQUENCE OCCURS BY MATCHING KEYS OF DATABASE1 AND DATABASE2 RECORDS. WHEN A MATCH OCCURS, A COPY OF THE EXISTING DATABASE1 RECORD IS WRITTEN OUT TO $$.DAT AND TO A BUFFER. THE DATABASE1 RECORD IS THEN UPDATED IN MEMORY AS INDICATED BY THE OPERATORS IN THE COMMAND. IF AN OVERFLOW ERROR OCCURS IN ANY FIELD OF THE RECORD, THE DATABASE1 RECORD IS NOT UPDATED ON THE DISK, AND NOT WRITTEN TO $$.DAT. AN OVERFLOW ERROR WILL PRODUCE A MESSAGE ON THE CONSOLE SCREEN. THE OPERATOR WILL BE REQUIRED TO

INTERVENE AND EITHER CHOOSES TO ABORT THE PROGRAM OR CONTINUE ON, THEREBY SKIPPING THE UPDATE. IF THE ABORT OPTION IS CHOSEN, NO RESULT SET IS WRITTEN. NOTE: A PORTION OF THE MASTER MAY GET UPDATED IF PREVIOUS UPDATES HAD BEEN DONE WITHOUT ERROR.

IF NO FIELDS OR OPERATORS ARE SPECIFIED IN THE COMMAND LINE, (E.G., "POST ACCOUNTS RECEIPTS USING ACCTNO") THE COMMAND WILL DEFAULT TO A COMPLETE REPLACEMENT OPERATION. FOR EXAMPLE, "POST ACCOUNTS RECEIPTS USING ACCTNO" WILL CAUSE THE DEFAULT "AND REPLACE F1,F2,F3,.." WHERE F1,F2,F3.. ARE ALL THE FIELDS IN DATABASE2. NOTE: WHEN USING THE DEFAULT, ALL FIELDS IN DATABASE2 MUST BE A SUBSET OF DATABASE1. THE PROGRAM WILL ABORT IF THIS CONDITION IS NOT TRUE.

THE RESULT DATA BASE CREATED PROVIDES A CONVENIENT WAY TO MAINTAIN AN AUDIT TRAIL AND TO ALSO VERIFY THAT ALL DATABASE2 RECORDS HAVE BEEN POSTED. (SEE COMPARE COMMAND). $$.DEF AND $$.FRM ARE WRITTEN FOR CONVENIENCE WHICH ARE EXACT COPIES OF THE DATABASE1 DEFINITION AND FORM FILES, RESPECTIVELY.

THE POST COMMAND WILL PERFORM IN THE SHORTEST TIME WHENEVER THE TWO DATA BASES ARE IN SORT BY THE KEYS SPECIFIED IN THE COMMAND LINE. A SIGNIFICANT INCREASE IN PERFORMANCE IS ATTAINED WITH SETS IN APPROXIMATE SORT. THAT IS, IF ONLY A FEW RECORDS HAVE BEEN ADDED TO A MASTER FILE THAT WAS PREVIOUSLY IN SORT, SORTING AGAIN AND RUNNING THE POST PROGRAM WILL NOT SAVE MUCH TIME.

PRINT

SEE "LIST/PRINT" COMMAND DESCRIPTION.

NC

SELECT

PURPOSE: THIS COMMAND WILL WRITE A SUBSET OF A DATA BASE.

SYNTAX: SELECT DATABASE [WHERE CONDITION]

WHERE CONDITION IS DEFINED IN THE UPDATE COMMAND

RESULTANT SET: $$.DAT, $$.DEF AND $$.FRM

EXAMPLES:

```
SELECT CLIENTS WHERE SEX IS MALE
SELECT CLIENTS WHITE WHERE AGE IS GT 30
SELECT CLIENTS.DEF WHERE ITEM IS LT 10
```

USAGE: WHEN USING THIS COMMAND, RECORDS MEETING THE SEARCH CONDITION ARE SELECTED FROM THE PARENT DATA BASE AND WRITTEN TO A NEW DATA FILE CALLED $$.DAT. DUPLICATE COPIES OF THE PARENT DEFINITION AND FORM FILES WILL BE CREATED UNDER THE NAMES $$.DEF, $$.FRM. THE NEW DATA FILE, $$.DAT, WILL CONTAIN ALL THE ITEMS OF THE PARENT SET IN THIS CASE. NOTE: SELECT WILL ONLY OUTPUT ACTIVE RECORDS, THUS CAN BE USED TO "CLEAN" A FILE WITHOUT SORTING. IN THIS CASE, IT IS UP TO THE USER TO ERASE THE ORIGINAL SET FOLLOWING THE SELECT AND RENAME THE RESULT SET TO THAT OF THE ORIGINAL.

IN ORDER TO USE THE SELECTED DATA BASE THE USER MUST ENTER A RECORD INTO THE DICTIONARY CALLING THE NAMES OF THE OUTPUT FILES OR BY RENAMING OR COPYING THEM ONTO A NEW DISKETTE, WHERE A DICTIONARY ENTRY MAY ALREADY EXIST. IN GENERAL, ONE CAN RESERVE A DICTIONARY ENTRY UNDER THE TITLE "RESULT" AND SPECIFY THE DATA, FORM, DEFINITION AND CODED VALUE FILES AS $$. THESE FILES ARE CREATED ON THE CURRENT DISK DRIVE.

SORT

PURPOSE: THE COMMAND SORT ALLOWS THE USER TO SORT THE RECORDS OF A FILE IN ASCENDING OR DESCENDING ORDER BY AN ITEM VALUE(S).

SYNTAX: SORT DATABASE BY F1,F2,F3,...,F32{[D]}

RESULTANT SET: FILE IS RE-WRITTEN

EXAMPLES:

```
SORT CLIENTS BY NAME
SORT CLIENTS BY CITY STATE AND ZIP-CODE
```

USAGE: THE SORT WILL ORDER F1, AS THE MAJOR SORT ELEMENT THEN F2, THEN F3, AND SO ON. UP TO 32 ITEMS MAY BE SPECIFIED, DELIMITERS MAY BE SPACES, COMMAS OR "AND". IF "[D]" FOLLOWS THE COMMAND LINE, THE FILE WILL BE WRITTEN IN DESCENDING ORDER.

A TEMPORARY WORK FILE ($$.SRT) IS CREATED ON THE CURRENT DISK DRIVE FOR ALL SORTS AND THE DATA FILE IS REWRITTEN IN THE ORDER SORTED. THE WORK FILE IS ERASED AT THE END OF THE SORT. IF WORK SPACE IS IN QUESTION, IT IS A GOOD IDEA TO PUT A BLANK DISK INTO AN UNUSED DRIVE, LOG ONTO THE DRIVE AND START THE SORT TO ALLOW THE MAXIMUM AMOUNT OF ROOM TO HOLD THE WORK FILE. THERE MUST BE 1.0 TIMES THE SPACE OF THE ORIGINAL FILE AVAILABLE ON THE SORT DISK OTHERWISE AN OUT OF SPACE ERROR WILL OCCUR AND THE SORT ABORTED.

THE SORT PROGRAM WILL ALSO CLEAN UP THE FILE BY SIFTING OUT PREVIOUSLY DELETED (INACTIVE) RECORDS.

THE MAXIMUM FILE SIZE IS 128K BYTES FOR THE SORT PROGRAM. LARGER FILES MAY BE SORTED BY USING THE SELECT COMMAND TO BREAK THE FILE INTO SMALLER SECTIONS AND THEN SORT THE SECTIONS.

STAX(tistics)

PURPOSE: THIS WILL COMPUTE FILE STATISTICS ON SELECTED NUMERIC ITEMS. THESE STATISTICS INCLUDE MINIMUM, MAXIMUM, TOTAL, AND AVERAGE VALUES.

SYNTAX: STAX DATABASE BY ITEM1 ITEM2..ITEM 32{[P]}

EXAMPLES:

```
STAX CLIENTS BY AGE
STAX CLIENTS BY WAGE AND AGE[P]
```

USAGE: AS AN EXAMPLE, ONE CAN DETERMINE THE SALES EXPENSES AND COSTS FOR THE A DATA BASE CALLED "ACCOUNTS". THE USER WOULD TYPE THE COMMAND:

```
STAX ACCOUNTS BY SALES AND COSTS
```

THIS WOULD GIVE THE FOLLOWING OUTPUT:

```
SALES
  Minimum = 3.00
  Maximum = 55.00
  Total   = 10345.00
  Average = 41.38

COSTS
  Minimum = 4.00
  Maximum = 220.00
  Total   = 5435.00
  Average = 21.74
Total Records = 250
```

A MAXIMUM OF 32 ITEMS CAN BE INDICATED FOR THE STAX COMMAND. ALPHA OR ALPHANUMERIC ITEMS CANNOT BE REFERENCED FOR THE STAX COMMAND. OVERFLOW CAN OCCUR IF THE TOTAL OR CALCULATION OF THE AVERAGE PRODUCES A NUMBER GREATER THAN 4 BYTES. AN ERROR MESSAGE WILL BE DISPLAYED IN THIS CASE. IF [P] FOLLOWS THE COMMAND, THE OUTPUT WILL BE SENT TO THE PRINTER.

TERM

PURPOSE: SELECTS CRT TERMINAL AND SETS SCREEN CODES

SYNTAX: TERM [Type]

EXAMPLES:
```
TERM
TERM LEAR
TERM ADDS
TERM MISC
```

USAGE: THE USER CAN SELECT 8 DIFFERENT TERMINAL TYPES BY SPECIFYING A FOUR LETTER CODE. THE TERMINALS COMPATIBLE WITH CONDOR SERIES 20/DBMS ARE THE FOLLOWING TYPES:

ADDS	-	ADDS REGENT 20 25 40 60 100 200
BEEH	-	BEEHIVE MICRO-B1 MICRO-B2 B100 B150 B550
HAZL	-	HAZELTINE 1500 1510 1520
INFO	-	INFOTON (GTC) 100 101 200 400
LEAR	-	LEAR SEIGLER ADM-1A ADM-2 ADM-3A ADM-31 ADM-42
PERK	-	PERKIN ELMER FOX 1100 OWL 1200
SORC	-	SOROC IQ-120 IQ-140
TRS2	-	PICKLES AND TROUT CP/M FOR TRS80-II

ANY TERMINAL THAT EMULATES THE ABOVE TYPES WITH RESPECT TO THE ERASE SCREEN, CURSOR CONTROL AND HOME FUNCTIONS MAY BE USED. IF THE CORRECT TERMINAL TYPE IS CHOSEN, THE "||READY||" MESSAGE WILL BE DISPLAYED IN THE MIDDLE OF A NEWLY ERASED SCREEN.

IF THE "TERM MISC" IS CHOSEN, THE USER CAN PROVIDE THE SCREEN FUNCTION CODES TO THE PROGRAM AND DEFINE A FOUR LETTER TERMINAL TYPE THAT CAN BE SAVED. NOTE: IT MAY NOT BE NAMED ANY OF THE ABOVE TERMINAL CODES.

TITLE

PURPOSE: PRINTS TITLE, DATE ON EVERY PAGE OF A REPORT

SYNTAX: TITLE CLAUSE

WHERE CLAUSE CONSISTS OF ONE OR MORE OF THE CHARACTERS OR WORDS DEFINED BELOW:

```
DATE = PRINTS CURRENT SIGN ON DATE
L    = LINE FEED
R    = CARRIAGE RETURN
T    = TOP OF FORM
X    = SPACE
' '  = ENCLOSED TITLE
```

EXAMPLES:

```
TITLE 'FINANCIAL REPORT FOR ABC COMPANY AS OF ',DATE
TITLE L,L,'TWO LINE FEEDS',L,'AND A TWO LINE TITLE'
TITLE
TITLE 'THE ABOVE CLEARS TITLE PRINTING',X,X,X,DATE
```

USAGE: THE TITLE COMMAND WILL PRINT THE CLAUSE ON THE TOP OF EVERY PAGE OF A REPORT. THIS OCCURS AFTER 64 LINES HAVE BEEN SENT TO THE PRINTER THEN A FORM FEED IS ISSUED, FOLLOWED BY THE CLAUSE AND TWO MORE LINE FEEDS. WHEN THERE HAS BEEN NO TITLE GIVEN ONLY A FORM FEED WILL BE ISSUED AT THE END OF 64 LINES. THE TITLE IS CLEARED BY GIVING THE COMMAND "TITLE". THE PAGE NUMBER IS ALWAYS PRINTED AFTER ALL THE TEXT AND CONTROL CHARACTERS HAVE BEEN PRINTED. THE DATE WILL ALWAYS APPEAR IN THE FORMAT mm/dd/yy USING THE CURRENT SIGN ON DATE. IF BLANK IT WILL PRINT NOTHING. THE DATE APPEARS WHEREVER IT HAS BEEN INSERTED IN THE TITLE LINE.

UPDATE

PURPOSE: TO UPDATE ANY DATA VALUE IN ANY RECORD OF THE DATA BASE.

SYNTAX: UPDATE DATABASE [WHERE CONDITION]

DELIMITER SYNONYMS: "ST" OR "WHOSE" MAY SUBSTITUTE FOR "WHERE"

USAGE: THE TYPICAL USE FOR THIS COMMAND IS TO MAKE AN INQUIRY OR UPDATE A RECORD OR A SET OF RECORDS OF A DATA BASE. THE DATA BASE CAN ALSO BE A DEFINITION OR DICTIONARY FILE. IT MAY ALSO BE THE DIRECTORY OF THE DATA FILE. IN THESE ONES, TYPE THE APPROPRIATE EXTENSION .DEF, .DIC, OR .DIR.

DEFINITION OF TERMS: CONDITION IS DEFINED AS CLAUSE1 CONJ CLAUSE2...

CONJ IS THE LOGICAL OPERATOR "AND" OR THE NON-EXCLUSIVE "OR". BOTH CANNOT APPEAR IN THE SAME COMMAND LINE.

CLAUSE1, 2,...ETC ARE DEFINED AS ONE OR MORE OF THE RELATIONAL CLAUSES:

```
DATA-ITEM VERB VALUE1[,VALUE2,...]
```

VERB IS ONE OF THE RELATIONAL VERBS (SYNONYMS SHOWN):

=	IS, ARE, EQUALS, EQ
<>	NE, IS NOT, ARE NOT
<	LT, IS LT, IS NOT GE
>	GT, IS GT, IS NOT LE
<=	LE, IS LE, IS NOT GT
>=	GE, IS GE, IS NOT LT

VALUES CAN BE EITHER CONSTANTS OR VARIABLES (DATA-ITEM NAMES). A VARIABLE MUST BE PRECEEDED BY THE COMMERCIAL AT SIGN (@), OTHERWISE IT IS CONSIDERED A CONSTANT. ONE OR MORE SPACES ARE REQUIRED BETWEEN VARIABLES, VERBS, AND VALUES, AS WELL AS BETWEEN CONJ AND THE RELATIONAL CLAUSES.

EXAMPLES:

```
UPDATE CLIENTS WHERE NAME IS SMI*
UPDATE CLIENTS WHERE AGE GT 34
UPDATE CLIENTS FEMALE WHOSE NAME IS "DOE JANE"
UPDATE DEFSET.DEF ST FIELD IS 1 2 4
UPDATE DATA.DIC WHERE TITLE IS CLIENTS
UPDATE CLIENTS ST ENROLL-DATE GT @TERM-DATE
```

ENTERING DATA INTO EXISTING DATA BASES

USING THE U P D A T E COMMAND

The following data bases are ready for update:

MF-DEL-1: now contains the names of

cut tube lengths

MF-DEL-H: now contains information for

the two hoists 1030 and 2503.

With the disk containing these data bases in **drive B:** and with the **CP/M** disk in **drive A:** we begin data base operations by doing the following:

NOTE: user responses are underlined.

```
ASUBMIT START
 ------------
```

The machine goes through its paces and comes to a stop with

```
TERM
TERMINAL = DEL1
           ----
```

After your DEL1 response user will be requested to insert today's date.

This must be entered in the form **mm/dd/yy** for month/day/year, **note two letters for each.**

The system displays a prompt:

If after testing your terminal codes, it is necessary to change codes, type <u>TERM<C/R></u> <u>MISC</u> <u><C/R></u>. The default values will now be the codes entered previously. Respond with <u><C/R></u> to retain correct codes. Type new codes for codes to be changed. For example, if we wish to change the number of nulls we might type the following commands:

```
A>TERM<C/R>

Terminal = MISC<C/R>
Specify CRT screen control codes by typing the DECIMAL value of the code(s)
 Parentheses () indicate the number of codes or range expected
 Brackets [] indicate the current default values
 Entering only a <C/R> causes the default values to be used

Enter screen clear sequence (1-2) [ 27 42 ]:<C/R>
      Screen clear delay in nulls (0-99) [ 6 ]:8<C/R>
Enter home cursor seqrence (1-2) [ 30 ]:<C/R>
Enter cursor addressing sequence (2) [ 27 61 ]:<C/R>
      Line or Column sent first (1) [L]?:<C/R>
      Line and Column offsets (2) [ 32 32 ]:<C/R>
      Cursor addressing delay in nulls (0-99) [ 6 ]:8<C/R>
Enter back cursor sequence (1-2) [ 8 ]:<C/R>
      forward cursor sequence (1-2) [ 12 ]:<C/R>
      up cursor sequence (1-2) [ 11 ]:<C/R>
      down cursor sequence (1-2) [ 10 ]:<C/R>

Parameters OK (Y/N)? Y
Enter Terminal ID (4) [IQ20]:<C/R>

|| READY ||

A>
```

Note that the default values this time were values previously defined.

APPENDIX B

TERM COMMAND - MISC(ELLANEOUS) OPTION

THE "MISC" OPTION OF THE CONDOR SERIES 20/DBMS "TERM" COMMAND ALLOWS FOR DEFINING SCREEN FUNCTION CODES TO THE COMPUTER PROGRAM AND A FOUR LETTER TERMINAL TYPE CODE THAT CAN BE SAVED. ONCE THE FUNCTION CODES AND TYPE CODE ARE PROVIDED, ONLY THE TYPE CODE NEED BE ENTERED TO SELECT THE CRT TERMINAL. THE OPTION IS USED TO PROVIDE FOR TERMINALS, WHICH MEET CONDOR SERIES 20/DBMS REQUIREMENTS, BUT ARE NOT DESCRIBED IN THE TERM COMMAND OF APPENDIX A.

PERFORM THE FOLLOWING STEPS TO SET FUNCTION AND TYPE CODES:

Type: A:TERM <C/R>

Computer will respond "Terminal =".

Type: MISC <C/R>

Computer responds with instructions, syntax description, and prompting messages, as illustrated below. The responses to computer prompts are underlined.

```
**********************************************************

A>TERM<C/R>

Terminal = MISC<C/R>
Specify CRT screen control codes bŷ typing the DECIMAL value of the code(s)
 Parentheses () indicate the number of codes or range expected
 Brackets [] indicate the current default values
 Entering only a <C/R> causes the default values to be used

Enter screen clear sequence (1-2) [ 26 ]:27 42<C/R>
      Screen clear delay in nulls (0-99) [ 0 ]:6<C/R>
Enter home cursor seqrence (1-2) [ 30 ]:<C/R>
Enter cursor addressing sequence (2) [ 27 61 ]:<C/R>
      Line or Column sent first (1) [L]?:<C/R>
      Line and Column offsets (2) [ 32 32 ]:<C/R>
      Cursor addressing delay in nulls (0-99) [ 0 ]:6<C/R>
Enter back cursor sequence (1-2) [ 8 ]:<C/R>
      forward cursor sequence (1-2) [ 12 ]:<C/R>
      up cursor sequence (1-2) [ 11 ]:<C/R>
      down cursor sequence (1-2) [ 10 ]:<C/R>

Parameters OK (Y/N)? Y
Enter Terminal ID (4) [MINE]:IO20<C/R>

|| READY ||

A>

**********************************************************
```

APPENDIX C

DEFINE COMMAND, DESCRIPTION OF OPTIONS

The CONDOR SERIES 20/DBMS "DEFINE" command provides various options for creating alternate data bases or differant "views" of a data base. If a "normal" data base could be described, the data base files (.FRM, .DEF, and .DAT) would each have a common data base name. This name would also be the TITLE in the data dictionary. For example, an EMPLOYEE data base may be titled EMPLOYEE and be composed of the files EMPLOYEE.FRM, EMPLOYEE.DEF, and EMPLOYEE.DAT. The greater percentage of your data bases will be of this type but there will be requi[illegible] where alternates are desirable.

This appendix describes how alternate files are defined, how to erase a data base, and how to change a data item definition in a data base, all with the DEFINE command. Actual computer illustrations are used and entries from the video terminal are underlined.

ERASING A DATA BASE

The session below illustrates DEFINE command responses during an erase operation. To begin type "DEFINE database" where database is the name of the data base to be erased. The responses to computer prompts are underlined.

```
***************************************************************
*                                                             *
*        A>DEFINE TESTX<C/R>                                  *
*        Condor DBMS Version 1.05                             *
*                                                             *
*        This Data Base exists already                        *
*        Choose Option:Re-define(R), Erase(E), or End Program <C/R>:E
*        Confirmation required to erase this Data Base (Y/N)?Y *
*        PARTS   FRM also be erased (Y/N)?N                   *
*        A>                                                   *
*                                                             *
*                                                             *
***************************************************************
```

Note that it only prompted for files in the data base that do not have the same name as the data base.

```
***************************************************************
*                                                             *
*       A>DEFINE TESTX<C/R>                                   *
*       Condor DBMS Version 1.05                              *
*                                                             *
*       This Data Base exists already                         *
*       Choose Option: Re-define(R), Erase(E), or End Program <C/R>:E
*       Confirmation required to erase this Data Base (Y/N)?N *
*                                                             *
*       ** DEFINE Aborted **                                  *
*       A>                                                    *
*                                                             *
*                                                             *
***************************************************************
```

DEFINING AN ALTERNATE DATA BASE

The second session illustrates defining a data base when a .FRM file of the same name does not exist. For example, the command DEFINE TESTX is entered and the file TESTX.FRM does not exist though a form file does exist within another data base.

```
A>DEFINE TESTX<C/R>
Condor DBMS Version 1.05

TESTX   FRM does not exist - Enter different name or End Program <C/R>:PARTS<C/R>

Do you wish to use an existing Definition file (Y/N)?N

Enter data definitions in the following format:

>FIELD-NAME, FIELD-TYPE, FIELD-SIZE, MIN-VALUE, MAX-VALUE, "DEFAULT-VALUE"
   Choices :  (ANJ$R)   (1-127 bytes)  (1 - 10 digits)    (0-15 Characters)

>1:PART,A<C/R>,9,0,9
>2:COLOR,A<C/R>,17,0,17
>3:QUANTITY,N<C/R>,2,0,32767
>4:TYPE,*<C/R>,9,0,9                (* and AN are synonyms)
>5:MFR,*<C/R>,30,0,30
>6:[[ END ]]

Definitions OK (Y/N)?Y

Busy

Attributes for Data Base: TESTX

Number of Fields Defined = 5
Record Size (Bytes) = 68

* Definition Completed *

A>
```

REDEFINING WITH DATA IN DATA FILE

This session describes "redefining" mode. In the illustration below, the data base TESTX already exists and the data file is not empty.

```
A>DEFINE TESTX<C/R>
Condor DBMS Version 1.05

This data Base exists already
Choose Option: Re-define(R), Erase(E), or End Program <C/R>:R
Data file is not empty: OK to delete all records (Y/N)?Y

Enter data definitions in the following format:

>FIELD-NAME, FIELD-TYPE, FIELD-SIZE, MIN-VALUE, MAX-VALUE, "DEFAULT-VALUE"
   Choices :  (ANJ$R)   (1-127 bytes)  (1 - 10 digits)    (0-15 Characters)

>1:PART,AN<C/R>,9,0,9
>2:COLOR,<C/R>
>#<C/R>{( END )}

Definitions OK (Y/N)?Y

Busy

Attributes for Data Base: TESTX

Number of Fields Defined = 5
Record Size (Bytes) = 68

* Definition Completed *

A>
```

Note that entering #<C/R> ends the data definition sequence.

REDEFINING WITH EMPTY DATA FILE

Redefining to change one or more data item definitions. Data file is empty.

In the illustration, we need to change the data definition of data item QUANTITY.

```
A>DEFINE TESTX<C/R>
Condor DBMS Version 1.05

This Data Base exists already
Choose Option: Re-define(R), Erase(E), or End Program <C/R>:R

Enter data definitions in the following format:

>FIELD-NAME, FIELD-TYPE,_FIELD-SIZE, MIN-VALUE, MAX-VALUE, "DEFAULT-VALUE"
   Choices :  (ANJ$R)   (1-127 bytes)  (1 - 10 digits)    (0-15 Characters)

>1:PART,<C/R>
>QU*<C/R>
>3:QUANTITY,NR<C/R>,2,0,32767
>4:TYPE,<C/R>
>*<C/R>(( END ))

Definitions OK (Y/N)?Y

Busy

Attributes for Data Base: TESTX

Number of Fields Defined = 5
Record Size (Bytes) = 68

* Definition Completed *

A>
```

****************** CONDOR SOFTWARE UPDATE NOTICE - 00013 ******************

: 11/7/80
JCT: SERIES 20 DBMS (Level I)
CT: DEFINE, DESTROY, HELP, READ, REORG, TERM, WRITE COMMANDS
ION: 1.10

****

1. Four New commands have been added to release 1.10.
   These are DESTROY, READ, REORG, WRITE commands.
   They are described in the addendum dated November 1980.

2. DEFINE.COM: Checks now if form has more than 127 labels, aborts with error message.

3. HELP.COM: Will not prompt for number if no menu items are bracketed (viz., system.hlp). In this case, any keyboard character will erase display when struck and return user to command level.

4. TERM.COM: Permits 1 or 2 character sequence for cursor address load-in sequence. Also permits cursor address offset to be 128. This fix allows the use of Vector Graphic microcomputers with its memory mapped CRT.

5. MISC: Enable interupt (EI) instruction removed, which affected only the Vector Graphic microcomputers when DBMS system loaded.

```
------------------ CONDOR SERIES 20 DBMS COMMANDS ----------
1.09   APPEND   :APPEND database1 database2
.03    COMPARE  :COMPARE database1 database2 (not)(matching) f1 f2 ... fie
1.06   COMPUTE  :COMPUTE database st fr = f1 opr f2 opr ...f31
1.2    DATE     :DATE (mm/dd/yy)
1.0    DEFINE   :DEFINE database (#)
1.0    DESTROY  :DESTROY database (OK)
1.03   EMPTY    :EMPTY database
1.07   ENTER    :ENTER database
1.04   FORMAT   :FORMAT formfile(.frm)
1.00   HELP     :HELP file(.hlp)
1.06   LIST     :LIST database (by field1 field2  ...field32)
1.04   POST     :POST database1 database2 (matching) k1,k2..(OPR f3,f4,..f
1.06   PRINT    :PRINT database (by field1 field2 ... field32)
* 1.10 READ     :READ database (input-file)
1.10   REORG    :REORG database (formfile.frm)
1.08   SELECT   :SELECT database (where condition)
1.08   SORT     :SORT database by field1 field2 ....field32
1.02   STAX     :STAX database by field1 field2 ....field32
1.10   TERM     :TERM ADDS | BEEH | HAZL | INFO | LEAR | PERK | SORC | TRS
1.08   TITLE    :TITLE 'text enclosed in literals or quotes',T,L,R,DATE
1.08   UPDATE   :UPDATE database (where condition)
* 1.10 WRITE    :WRITE database (outfile) (switch option)
```

* updated

CONDOR SERIES 20 DBMS CDOS OPERATION
(LEVEL I)

****

WHEN USING THE CONDOR SERIES 20 DBMS CDOS VERSION, THERE ARE A FEW SYNTACTICAL DEPARTURES FROM THE MANUAL AS DESCRIBED BELOW:

1. CONDOR SERIES 20 DBMS IS COMPATIBLE WITH ONLY CDOS SERIES 2.XX THERE IS A SPECIAL VERSION OF SERIES 20 DBMS WRITTEN FOR CDOS OPERATION. A SIGN ON MESSAGE WILL INDICATE THE VERSION IS CDOS COMPATIBLE. IF THIS MESSAGE IS NOT DISPLAYED YOU HAVE A CP/M VERSION. PLEASE RETURN YOUR MASTER COPY TO CONDOR REQUESTING A SERIES 20 DBMS CDOS VERSION.

2. THE COMMAND LINE PROMPT IS "x.", (e.g., A.)

3. USE XFER.COM ON CDOS TO MAKE COPY (I.E., XFER A:=B:*.*)

4. SUBSTITUTE ALL OCCURRENCES OF "SUBMIT" WITH "@" (BATCH). NOTE THAT THE DEMO DISKETTE HELP FILE "RESTART.HLP" MUST BE EDITED WITH THIS CHANGE.

5. SUBSTITUTE ALL OCCURRENCES OF FILENAMES WITH A ".SUB" EXTENSION WITH A ".CMD" EXTENSION. NOTE ON THE DEMO DISKETTE ALL BATCH FILES WITH THE .SUB EXTENSION MUST BE RENAMED TO THE .CMD EXTENSION.

6. FILENAMES OF THE TYPE $$.EXT APPEAR AS THE SAME NAMES IN BATCH FILES (NOTE: CP/M REQUIRES IT TO BE $$$$.EXT) NOTE: ON THE DEMO DISKETTE, "ACCNTS.SUB" MUST BE EDITED SO THAT $$.DAT REPLACES $$$$.DAT

7. BATCH PROCEDURES CAN BE INITITATED AND EXECUTED ON ANY DRIVE.

8. CDOS EDITOR IS CALLED "EDIT". REPLACE ALL OCCURENCES OF "ED", WITH "EDIT". THE ESCAPE KEY ENDS EDIT MODE.

ADDENDUM

SERIES 20 DBMS

ADDITIONAL COMMANDS

(LEVEL I)

CONDOR COMPUTER CORPORATION
3989 RESEARCH PARK DRIVE
ANN ARBOR, MICHIGAN 48104

NOVEMBER 1980

DESTROY

PURPOSE: ERASES ALL FILES OF A DATABASE

SYNTAX: DESTROY DATABASE [OK]

EXAMPLES:

```
DESTROY CLIENTS
DESTROY PARTS OK
```

USAGE: THE DESTROY COMMAND ERASES ALL FILES OF A DATABASE PERMANENTLY AND DELETES THE TITLE FROM THE DATA DICTIONARY. A CONFIRMATION IS ALWAYS REQUIRED TO PROCEED. IF THE DATABASE TITLE IS DIFFERENT FROM ANY OF THE FILES MAKING UP THE DATABASE (I.E., THE DATA, DEFINITION OR FORM FILENAMES), THE USER WILL BE PROMPTED TO ERASE THESE AS WELL. IF THE CONFIRMATION "OK" IS GIVEN IN THE COMMAND LINE, NO PROMPTS WILL BE REQUESTED AND ALL FILES ASSOCIATED WITH THE DATABASE WILL BE ERASE, REGARDLESS OF THE NAMES. CAUTION: ONCE A DATABASE HAS BEEN ERASED, THERE IS NO WAY TO RECOVER THE DATA.

PURPOSE: CONVERTS AN ASCII DATA FILE INTO A SERIES 20 DATABASE

SYNTAX: READ DATABASE [INPUT-FILENAME]

WHERE IF NO INPUT FILE SPECIFIED, $$.ASC IS READ

EXAMPLES:

```
READ CLIENTS
READ CLIENTS APPLCNT.ASC
```

USAGE: THE READ COMMAND IS USEFUL FOR CONVERTING EXISTING DATA FILES THAT ARE IN ASCII STRING FORMAT TO A SERIES 20 DATABASE. THE DATABASE MUST BE DEFINED PREVIOUSLY (SEE DEFINE COMMAND). THE ASCII DATA FILE CAN CONTAIN EITHER FIXED OR VARIABLE LENGTH RECORDS ACCORDING TO ANY OF THE FORMATS DESCRIBED BELOW:

FORMAT	RECORD TYPE
1.FIXED LENGTH	CONTINUOUS STREAM OF STRING CHARACTERS,NO BINARY CHARACTERS DEMARCATING FIELDS OR RECORDS.
2.FIXED LENGTH	STRING CHARACTERS WITH <C/R> OR <C/R>-<LF> DEMARCATING RECORDS, NO FIELD DELIMITERS.
3.FIXED OR VARIABLE LENGTH	STRINGS ENCLOSED IN QUOTES ("), COMMAS DELIMITING FIELDS, <C/R> OR <C/R>-<LF> DEMARCATING RECORDS.
4.VARIABLE LENGTH	STRINGS SEPERATED BY TABS (ASCII 09), RECORDS DEMARCATED BY <C/R>-<LF>.

NOTE: ALL ASCII DATA FILES MUST BE TERMINATED WITH A CONTROL-Z (ASCII 01AH), INDICATING END OF FILE.

ALL INPUT RECORDS ARE FORMATTED AND ENCODED BY THE DATABASE FORM AND DEFINITION FILES. THE INPUT RECORD IS FORMATTED ACCORDING TO THE NUMBER OF UNDERSCORES GIVEN TO EACH DATA ITEM IN THE FORM. FOR EXAMPLE, IF A DATA ITEM CALLED [NAME] HAS 12 UNDERSCORES IN THE FORM, THE INPUT STRING WILL BE READ AS IF IT IS 12 CHARACTERS IN LENGTH. IF THE STRING IS DELIMITED AS DESCRIBED ABOVE AND IS LESS THAN 12 CHARACTERS, THE REMAINDER OF THE FIELD WILL BE BLANK FILLED. IF THE STRING IS GREATER THAN 12 BYTES AND DELIMITED, AN ERROR MESSAGE WILL BE DISPLAYED ON THE CONSOLE INDICATING WHICH

ASCII RECORD NUMBER WAS IN ERROR. THIS RECORD WILL BE SKIPPED AND NOT READ INTO THE DATABASE. IF THE INPUT RECORD CONTAINS BINARY DATA OTHER THAN FIELD OR RECORD DELIMITERS, IT TOO WILL CAUSE AN ERROR AND TREATED AS DESCRIBED ABOVE. EACH FIELD WILL BE ENCODED AS DESCRIBED IN THE DEFINITION FILE AND CHECKED FOR CONSISTENCY WITH THE DECLARED FIELD TYPE. IF THEY DO NOT AGREE, AN ERROR MESSAGE IS DISPLAYED AND THE RECORD SKIPPED.

PURPOSE: REORGANIZE DATA ITEMS OF A DATABASE

SYNTAX: REORG DATABASE [FORMNAME.FRM]

RESULT SET: $$.DAT, $$.DEF, $$.FRM OR DATABASE REPLACED

EXAMPLES:

```
REORG CLIENTS
REORG CLIENTS NEWFORM.FRM
```

USAGE: THE REORG COMMAND PERMITS THE USER TO CHANGE THE ORDER OF, AND/OR ADD/DELETE DATA ITEMS IN A DATABASE. EACH RECORD OF THE DATABASE IS REARRANGED INTO A NEW FORMAT AS DEFINED BY A SECOND FORM (THE NEW FORMAT). THE USER MUST CREATE THE NEW FORM USING THE FORMAT COMMAND. IF THE NEW FORMAT HAS NEW DATA ITEM NAMES, THE USER WILL BE PROMPTED TO SUPPLY APPROPRIATE DEFINITIONS (SEE DEFINE COMMAND). NEW DATA VALUES ARE INITIALIZED WITH NULLS IN THE RECORDS. IF ANY DATA ITEMS ARE DELETED, THE USER WILL BE PROMPTED TO CONFIRM THESE AS LISTED ON THE CONSOLE.

THE DATABASE IS READ AND REORGANIZED ACCORDING TO THE NEW FORMAT AND A RESULT CREATED. THE RESULT SET MAY BE USED TO REPLACE THE ORIGINAL DATABASE OR USED AS AN INTERMEDIATE RESULT.

IF THE NEW FORM NAME IS SUPPLIED ON THE COMMAND LINE, THE USER WILL NOT BE PROMPTED UNLESS NEW DEFINITIONS ARE REQUIRED. IF NO NEW DEFINITIONS ARE REQUIRED, THE REORG COMMAND CAN BE USED IN A BATCH STREAM. NOTE: IN BATCH, THE RESULT SET WILL ALWAYS REPLACE THE ORIGINAL DATABASE.

WRITE

PURPOSE: CONVERTS A SERIES 20 DATABASE INTO AN ASCII DATA FILE

SYNTAX: WRITE DATABASE [OUTPUT-FILENAME] ([SW])

WHERE IF NO OUTPUT FILE SPECIFIED, $$.ASC WRITTEN

EXAMPLES:

```
WRITE CLIENTS
WRITE CLIENTS REPORT
WRITE CLIENTS REPORT1 [B]
```

USAGE: THE WRITE COMMAND IS USEFUL FOR CONVERTING A SERIES 20 DATABASE INTO AN ASCII DATA FILE THAT CAN BE READ BY OTHER APPLICATION PROGRAMS OR REPORT GENERATORS (E.G., APPLICATIONS WRITTEN IN A HOST LANGUAGE SUCH AS BASIC, FORTRAN, ETC).

FOUR DIFFERENT TYPES OF OUTPUT FORMATS MAY BE GENERATED BY CHOOSING THE APPROPRIATE SWITCH OPTION [SW] AS DESCRIBED IN THE FOLLOWING TABLE:

SWITCH OPTION	OUTPUT RECORD FORMAT
NONE	FIXED LENGTH, CONTINUOUS ASCII STREAM NO DELIMITERS BETWEEN DATA ITEMS. NUMERIC DATA ITEMS ARE RIGHT JUSTIFIED.
[A]	ABRIDGED - VARIABLE LENGTH - DATA ITEMS SEPERATED BY TAB (ASCII 09), RECORDS DEMARCATED BY <C/R>-<LF>. ALL TRAILING BLANKS ARE TRUNCATED.
[B]	BASIC - VARIABLE LENGTH - DATA ITEMS ENCLOSED IN QUOTES (") AND DELIMITED BY COMMAS. RECORDS DEMARCATED BY <C/R> <LF>.
[R]	RPG - FIXED LENGTH - NO DELIMITERS BETWEEN DATA ITEMS, RECORDS ARE DEMARCATED BY <C/R>-<LF>. NUMERIC FIELDS ARE RIGHT JUSTIFIED.

NOTE: ALL ASCII FILES WILL BE TERMINATED WITH A CONTROL-Z (ASCII 01AH), INDICATING THE END OF FILE.

THE "B" FORMAT IS COMPATIBLE WITH MICROSOFT'S BASIC AS WELL AS CBASIC. ALL DATA ITEMS SHOULD BE INPUT AS STRING VARIABLES AND NUMERIC ITEMS CONVERTED TO THE APPROPRIATE STORAGE TYPE.