July 1978

This manual describes the use of DATATRIEVE to extract and display data from RMS-11K files on TRAX operating systems.

TRAX
DATATRIEVE
User's Guide

Order No. AA-D347A-TC

OPERATING SYSTEMS AND VERSIONS: TRAX Version 1.0

SOFTWARE VERSION: 1.0

To order additional copies of this document, contact the Software Distribution Center, Digital Equipment Corporation, Maynard, Massachusetts 01754

digital equipment corporation · maynard, massachusetts

First Printing, July 1978

The information in this document is subject to change without notice and should not be construed as a commitment by Digital Equipment Corporation. Digital Equipment Corporation assumes no responsibility for any errors that may appear in this document.

The software described in this document is furnished under a license and may only be used or copied in accordance with the terms of such license.

No responsibility is assumed for the use or reliability of software on equipment that is not supplied by DIGITAL or its affiliated companies.

The postage-prepaid READER'S COMMENTS form on the last page of this document requests the user's critical evaluation to assist us in preparing future documentation.

The following are trademarks of Digital Equipment Corporation:

DIGITAL	DECsystem-10	MASSBUS
DEC	DECtape	OMNIBUS
PDP	DIBOL	OS/8
DECUS	EDUSYSTEM	PHA
UNIBUS	FLIP CHIP	RSTS
COMPUTER LABS	FOCAL	RSX
COMTEX	INDAC	TYPESET-8
DDT	LAB-8	TYPESET-11
DECCOMM	DECSYSTEM-20	TMS-11
ASSIST-11	RTS-8	ITPS-10

CONTENTS

CONTENTS (Cont.)

FIGURES

TABLES

PREFACE

This manual is directed toward the programmer who combines familiarity and access to RMS-11 files with a need to rapidly extract and display data from those files. This manual serves as both a user's guide and reference manual to DATATRIEVE-11. As a minimum, the reader should have a basic knowledge of TRAX, and should have read the **Introduction to RMS-11**.

Ideally, the reader is a programmer or Data Administrator experienced with at least one programming language such as COBOL or BASIC-PLUS-2. Regardless of the amount of actual programming experience, the user would have studied the language reference manual and user's guide for at least one of these languages. Thus, it is also assumed the reader has acquired basic input/output and file concepts from these manuals.

This user would be experienced with various keyboards found on terminal devices and would be able to adapt if a given keyboard used the DELETE or RUBOUT key to perform an erasing function.

Any reader lacking the above qualifications should not proceed further, but should consider whether it would be more appropriate to read instead the **Beginner's Guide to DATATRIEVE-11**. This latter manual describes a subset of the material included here, and in more elementary terms. Not every user requires -- or wants to learn -- all the capabilities of DATATRIEVE-11. This is something that each user's management must determine on an individual basis.

The organization of this manual leads you into the full language capabilities first. The topics are presented in a logical learning sequence in Chapters 1 through 4, rather than by any alphabetical or implied functional order. In Chapter 5, you will find all the information necessary to learn to use the Report Writer facility to customize and print a variety of types of reports using data from RMS-11 files. You will find that all the syntax rules you learned in Chapter 3 carry over directly into the Report Writer statements.

Chapter 6 introduces procedure definitions and presents the method used to invoke a procedure.

Chapter 7 presents topics of special interest to Data Administrators only. The general reader will probably want to skip Chapter 7.

Appendix A is worth your study once you have read the entire manual, since it provides a sample of an interactive DATATRIEVE-11 Session. It contains a number of additional examples of the commands, and should help reinforce what you have already learned.

Appendixes B, C, and D provide summary information for convenient reference. Here you will find a summary of all the command formats, a list of DATATRIEVE-11 keywords, and an alphabetized list of all the error messages.

Appendix E summarizes some COBOL rules that apply to data definitions used by DATATRIEVE-11.

SYMBOLOGY

Throughout this manual you will encounter certain consistent usages of symbols, colors, and underlining for clarifications. These conventions are summarized in Table P-1.

Table P-1
Symbols Used in This Manual

Symbol	Name	Meaning
RET	Return	Symbolizes the carriage return key or function on terminal keyboards.
[ ]	Syntax Brackets	A syntax symbol in command formats that indicates select one of the enclosed entries if you wish.
[ ]	Square Brackets	These smaller square brackets are keyboard symbols that you enter in the places shown.
{ }	Braces	A syntax symbol in command formats that indicates select one of the enclosed items as a required entry.
...	Ellipsis	A syntax symbol denoting possible repetition of the preceding elements.
red print		Anything you would type on your terminal. All system printout appears in black print.
^	Control Character	Represents the CTRL key on many terminals. Normally struck in unison with the alphabetic character that immediately follows.
XXXXX	Uppercase Letters	Used for command names or other elements in the statements that must be spelled as shown.
xxxxx	Lowercase letters	Used in the command syntax to represent items to be replaced by the appropriate names or values.
␣	Graphic blank or space	Symbolizes the blank or space character. Only used here to denote **required** spaces.
n	Lowercase n (an integer number)	Represents an integer decimal number.
___	Underlining	Used in the command syntax with uppercase words to represent a keyword that must be specified in every use of the command, unless the keyword is enclosed in outer syntax brackets.

CHAPTER 1

GENERAL CONCEPTS

This chapter presents the definitions and fundamentals necessary to learn DATATRIEVE-11 quickly. DATATRIEVE-11 accepts English-like commands from the user and reacts by extracting data from RMS-11 files. It can even be directed to modify or update the file. This entire process occurs in **interactive mode** -- which means that you issue a command, the software responds, and you may either correct an error, continue by issuing more commands, or stop. Most users become so comfortable with the language they are able to construct the commands while at the terminal and adapt to the ongoing changes; they do not find it necessary to preplan on paper every command that will be entered. In this way it works much like a conversation with a friend, and is sometimes called a dialogue.

Everything that can be done in DATATRIEVE-11 can be done in application programs. However, there are some distinct advantages to the interactive approach:

- DATATRIEVE-11 "programs" are specific to a given need.
- DATATRIEVE-11 sessions are shorter with less coding overhead.
- Time consuming compilations are not needed.
- Users can respond to errors or unexpected variations in results at execution time.
- Any particular session can be lengthened or shortened at will.

The Report Writer facility of DATATRIEVE-11 will take data operated on by DATATRIEVE-11 (or straight from an RMS-11 file) and print it to conform to user-specified formats.

1.1 INTRODUCTION TO DATATRIEVE-11 INQUIRY FACILITIES

You direct DATATRIEVE-11 with English-like **commands.** DATATRIEVE-11 is a command interpreter that both interprets and acts on each statement as received. Section 2.4 and Chapter 3 provide you with all the common rules that apply to using the commands. Your major task will be to formulate the commands that express your desires accurately to DATATRIEVE. A number of requirements can be fulfilled by alternate combinations of commands. This means that the "right" solution to a DATATRIEVE-11 problem is any solution that works.

When you make mistakes, you will be pleased to find that the consequences are seldom very severe. You can always go back to where you started and issue a corrected version of the command(s). This may be a bit of a nuisance, but at least you will not have lost or destroyed your data in the meantime.

There will be certain sequences of commands that must be issued on a recurring basis at every installation. As a result, DATATRIEVE-11 has a feature that permits the definition and use of **procedures.** These are nothing but DATATRIEVE-11 commands that are generalized and saved with a special name, a procedure name. Later at any time in a DATATRIEVE-11 sequence of commands you can invoke the procedure by calling out its name. This saves you some redundant typing. It also provides a means to establish and distribute more complicated sequences of commands so that everyone within the installation can benefit from them without necessarily having to reinvent them. Chapters 6 and 7 provide more discussion of procedures.

1.2 INTRODUCTION TO DATATRIEVE-11 REPORT WRITER FACILITIES

Once you have successfully invoked DATATRIEVE, you can specify that you wish a certain format of report printed on one of the available devices. The data to be used in the report may come directly from the RMS-11 files or may have been preselected and manipulated through a series of DATATRIEVE-11 commands.

Each report specification is enclosed between a pair of commands: REPORT and REPORT END. We will call this entire grouping an **RW sequence,** where RW refers to the Report Writing facility of DATATRIEVE-11.

More than one Report Writer exercise can be incorporated at a time in a given session. In fact, it is possible to bounce back and forth between inquiry processing and RW processing. The only restriction on intermixing inquiry processing and RW statements in a session is that inquiry processing statements must not occur inside an RW sequence.

1.3 SAMPLE DATA FOR EXAMPLES

Throughout the manual the examples use one set of sample data. This data is depicted in Figure 1-1.

The sample data is based on an inventory of yachts, as kept by Star-Key Marina, Inc. Seven pieces of information are kept for each type of yacht: the manufacturer, the model number, the rig type, the length-over-all, the displacement (or weight), the beam size, and the price.

1.4 DEFINITIONS: FILES, DOMAINS, COLLECTIONS, RECORDS, AND FIELDS

This section defines the terms domain, collection, record, file, and field.

Records are groups of related items of data that are treated as a unit. For example, all the pieces of data describing a model of yacht in the marina could be grouped to comprise that yacht's record.

Each of the individual pieces of data in a record is referred to as a **field.** For example, the yacht's model number, length, and price are all potential fields in the yacht record.

The term **files** is used to refer to the logically related groups of data that are kept by RMS-11. For example, we might put all of the yacht records for the current inventory at Star-Key Marina, Inc., into one file.

Domains are named groups of data containing records of a single type. An RMS-11 domain consists of all the records in a given RMS-11 file. In this case, we could say that all the yacht records for the current inventory are kept in the YACHTS domain. The number of records in any domain may change as new records are stored or old records are erased.

A record **collection** is a subset of a domain. It may consist of no records, one record, or up to all the records in the domain. Using our previous example, we could say that all the yachts manufactured by Grampian could be made to form the Grampian-collection, while those yachts manufactured by Islander could be used to form the Islander-collection. To carry this example one step further, if the inventory is currently out of stock of yachts manufactured by Seaworthy, the Seaworthy-collection will be empty, or **null**.

1.5 INTRODUCING THE DATA DICTIONARY

The Data Dictionary is a location where the definitions for procedures, records, and domains are kept in a standard fashion by DATATRIEVE-11. The Data Administrator will be concerned with the creation and maintenance of Data Dictionary information. Certain users will be able to display certain information from within this dictionary, but only management will be concerned with defining it.

1.6 PROCESSING A FILE AS A COLLECTION

The most important concept to master is collection processing. DATATRIEVE-11 operates on collections of records taken from the files. To get down to the level of record processing you must first use the FIND and SELECT commands to gather the collection and extract desired records. DATATRIEVE provides a **collection cursor** to keep track of your place in the collection. Figure 1-2 illustrates the cursor as a place marker. You will learn how to manipulate the cursor, moving it through the collection, from the first, to possibly the next or the nth record, all the way to the last. Your moves need not always follow a forward direction; you can move about the collection at will.

In most DATATRIEVE-11 operations, the files are never changed, but a great deal of manipulation occurs on the collections. Thus, the collections can be thought of as just a sort of temporary storage, kept for immediate purposes, and then released.

MANUFACTURER	MODEL	RIG	LENGTH OVER ALL	WEIGHT	BEAM	PRICE
ALBERG	37 MK II	KETCH	37	20,000	12	$36,951
ALBIN	79	SLOOP	26	4,200	10	$17,900
ALBIN	VEGA	SLOOP	27	5,070	08	$18,600
AMERICAN	26	SLOOP	26	4,000	08	$9,895
AMERICAN	26-MS	M/S	26	5,500	08	$18,895
BAYFIELD	30/32	SLOOP	32	9,500	10	$32,875
CAPE DORY	28	SLOOP	28	9,000	09	$21,990
CAPE DORY	TYPHOON	SLOOP	19	1,900	06	$4,295
CAPE DORY	25	SLOOP	25	4,000	07	$8,995
CARIBBEAN	35	SLOOP	35	18,000	11	$37,850
CHALLENGER	32	SLOOP	32	12,800	11	$31,835
CHALLENGER	35	SLOOP	35	14,800	12	$39,215
CHRIS-CRAF	CARIBBEAN	SLOOP	35	18,000	11	$37,850
CLIPPER	CM 30	SLOOP	30	3,800	08	$9,500
CLIPPER	CM 32	SLOOP	32	4,500	08	$12,950
EASTWARD	HO	M/S	24	7,000	09	$15,900
GRAMPIAN	26	SLOOP	26	5,600	08	$11,495
GRAMPIAN	28	SLOOP	28	6,900	10	$14,475
GRAMPIAN	34	KETCH	33	12,000	10	$29,675
HUNTER	27	SLOOP	27	6,500	09	$14,999
I. TRADER	37	KETCH	36	18,600	12	$39,500
IRWIN	37 MARK II	KETCH	37	20,000	11	$36,950
IRWIN	25	SLOOP	25	5,400	12	$10,950
ISLANDER	28	SLOOP	28	5,994	10	$15,908
ISLANDER	BAHAMA	SLOOP	24	4,200	08	$6,500
LINDSEY	39	M/S	39	14,500	12	$35,900
RYDER	S. CROSS	SLOOP	31	13,600	00	$32,500
SABRE	28	SLOOP	28	7,400	09	$22,000
SALT	19	SLOOP	25	2,600	07	$6,590
TANZER	26	SLOOP	26	4,350	09	$11,750
TANZER	28	SLOOP	28	6,800	10	$17,500
VENTURE	21	SLOOP	21	1,500	07	$2,823
VENTURE	222	SLOOP	22	2,000	07	$3,564
WESTERLY	CENTAUR	SLOOP	26	6,700	08	$15,245
WINDPOWER	IMPULSE	SLOOP	16	650	07	$3,500

Figure 1-1 Sample Data

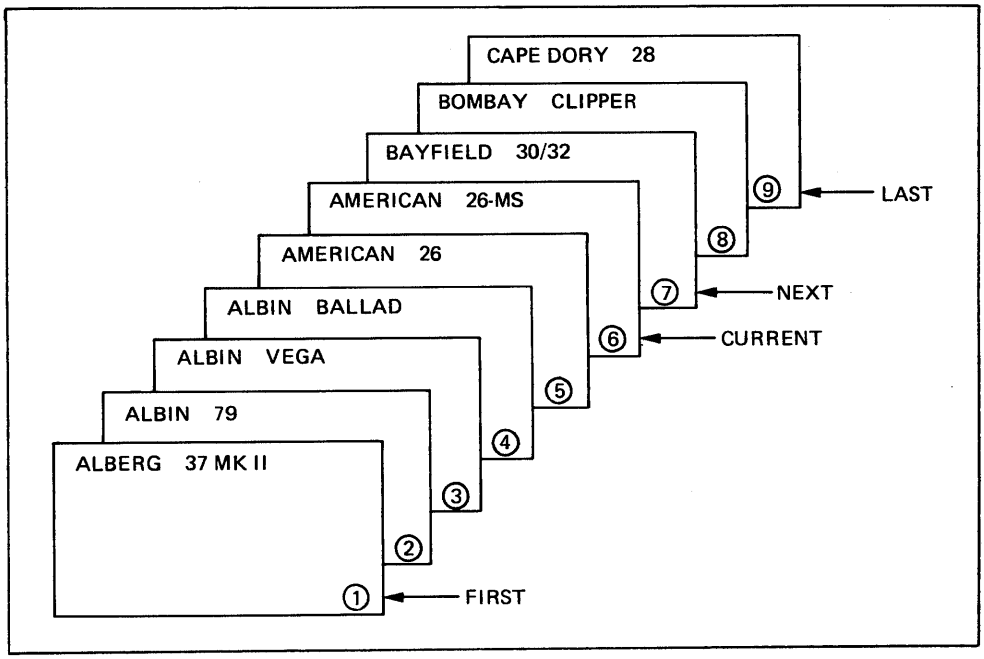


Figure 1-2 Collection Cursor

Just as you would expect, **the FIRST record** of the collection is the first one encountered when it was being established or collected. The record numbered with ① in Figure 1-2 identifies the first record in this example.

The NEXT record is always the one immediately after the CURRENT one. In the figure, **the CURRENT record** is the sixth one, so the NEXT record must be the seventh.

We have been referring to records according to their numeric position in the collection, that is, sixth, seventh, and so forth. This is a proper form of reference, and in the DATATRIEVE-11 language, you will use digits for the position number whenever a format specifies **the nth record.**

The LAST record is the one located at the very end of the collection.

If there happen to be no records at all in a collection, we say the collection is **null,** and the collection cursor will have to be null, too.

Generally, you will work on just one collection at a time, the CURRENT one. However, there is the facility to name a collection and refer back to it later in the same session, if you feel that it has enough importance or probability of reuse. (See Section 4.4.)

Thus, you may find yourself working with a number of collections, one of which is the CURRENT one, the one most recently the object of a FIND command. **Each of the collections has its own collection cursor** and may have a designated **CURRENT record,** as well. However, only one collection may be CURRENT at a time.

The lifetime of any collection is limited to a given session. Collections cannot be saved, nor can they be shared by several users. However, you can always reproduce a **collection of identical characteristics** (although not necessarily identical records since the files can change in the interim). You simply repeat the sequence of FIND command(s) used initially to establish and/or refine the collection. If you find this to be a frequently recurring requirement, perhaps your installation should define the steps as a procedure. (See Chapter 6 and Section 7.2.)

Collections may be ordered on command. When you sort a collection, you may well change the position a particular record holds.

1.7 PASSWORDS

Certain commands you will use in DATATRIEVE have an optional password parameter. This password parameter prevents sensitive data from being corrupted by unauthorized persons. If you are authorized, you will be given a password that will "unlock" certain domains, records, and/or procedures.

A password can contain from one to ten characters, such as, STAR-KEY or SUNSHINE. You may enter a password directly with the command or enter an asterisk (*) instead. If you enter an asterisk, DATATRIEVE invokes a NO-ECHO operation. When DATATRIEVE prompts for a password the password you enter will not appear on the terminal. This prevents unauthorized persons from looking over your shoulder to learn a password. A more detailed discussion of passwords and data security intended for the Data Administrator appears in Section 7.2.

CHAPTER 2

USING DATATRIEVE-11

In this chapter you will learn how to know when DATATRIEVE-11 is ready to accept a command, how to terminate a session, what to expect in the error messages, and the general mechanics of entering commands. However, the commands are described in later chapters.

2.1 INVOKING DATATRIEVE-11

Once you have logged onto TRAX and instructed the operating system to invoke DATATRIEVE-11, you will see the prompt DTR> on your terminal. This means you are ready to enter a DATATRIEVE-11 command.

2.2 STOPPING

You may leave a DATATRIEVE-11 session at any time by means of the EXIT command (Section 4.2) or the CTRL Z (^Z) key combination.

2.3 ERROR MESSAGES

Because you operate in interactive mode, you get immediate notification when an error is detected. This gives you the following options:

- revise your work (if this was a major error),
- correct the problem in the last command, or
- terminate the session.

All error messages are listed in alphabetical order in Appendix D. You will observe that the messages are sentence-like and quite descriptive. There will be no need to "decode" an error code, as you may have done in some other languages.

2.4 ENTERING COMMANDS

As you learn the commands in the following chapters, you will find certain common rules apply to entering them. For example, you can only enter a command after the prompt DTR>. Furthermore, you must terminate a command with a final carriage return. Once a command is entered, DATATRIEVE-11 begins execution immediately. If you have made

an error, but no significant changes were made to the collection, you may generally simply reissue the command in corrected form to obtain the desired results.

CHAPTER 3

SYNTAX RULES FOR DATATRIEVE-11 COMMANDS

This chapter presents all the basic rules that apply to the construction of DATATRIEVE-11 commands. You must understand the significance of spelling, spacing, syntax symbols, and sequence when using the commands. You will learn what characters are permitted in the commands, how to form names and literals, how to insert comments, and how to continue and terminate command lines. You will also see how to interpret syntax symbols, how to delimit (separate) the parts of a command, and what the keywords in the commands are.

3.1 SEQUENCE OF COMMANDS

There are just a few restrictions regarding the sequence of the commands. EXIT is always the last command. READY is often the first command, but it could be preceded by others. Beyond that, you can intermix commands with each other, and likewise, you can intermix RW commands with each other, with a great deal of freedom. However, as mentioned in Chapter 1, once you begin the specification of a report, you cannot introduce other commands until the REPORT END command is processed.

3.2 SPELLING

Spelling is critical: you must enter commands as they are shown in their "Format" description. As a general rule, command names or elements may not be abbreviated unless the particular command description tells you to do so.

3.3 SPACING

DATATRIEVE-11 is a free-form language, which means that you may have any number of intervening spaces or tab characters between language elements. Since DATATRIEVE-11 is English-like, the examples generally show single spaces between elements.

3.4 SPECIAL SYMBOLS -- BRACKETS, BRACES, ELLIPSIS, UPPERCASE, AND LOWERCASE

The preface introduced these special symbols, but it may be helpful to supplement the definitions with examples. Special symbols are used in the syntax descriptions to clarify proper usage of the command elements. While they are discussed individually here, you will note

in Figure 3-1 that they can be combined in a given syntactical expression in numerous ways, even nesting.

Syntax Brackets (**[ ]**) denote items that are optional. If more than one item is enclosed in brackets, you may chose one (or none) of the items.

Square Brackets ([ ]) denote the bracket symbols on the keyboard and are required where shown.

Braces ({}) enclose one or more items that are required. If more than one possible entry is shown, you must include one of the choices, but you can specify no more than one.

Ellipsis (...) imply that you can choose to repeat a version of the immediately preceding command element. In DATATRIEVE-11 there are no restrictions on the number of repetitions, unless you are explicitly told so. Watch for punctuation characters needed to serve as separators for the repetitions. Note that repetitions only occur at your option.

Uppercase letters denote required words when underlined. When not underlined, these words are optional.

Lowercase letters are used with generic terms that must be replaced by the appropriate names or values.

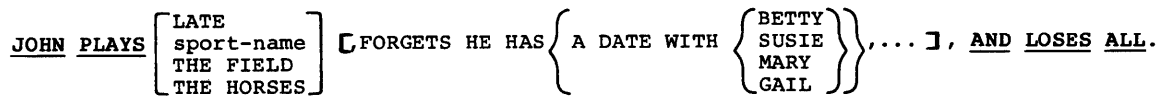
Underlining (__) denotes that the uppercase word is required in every use of the command or statement, unless enclosed in outer brackets.

Figure 3-1 illustrates a fairly complex syntactical structure, employing all these symbols in an English sentence rather than a command. Observe how the symbols can be nested. Wherever nesting occurs, the meaning of the outermost pair of symbols prevails. Thus, when a seemingly required item inside braces occurs inside a pair of outer brackets, the entire group is considered to be optional.

As you study Figure 3-1, notice the large number of legal sentences that can be constructed to conform to the syntax; only a few possibilities are shown here.

Later, when you see these same symbols used in actual DATATRIEVE-11 command definitions, keep in mind the large number of combinations of command elements they imply. Do not be concerned that you will need to enter these special symbols as part of the command. They only exist as part of the description of the command, as a sort of code to indicate a number of ways you can choose to construct each command from its parts.

Assume the following syntactical structure:



The following sentences are among the many possibilities the user could construct:

JOHN PLAYS, AND LOSES ALL.

None of the optional bracketed items is included; only the required underlined words in uppercase are kept.

JOHN PLAYS THE HORSES, AND LOSES ALL.

An object word for PLAYS is supplied, but the bracketed FORGETS clause is omitted.

JOHN PLAYS, FORGETS HE HAS A DATE WITH MARY, AND LOSES ALL.

The optional bracketed items after PLAYS are omitted, but the FORGETS clause is added.

JOHN PLAYS THE FIELD, FORGETS HE HAS A DATE WITH SUSIE, AND LOSES ALL.

All options included except the repetition.

JOHN PLAYS BASEBALL, FORGETS HE HAS A DATE WITH BETTY, A DATE WITH GAIL, AND LOSES ALL.

Observe the substitution of BASEBALL for the lowercase word sport-name and the repetition at the site of the ellipsis.

Figure 3-1 Illustrating Syntactical Symbols

3.5 CHARACTER SET

The character set consists of the letters A through Z, numbers 0 through 9, and certain special symbols. See Table 3-1.

Table 3-1
DATATRIEVE-11 Character Set

A	T	;
B	U	:
C	V	(
D	W	)
E	X	[
F	Y	]
G	Z	+
H	0	-
I	1	*
J	2	/
K	3	"
L	4	'
M	5	=
N	6	<
O	7	>
P	8	$
Q	9	!
R	.	&
S	,	

3.6 COMMENTS

You may introduce a comment anywhere in your session if you like, by keying an exclamation point character (!), following it with text, and terminating with a carriage return (RET). The text may consist of any characters in the character set including the exclamation point.

Comments may begin anywhere on an input line and are **not** required to start at the beginning of the line.

The value of comments diminishes in an interactive environment. You may find you wish to include them in indirect command files (Section 7.1). However, you will find that if you put them in your procedure definitions (Section 7.1.3), they are not retained in the Data Dictionary.

3.7 NAMES

Names are constructed from the letters A through Z, numbers 0 through 9, and the special character, hyphen (-). The construction of names is subject to five rules:

1. Every name must **begin** with a letter.

2. Names must **never end** with a hyphen.

3. Names can only be continued from the end of one input line to the beginning of the next by means of a hyphen, as will be explained in Section 3.9.

4. You must be careful when creating names to avoid the DATATRIEVE-11 keywords. (See Section 3.11.) For your convenience, these special names are also listed alphabetically in Appendix C.

5. Names of records, domains, procedures, and collections are kept in the Data Dictionary and are subject to a size limitation of 30 characters.

3.8 LITERALS

There are two categories of literals: alphanumeric and numeric. Thus, different rules apply for specifying each.

Alphanumeric literals may consist of alphabetic, numeric, and special characters. All alphanumeric literals must be enclosed in quotation marks ("). For example, if you use the phrase BUILDER EQUAL "O'DAY", "O'DAY" is an alphanumeric literal. Similarly, in MODEL EQUAL "37 MK II", "37 MK II" is an alphanumeric literal. To include a quotation mark as a legal character within an alphanumeric literal, use two consecutive quotation marks. For example, you might want to display an alphanumeric message, "CAPT. JACK SAYS, ""TIME TO REORDER""", such that TIME TO REORDER would appear in quotes.

Combinations using the CTRL (^) character are not permitted as special characters in an alphanumeric literal.

Numeric literals consist of digits and an optional decimal point (.), and they are written without quotation marks. For example, you may use the phrase, BEAM EQUAL 12. Here, 12 is a numeric literal. If the decimal point is included, it must not be the first character in the number. Thus, 0.5 is permitted as a numeric literal, but .5 is not.

3.9 CONTINUATION LINES -- CARRIAGE RETURN (RET)

Some of the DATATRIEVE-11 statements can become longer than a single line length. DATATRIEVE-11 permits continuation lines. You simply enter a carriage return (RET) as you approach the end of each line and continue the statement on the next line or lines.

DATATRIEVE is able to recognize all the possible elements in a command and to interpret whether there should be more input to follow any given carriage return. As proof of this, you may see DATATRIEVE-11 responses that say: [LOOKING FOR element-name] where element-name is the syntactical element that DATATRIEVE-11 has determined should follow the preceding portion of a command.

CAUTION

If you enter a carriage return at a point where a logical termination could occur, DATATRIEVE assumes it has found the termination and looks no further. You may use the hyphen (-) to remove all ambiguity: a hyphen preceding a carriage return always implies a command continuation to DATATRIEVE. However, **you should not continue more than two successive lines with hyphens.**

Also note that the hyphen provides the only proper way to continue a literal or name from one line to the next. To continue a hyphenated name, such as SMALL-YACHTS, you could specify any of the following forms, or variations of them.

```
SMALL- RET
-YACHTS

     or

SMALL-- RET
YACHTS

     or

SM- RET
ALL-YACHTS
```

Figure 3-2 illustrates these concepts with an ordinary English sentence.

```
Assume the sample sentence:

              JOHN WENT TO THE OFFICE AND RAN INTO MARY

Case 1

JOHN (RET)                      Subject, but no verb, so continue.
WENT (RET)                      Missing the object, continue.
TO THE OFFICE (RET)             Sentence appears complete, stop.
AND RAN INTO MARY (RET)         Attempt to process as a new sentence.

        To remedy the premature termination, try this instead:

Case 2

JOHN (RET)
WENT (RET)
TO THE OFFICE AND (RET)         Now it is obvious there is more.
RAN (RET)
INTO (RET)
MARY (RET)                      Structure appears complete, stop.

            You could have used a hyphen (-) after "OFFICE"
                        to indicate continuation:

Case 3

JOHN (RET)                      Subject, but no verb, so continue.
WENT (RET)                      Missing the object, continue.
TO THE OFFICE- (RET)            Hyphen before return, continue.
AND RAN INTO MARY (RET)         Structure appears complete, stop.
```

Figure 3-2 Example of Carriage Return Interpretation

3.10 COMMAND TERMINATION--SEMICOLON

Since DATATRIEVE recognizes the command elements, it is not necessary to terminate each command with some form of punctuation such as a semicolon. However, the option exists for you to specify the semicolon, if you desire, as the formal designation of command end. Two commands, DELETE and DEFINE, require the semicolon.

If you want to issue more than one command per input line, you must use the semicolon to separate the commands. Furthermore, DATATRIEVE will not begin to process any commands on the same input line until a carriage return is detected.

Spaces or tab characters at the end of a line are ignored.

3.11 KEYWORDS

Keywords are language elements that have a specific meaning to DATATRIEVE-11. If you use them in any other context, you may confuse DATATRIEVE about your intentions. Thus, it is a good policy to avoid the use of these words as names of domains, procedures, records, fields, and collections.

The full list of keywords is reproduced in Table 3-2 and Appendix C.

Table 3-2
Keywords

ABORT	DESC	LESS-EQUAL	READ
ADVANCED	DESCENDING	LESS-THAN	READY
ALL	DISPLAY	LINES-PAGE	RECORD
AND	DOMAIN	LT	RECORDS
ASC	DOMAINS	MAX	RELEASE
ASCENDING	EDIT-STRING	MAX-LINES	REPEAT
AT	ELSE	MAX-PAGES	REPORT
AVERAGE	END	MIN	REPORT-HEADER
BEGIN	END-PROCEDURE	MODIFY	REPORT-NAME
BETWEEN	EQ	NE	SELECT
BOTTOM	EQUAL	NEW-PAGE	SEPARATE
BT	ERASE	NEW-SECTION	SET
BY	EXCLUSIVE	NEXT	SHARED
CHARACTER	EXIT	NO	SHOW
CLOSE	EXTEND	NO-DATE	SHOWP
COL	FILLER	NO-NUMBER	SIGN
COLLECTIONS	FIND	NOT	SKIP
COLUMN	FINISH	NOT-EQUAL	SORT
COLUMN-HEADER	FIRST	NUMBER	SORTED
COLUMNS-PAGE	FOR	OF	SPACE
COMP	GE	ON	STORE
COMP-1	GREATER-EQUAL	OPEN	TAB
COMP-2	GREATER-THAN	OR	THE
COMP-3	GT	PAGE	THEN
COMP-5	HELP	PIC	TOP
COMP-6	IF	PICTURE	TOTAL
COUNT	IN	PRINT	TRAILING
CURRENT	INCREASING	PROCEDURE	UIC
DATE	IS	PROCEDURES	USAGE
DECREASING	JUSTIFY	PROTECTED	USING
DEFINE	LAST	PW	VERIFY
DEFINEP	LE	QUERY-HEADER	WITH
DELETE	LEADING	QUERY-NAME	WRITE
DELETEP			

3.12 DELIMITERS

Delimiters are required around keywords, names, and literals. These delimiters are separators, and you may select any of the three choices:

- the space,
- the tab character, and
- the carriage return.

Any given DATATRIEVE-11 session may use all three delimiters many times in many different places.

3.13 SEQUENCE OF COMMAND ELEMENTS

Each possible command element has a position in the command format. This relative location is always illustrated when the command formats are given. It is extremely important that you obey this sequence when you formulate your commands. When you decide to omit one of the optional elements, just leave its relative position in the command empty, but proceed to the remaining elements in the left-to-right sequence.

CHAPTER 4

COMMANDS FOR INQUIRY AND UPDATE

This chapter presents the commands that are part of the DATATRIEVE-11 facility for inquiry and update.

4.1 STARTING A DATATRIEVE-11 SESSION WITH READY

As soon as you have successfully invoked DATATRIEVE-11 and have obtained the DTR> prompt, you are able to enter your first command. If you have been away from the system for a while, you may wish to refresh your memory by means of the HELP command that is described as the very last item in this chapter. (See Section 4.14.)

The operations you are about to attempt will probably require the use of one or more domains. Before you can use any domain you must ready it, so you will probably want to issue a READY command as one of your commands, if not the first. The purpose of readying a domain is to claim it for your use. This means that if other users want to use it too, they may be subject to certain restrictions. It is important that any shared resource be handled according to certain rules.

The data processing management at your facility will have given considerable thought to whether certain resources may be shared. One issue is whether to restrict the ability to change any of the record fields in a file to only certain users. Another concern is whether to permit sharing at all. By the time you are ready to run your first DATATRIEVE-11 request, these issues should be resolved.

The types of access modes and modifiers are described in Table 4-1.

Table 4-1
Types of File Access Modes and Modifiers

Modifier	Meaning
SHARED	Any other user may have concurrent access to the records in this domain, for any purpose.
PROTECTED	Any other user may access, but not concurrently write, modify, or extend (add) records in this domain.
EXCLUSIVE	No other user may concurrently execute a READY for this domain -- for any type of access.

Mode Name	Meaning
READ	Retrieve only.
MODIFY	Retrieve and change only.
WRITE	Retrieve, change, add, or delete.
EXTEND	Add only.

NOTE: The Data Administrator will have some control over which modes of access (READ, MODIFY, WRITE, or EXTEND) are permitted for each domain. (See Chapter 7.) However, you are the only one controlling the acceptance of the access modifiers (SHARED, PROTECTED, or EXCLUSIVE).

READY

Purpose

This command informs the system that you expect to reference a certain domain. At the same time, you can declare how you intend to use the domain. For example, you can indicate that you just want to read from it, or that you expect to modify it, or that you will want exclusive use of it.

Format

```
READY  domain-name-1 [(password-str-1)] [SHARED     ] [READ  ]
                                        [PROTECTED  ] [MODIFY]
                                        [EXCLUSIVE  ] [WRITE ]
                                                      [EXTEND]
```

1. You must supply a domain-name, and it must name a domain that exists in the Data Dictionary and on the RMS-11 files.

2. You may optionally wish to specify the access modifiers and/or access modes that you expect to use. See Table 4-1 for a description of each.

3. In some cases it may be necessary to know a password to gain access to a particular domain. If you are given a password to specify, be sure to enclose it in parentheses. If you enter an asterisk (*) for your password, you will be prompted to supply a password. In this mode your password specification will not be echoed on your terminal device to ensure greater password security.

4. If none of the access modifiers or modes is specified, it is assumed you wish to ready the domain for PROTECTED READ.

Requirements

None.

Restriction

You cannot issue a READY command in a procedure definition (Section 7.1.3), a BEGIN-END Block (Section 4.13.1), or THEN statement (Section 4.13.4).

Prompts & Responses

None, unless you specify an asterisk (*) for the optional password. This will cause the following prompt to appear:

```
PLEASE SUPPLY PASSWORD:
```

Effects

1. The domains are checked for existence and availability.

2. DATATRIEVE checks an internal password table for the named domain to ensure that you are allowed the type of access mode requested for that domain.

3. The record definition for this domain is also checked for existence.

4. You can only ready one domain at a time with this command. However, you can use the command multiple times in a session to ready a number of domains at once.

5. If you READY the same domain again but give different access qualifiers, the last READY to be executed prevails.

Examples

The following example readies the domain YACHTS (yacht records) for exclusive use by this user and allows the domain to be read, modified, erased, and/or extended. Since this is the full range of access capabilities and the most restrictive access modifier, it is logical to assume a password is necessary. Your password is FAIRWINDS.

```
DTR> READY YACHTS (FAIRWINDS) EXCLUSIVE WRITE; (RET)
```

The following example illustrates how to ready the same domain for shared read-only access when no password is required:

```
DTR> READY YACHTS SHARED READ; (RET)
```

Hints

You should adopt the practice of only readying those domains you really intend to use. However, since very little can be done in a DATATRIEVE session unless you have performed a READY, you will probably also want to adopt the habit of issuing the READY command(s) at the very beginning of the session.

It is also good practice to be as unrestrictive in the access mode specification as you can afford to be. For example, if you insist on demanding that the domains you use be kept EXCLUSIVE, you can hurt system performance for other users who could otherwise be sharing the domain.

Finally, if you only intend to retrieve records from the domain, keep in mind that READ mode is more efficient for this purpose than either WRITE or MODIFY mode. Selecting either of the latter two just to retrieve is overkill.

You can confirm that a domain has been readied by means of the SHOW READY command described in Section 4.3.

4.2 ENDING A DATATRIEVE-11 SESSION -- EXIT

If at any time you wish to leave your terminal or stop a DATATRIEVE session, you should know how to use the EXIT command. This command lets you terminate any session gracefully. It is the proper way to finish all sessions, and assures you that no unauthorized user can gain access to domains you have readied from your terminal. It is also the most considerate action to take so that domains are freed up for other users to gain access. It is good practice to perform an EXIT command before you leave the terminal.

EXIT

Purpose

When you execute this command, you close a DATATRIEVE-11 session.

Format

EXIT

Requirements

You may issue this command at any time in a DATATRIEVE-11 session.

Prompts & Responses

No prompting messages appear, but you will be informed that the session is terminated. The response assumes the form:

BYE

Effects

1. DATATRIEVE-11 processing is terminated.
2. Any domains that had been readied are automatically closed, and all collections established in this session are released. This is true for both the named and unnamed collections.
3. You cannot issue another DATATRIEVE-11 command until you reinvoke DATATRIEVE-11.

Example

DTR>EXIT; (RET)

BYE

Hint

You can accomplish the functions of the EXIT command with a CTRL Z (^Z).

4.3 GETTING AN OVERVIEW OF YOUR DATA -- SHOW

If at any time you begin to wonder about the contents of a particular domain or record, you may want to issue the SHOW command. Each installation can provide its users with current printouts depicting the domain and record definitions that will be used. However, it is the SHOW command that is used to generate them, and it is always helpful to know how to use this command yourself.

In some installations domain and record information will be quite static, while in others, there may be frequent changes. When in doubt, it is easy enough to request a copy with the SHOW command.

Furthermore, since a collection is a temporary formation, you may want to monitor your collections yourself. All these capabilities, and more, are possible with the SHOW command.

SHOW

Purpose

This command requests a copy of the summary details maintained in the Data Dictionary for your data. You may request to see all the information maintained, or you may just request to see specific portions of it. For example, you might limit your review to the domain details, or you might restrict it even further to only one particular domain, collection, or record.

Format

SHOW show-item-1 [,show-item-2...]

1. You must supply at least one show-item, and it must be one of the following possible choices:

```
{ PROCEDURES                                    }
{ DOMAINS                                       }
{ COLLECTIONS                                   }
{ RECORDS                                       }
{ ALL                                           }
{ CURRENT                                       }
{ READY                                         }
{ procedure-name-1 [(password-str-1)]           }
{ domain-name-1 [(password-str-2)]              }
{ record-name-1 [(password-str-3)]              }
{ collection-name-1                             }
```

   Observe that three of the possible show-items may take an optional password string.

2. You may optionally wish to request additional information by appending one or more show-items.

3. If you specify PROCEDURES, DOMAINS, COLLECTIONS, or RECORDS, your display will show the names of the defined procedures, defined domains, established collections, or defined records, respectively, as they exist at this point in time.

4. If you specify ALL, your display will contain the names of all the defined procedures, defined domains, established collections, and defined records.

5. If you specify CURRENT, your display will provide only the information describing the CURRENT record collection.

6. If you specify READY, your display will identify the names of all domains that are presently ready, their file type, access mode, and access modifier.

7. If you name a specific procedure, domain, collection, or record, you will be given information regarding just that particular item, provided it is not protected from your view. Procedures, domains, and records can be protected so that only users with password knowledge can view or change them.

8. If you want, you may separate each of the show-items by a carriage return (RET). DATATRIEVE will execute each part of the command up to each carriage return, and will then come back to look for more show-items.

Requirements

The format of the SHOW command that specifies a resource-name (where domains, records, and procedures are the resources) requires that the issuer have access privileges. These checks are part of the data protection features described in Chapter 7.

The only other prerequisite concerns the specification CURRENT. You should only specify SHOW CURRENT when you have established a CURRENT collection with the FIND command. Section 4.4 describes the various FIND commands.

Restriction

You must not use this command within a procedure. (See Section 7.1.3.)

Prompts & Responses

If you specify an asterisk (*) for the optional password, you will be prompted to specify the full password by the message:

PLEASE SUPPLY PASSWORD:

Otherwise, no prompting messages appear, but you will find the display occurring automatically on your terminal device.

Effects

1. If you are deemed to have access privileges for a named resource (such as a particular record, domain, or procedure definition), the results of the request appear, in the order of the show-items, on your terminal device. The format of the output is explained by Figure 4-1.

2. For the other types of show items, no checking is necessary; the results appear, in the order of the show-items, on your terminal device. See Figure 4-1 for a sample of the printout format.

3. There is no effect on the collection cursor.

Examples

The following example displays the names of all the procedures, domains, collections and records.

DTR>SHOW ALL; RET

The following example illustrates how to display just the record definition for the record named COBOL-YACHTS. Assume that your password is SUNNY-DAY.

```
DTR>SHOW COBOL-YACHTS (SUNNY-DAY); (RET)
```

Hints

If you are going to use a password with the SHOW resource-name form of the command, consider specifying just an asterisk in parentheses (*). DATATRIEVE will respond by requesting the full password, but will not echo the characters you enter. This affords greater password security.

```
DTR>SHOW DOMAINS
DOMAINS:
        THINGS-TO-DO
        DICT-DOMAIN
        FORTRAN-YACHTS
        COBOL-YACHTS
        BOATS
        FEATURES
        DATA-TYPES
        MESSAGES
        RMS-MSGS
        COMP-SCALE
        MESSAGES-SHIP
        YACHTS

DTR>SHOW YACHTS
DOMAIN YACHTS
 USING YACHT ON
YACHT.DAT;

DTR>SHOW YACHT
RECORD YACHT
 USING
01 BOAT.
  03 TYPE.
     06 MANUFACTURER PIC X(10)
        QUERY-NAME IS BUILDER.
     06 MODEL PIC X(10).
  03 SPECIFICATIONS
     QUERY-NAME SPECS.
     06 RIG PIC X(6).
     06 LENGTH-OVER-ALL PIC XXX
        QUERY-NAME IS LOA.
     06 DISPLACEMENT PIC 99999
        QUERY-HEADER IS "WEIGHT"
        EDIT-STRING IS ZZ,ZZ9
        QUERY-NAME IS DISP.
     06 BEAM PIC 99.
     06 PRICE PIC 99999
        EDIT-STRING IS $$$,$$$ .;

DTR>READY YACHTS; FIND THE YACHTS WITH PRICE >0;
[35 RECORDS FOUND]
DTR>SHOW CURRENT
COLLECTION CURRENT
        DOMAIN: YACHTS
        NUMBER OF RECORDS: 35
        NO SELECTED RECORD

DTR>SHOW COLLECTIONS
COLLECTIONS:
        CURRENT

DTR>SHOW READY
READY DOMAINS:
        YACHTS: RMS INDEXED, PROTECTED READ

DTR>SHOW POUNDS-PER-FOOT
PROCEDURE POUNDS-PER-FOOT
DISP/LOA ("POUNDS"/"PER"/"FOOT") USING Z(5)
```

Figure 4-1 Sample Displays Resulting from the SHOW Command

```
DTR>SHOW ALL;
DOMAINS:
        THINGS-TO-DO
        DICT-DOMAIN
        FORTRAN-YACHTS
        COBOL-YACHTS
        BOATS
        FEATURES
        DATA-TYPES
        MESSAGES
        RMS-MSGS
        MESSAGES-SHIP
        YACHTS
RECORDS:
        THING-TO-DO
        DICT-REC
        FORTRAN-YACHT
        COBOL-YACHT
        FEATURE-RECORD
        DATA-TYPE-RECORD
        RMS-MSGS-RECORD
        MSG-RECORD
        MSG-SHIP-REC
        YACHT
PROCEDURES:
        REPORT
        POUNDS-PER-FOOT
        LOA-REPORT
        MESSAGE-REPORT
        FEATURE-REPORT
        MESSAGE-LISTINGS
        LIST-MESSAGES
        PRICE-PER-POUND
        PRICE-INCR
COLLECTIONS:
        SLOOPS (ALSO CURRENT)
READY DOMAINS:
        YACHTS: RMS INDEXED, PROTECTED READ
```

Figure 4-1 (Cont.) Sample Displays Resulting from the SHOW Command

4.4 GETTING A COLLECTION TOGETHER -- FIND

The first step toward manipulating a collection is to "establish" it. This process pulls the collection together and gives it an initial shape or definition. With the three versions of the FIND command described in this section, you can choose to:

- establish a new collection,
- refine an existing CURRENT collection, or
- name and/or sort a collection while establishing it.

4.4.1 Establishing the Collection

Before you can perform a number of DATATRIEVE operations, you must establish a collection of records for processing. If you wish, this collection may consist of all the available records in a domain. Or, you may restrict the collection to just a subset of the domain's records. You use the FIND command to establish collections. You may establish a number of collections in a given DATATRIEVE session through multiple usages of the FIND command. The last collection to be established is referred to as the CURRENT collection.

FIND

Purpose

This command pulls together or establishes a collection of records. This collection may consist of zero, one, or more records. In fact, it may contain all the available records. As a result of the FIND command, the collection becomes the CURRENT collection. Nothing else is done to the resultant collection; it is simply formed so that you can perform other operations on it such as SELECT, SORT, MODIFY, or PRINT.

Format

FIND domain-name-1 [**WITH** condition]

1. You must supply a domain-name, and it must name a domain that you have already readied.

2. You may optionally wish to restrict the records that are collected by specifying the word WITH followed by a condition. Section 4.5 provides all the rules necessary to formulate this Boolean expression. (Observe that a variety of compound Boolean expressions are permitted for the condition, so that you can effectively specify multiple conditions.)

Requirements

You must have already performed a successful READY on this domain with any of the access modes of READ, WRITE, or MODIFY, but not EXTEND. Section 4.1 describes the READY command.

Prompts & Responses

No prompting messages appear, but you will be informed how many records met your selection criterion. The response assumes the following form:

[n RECORD(s) FOUND]

where n is a number representing the number of records found.

Effects

1. The records are collected in the order in which they are found. You should not infer that there will be any order to the collection.

2. The collection just established by this command is now the CURRENT collection.

3. The collection cursor is left null.

Examples

The following example collects all 113 records in the domain YACHTS.

```
DTR>FIND YACHTS; (RET)
[113 RECORDS FOUND]
```

The following example illustrates how to collect only those records in the domain YACHTS that have an overall length that exceeds 32 feet and cost less than $50,000.

```
DTR>FIND YACHTS WITH LENGTH-OVER-ALL >32 AND (RET)
DTR>PRICE < 50000 (RET)
[36 RECORDS FOUND]
```

Note the use of a compound Boolean expression to check for two concurrent conditions.

Also note in this example that the first (RET) follows the logical operator AND so that there can be no confusion that the command has ended. Only when the second (RET) is detected will DATATRIEVE decide that the end of the command has been reached.

4.4.2 Refining the Collection

If you have established a collection, but after reviewing it or considering the number of records in it, realize you wish to restrict it to fewer records, you may want to "refine" it.

Although you could probably accomplish the same objective by revising your last FIND domain WITH condition command, you may want to start processing with the collection at hand, instead. This is, of course, the CURRENT collection, and it usually represents far fewer records than would the whole domain. Thus, time savings are possible when you rework an existing collection versus searching the entire domain again. The FIND CURRENT WITH condition command does just that.

FIND CURRENT

Purpose

This command generally applies a restrictive set of conditions to the CURRENT collection to produce a new CURRENT collection that is more explicitly selected. Although the specification of a condition is not mandatory, this form of the FIND command has no meaning unless a condition is given.

Format

FIND CURRENT [WITH condition**]**

1. You must have established a collection as the CURRENT collection.

2. You may optionally wish to restrict the records that are collected by specifying the word WITH followed by a condition. Section 4.5 provides all the rules necessary to formulate this Boolean expression. (Observe that a variety of compound Boolean expressions are permitted for the condition so that you can effectively specify multiple conditions.)

3. If no condition is specified, you have accomplished nothing except to make the cursor null since the resultant CURRENT collection will be identical to the initial one.

Requirements

You must have already successfully established a collection as the CURRENT collection.

Prompts & Responses

No prompting messages appear, but you will be informed how many records met your selection criterion. The response assumes the following form:

[n RECORD(s) FOUND]

where n is a number representing the number of records found.

Effects

1. The records are collected in the order in which they are found. You should not infer that there will be any order to the collection.

2. The collection just established by this command is now the CURRENT collection.

3. The collection cursor is left null.

4. The previous collection, unless named, is released.

Examples

The following example assumes you have collected all 113 records in the domain YACHTS into the CURRENT collection and now wish to refine the collection to include only those records of yachts manufactured by Caribbean.

```
DTR>FIND CURRENT WITH BUILDER EQ "CARIBBEAN"; (RET)
[1 RECORD FOUND]
```

The following example assumes you have just established as the CURRENT collection only those 59 records in the domain YACHTS that have an overall length less than 32 feet and a price less than $50,000. Now you must refine that collection to include only those with a displacement (weight) less than 7000 pounds.

```
DTR>FIND CURRENT WITH DISPLACEMENT < 7000; (RET)
[38 RECORDS FOUND]
```

4.4.3 Naming and Sorting the Collection

While you frequently use collections as temporary storage and may only need to use one collection (the CURRENT one) at a time, sometimes it is helpful to be able to give a collection a name. Once a collection carries a name, you can set it aside, establish other collections as the CURRENT one, at their separate times, and yet always return to process the named collection, when necessary.

You may also want to sort the collection. Although the SORT command described in Section 4.7 does this as its sole function, you may if you wish, sort the collection while establishing it. In this most powerful version of the FIND command you can establish, name, and sort a collection all at once.

Several points are extremely important regarding named collections:

- they only exist for your purposes during this session,
- they cannot be saved, and
- every type of normal or abnormal exit releases them.

You need never be concerned that the name you select for your collection might duplicate some other user's. Two concurrent users can use the same collection name without conflict because collections can never be shared. Your only concern should be that if you name more than one collection in a session, you might not keep your collection names unique. If in doubt, issue a SHOW COLLECTIONS command to determine the names you have used.

Refer to Section 3.7 for the rules for names.

FIND rse

Purpose

This command gives you the power to establish a collection, refine a collection, extract a subset of the collection, sort the collection, and/or name a collection. If you examine the command syntax carefully, you will see that this version of the FIND command is actually a superset of the previous two versions discussed. All this capability is possible through the record-selection-expression, or rse. While the rse is fully described in a separate section (see Section 4.5), this command offers a practical introduction to the key features of the rse.

Format

FIND record-selection-expression

where the record-selection-expression (rse) assumes the following form:

```
[ALL      ]
[FIRST n  ] [collectn-name-2 IN] {CURRENT       } [WITH conditn]
                                 {collectn-name-3}
                                 {domain-name-1  }

[SORTED BY key-1[,key-2...]]
```

The record selection expression follows all the rules presented in Section 4.5.3.

1. You must supply either a domain-name, a collection name, or CURRENT, and that identifier must pertain to a domain that you have already readied.

2. If you specify FIRST n, n can be any value expression that resolves into an integer representing the upper bound of the number of records found in the domain, CURRENT collection, or collectn-name-3.

3. If you specify the optional ALL, you obtain all records in the named collection or domain.

4. If you specify collectn-name-2, you will give the new collection a name. Be sure the name you assign does not conflict with a keyword (Table 3-2) or one of the existing names for a defined record, domain, or procedure (use SHOW ALL, if necessary to check).

5. You may optionally wish to restrict the records that are collected by specifying the word WITH followed by a condition. Section 4.5 provides all the rules necessary to formulate this Boolean expression. (Observe that a variety of compound Boolean expressions are permitted for the condition, so that you can effectively specify multiple conditions.)

6. The sort-keys assume the following form:

```
[ ASC [ENDING]  ]
[ DESC [ENDING] ]  field-name-1
[ INCREASING    ]
[ DECREASING    ]
```

Requirements

You must have already performed a successful READY on the source domain. Section 4.1 describes the READY command.

If you specify CURRENT, there must be an established CURRENT collection, formed by a previous FIND command.

Prompts & Responses

No prompting messages appear, but you will be informed how many records met your selection criterion. The response assumes the following form:

[n RECORD(s) FOUND]

where n is a number representing the number of records found.

Effects

1. The records are collected in the order in which they are found. You should not infer that there will be any order to the collection unless you include the optional SORTED BY clause.

2. If you specify the FIRST n clause, you obtain a subset of the collection as the final act **after** any restricting and sorting have occurred.

3. If the number specified for n in the FIRST n clause exceeds the number of records that can be found, the maximum will be found.

4. The collection just established by this command is now the CURRENT collection.

5. If you specify collectn-name-2, it becomes the name of the collection you have created.

6. The collection cursor is left null.

Example

The following example creates a new collection of ten records named the TEN-BEST-BUYS from the domain of YACHTS by restricting them to the sloops longer than 28 feet that weigh at least 7000 pounds and cost up to $35,000. It then sorts them into increasing sequence by price and skims off the first ten records.

```
DTR>FIND FIRST 10 TEN-BEST-BUYS IN YACHTS WITH (RET)
DTR>RIG="SLOOP" AND LOA > 28 AND (RET)
DTR>DISPLACEMENT > 7000 AND PRICE BETWEEN 1 AND (RET)
DTR>35000 SORTED BY INCREASING PRICE; (RET)
[10 RECORDS FOUND]
```

Hint

If you use an existing collection name, the existing collection is supplanted by the one being established.

4.5 RESTRICTING THE SEARCH WITH RECORD SELECTION EXPRESSIONS

In learning the FIND command in the previous three sections, you learned about a syntactical element called a condition. Conditions were used as the selection criteria to further define a group of records. Conditions may be expressed in terms of value expressions and Boolean expressions. These are important elements in the larger structure, the record selection expression. To acquire the background to study the full record selection expression, first look at the value and Boolean expressions.

4.5.1 Value Expressions

The value expression (value-exp) provides an unambiguous representation of some combination of items and operators that is generally evaluated as a numeric value. Later in this chapter you are going to see how value expressions are used in Boolean expressions, in the print-list of the PRINT command, in the record number specifier of the SELECT command, and in the FIRST n clause of the full record selection expression.

As shown by Table 4-2, the range of items that can be used as value expressions is quite large. Table 4-2 summarizes these possibilities and explains them. You should be aware that it is possible to create compound value expressions that become quite complex. For example, (A + B)/ C * D + 500 is a value expression.

Table 4-2
Value Expressions

Item Format	Description	Remarks
field-name-1	Contents of this field in the CURRENT record[1] are evaluated and used.	Uses CURRENT record[1] of CURRENT collection, so both must exist and contain a field with this name.
collectn-name-1.field-name-2	Contents of this field in the CURRENT record[1] of the named collection are evaluated and used.	Be sure to supply a period (.) as a connector. The field named must exist in the records of the named collection. The collection must be established and have a CURRENT record[1].
"character-string"	The character string is used as given, unless the context forces it to be treated as a number[2].	Any characters may be used except the carriage return, line feed, or CTRL Z (^Z). To include a quotation mark, use two successive quotation marks.
integer-1	A string of digits interpreted as a decimal number.	
- value-exp-1	The resultant value is the negation of the value of value-exp-1.	Take care when using character-strings for value-exp-1: they are changed into numbers.
(value-exp-2)	Causes the evaluation of the enclosed value expression to occur prior to that of any other value expression with which it appears.	
value-exp-3 { + / - / * / / } value-exp-4	Interpreted as the arithmetic result of the operation on the two values expressed by the value expressions.	The rules of precedence for arithmetic operations hold (unless you use parentheses to force certain operations to occur first): multiplications (*) and divisions (/) precede additions (+) and subtractions (-), working left-to-right.

(continued on next page)

Table 4-2 (Cont.)
Value Expressions

Item Format	Description	Remarks
value-exp-5 ¦ value-exp-6	Creates a new character string by combining the two character strings given for value-exp-5 and value-exp-6.	Special caution should be used with numbers. The result maintains a left-to-right pattern. For example, "ABC"¦"DEF" produces "ABCDEF", "12"¦"034" produces "12034", but 12¦034 produces 1234.
{ MAX / MIN / AVERAGE / TOTAL } value-exp-7 [OF rse]	Computes[2] your choice of the maximum, minimum, average, or total, respectively of this value expression for all records (in the CURRENT collection, by default). If the OF clause is used, then the computation can occur on your choice of the named collection, the named domain, or the CURRENT collection, and they can be further restricted by the rest of the expression.	
{ *.prompt-name-1 / **.prompt-name-1 }	Permits the specification of the value at execution time. Has special importance in procedures. Prompt-name-1 should be the name you want DATATRIEVE-11 to use to prompt you to supply a value.	DATATRIEVE-11 prints the message PLEASE SUPPLY VALUE FOR prompt-name-1: and waits for your response before it processes the command in which this value expression occurs. (Examples occur in Sections 4.11 and 4.13.2.)
COUNT [OF rse]	Computes the number of records in the object of the record selection expression.	The default is the number of records in the CURRENT collection.

[1]The CURRENT record is the record the collection cursor identifies. The collection cursor can only be moved to a record by the SELECT command, which is discussed in Section 4.6.

[2]Value expressions are treated as numbers in the SELECT command, in the FIRST n clause of the record selection expression, and when subject to arithmetic operators (+, -, *, and /) or computations for maximum, minimum, total, and average. Thus, these are not appropriate places for the alphanumeric character string.

4.5.2 Boolean Expressions

Boolean expressions describe conditions used to restrict a collection to records that satisfy these criteria. The Boolean expressions may be simple (used alone) or compound (combined with other Boolean expressions). When they are incorporated in commands, DATATRIEVE always tests to see whether they are **true** for a given record. If the expressions are true, DATATRIEVE performs the command function on the record. If the expressions are not true, the record is rejected by the command.

Simple Boolean expressions compare the value of a particular field in a record against specified values. The comparison is based on certain "operators" that are shown in the formats summarized in Table 4-3. Note that some operators are synonymous; choose the one you like the best.

Compound Boolean expressions combine two or more Boolean expressions to create a more precise definition of the conditions to apply. The compound Boolean expressions are given in the second half of Table 4-3.

Table 4-3
Boolean Expressions

	Operator Symbol	Format/Example	Meaning
Simple	>	A > B	True if A is greater than B
	GT	A GT B	
	GREATER-THAN	A GREATER-THAN B	
	GE	A GE B	True if A is greater than or equal to B
	GREATER-EQUAL	A GREATER-EQUAL B	
	<	A < B	True if A is less than B
	LT	A LT B	
	LESS-THAN	A LESS-THAN B	
	LE	A LE B	True if A is less than or equal in value to B
	LESS-EQUAL	A LESS-EQUAL B	
	EQ	A EQ B	True if A equals B
	EQUAL	A EQUAL B	
	=	A = B	
		A EQ B,C, [OR] D A EQUAL B,C, [OR] D A = B,C, [OR] D	True if A equals any of the three values, B, C, or D
	NOT-EQUAL	A NOT-EQUAL B	True if A does not equal B
	NE	A NE B	
	BT	A BT B [AND] C	True if the value of A falls in between the range of values of B through C, inclusive
	BETWEEN	A BETWEEN B [AND] C	
Compound	AND	BOOL-A AND BOOL-B PRICE>10000 AND (RET) BUILDER EQ "SEAWORTHY" (RET)	True if the values of the Boolean expressions BOOL-A and BOOL-B are both true
	NOT	NOT BOOL-A NOT PRICE>10000 (RET)	True if the value of the Boolean expression BOOL-A is false
	OR	BOOL-A OR BOOL-B MODEL="43K" OR MODEL="49K" (RET)	True if either of the Boolean expressions BOOL-A or BOOL-B is true
	( )	(BOOL-A) (BEAM<12 AND LOA BT 20 AND 35) (RET)	True if BOOL-A is true. The parentheses force the evaluation of this Boolean first. Note that BOOL-A can be a compound Boolean, and this is a means of grouping Booleans in case the normal precedence rules* would not produce the desired effect.

Note
The character A above represents a field name, while B, C, and D are value expressions (as described in Section 4.5.1).

*Normal precedence rules require evaluation of expressions with NOT first, followed by expressions with AND, followed by expressions with OR.

4.5.3 Full Record Selection Expressions

The full record selection expression (rse) combines a number of elements already learned, including value expressions and Boolean expressions, as well as a sorting clause. It is a very powerful subexpression in a number of commands and as such will be described in much the same way the commands are.

rse

Purpose

This subexpression specifies a collection of records.

Format

```
[ALL | FIRST n] [collectn-name-2 IN] {CURRENT | collectn-name-3 | domain-name-1} [WITH conditn]

[SORTED BY key-1[,key-2...]]
```

1. You must supply either a domain-name, a collection name, or CURRENT and your choice must refer to a source domain that you have already readied.

2. If you specify FIRST n, n can be any value expression that yields an integer representing the upper bound of the number of records desired for processing.

3. If you specify ALL, all records in the named domain or collection will be processed. This is also the default.

4. If you specify collectn-name-2, you will create a new collection name. The only case where this collection name is retained, however, is when the rse occurs in a FIND rse command. When the rse occurs in a MODIFY or PRINT command, the new collection name is lost when the command completes. Be sure the name you assign does not conflict with a keyword (Table 3-2) or one of the existing names for a defined record, domain, or procedure (use SHOW ALL, if necessary, to check existing names).

5. You may optionally wish to restrict the records that are collected by specifying the word WITH followed by a condition (Boolean expression). Section 4.5.2 provides all the rules necessary to formulate this Boolean expression.

6. If you specify the SORTED clause, the sort occurs on either the CURRENT collection, the domain, or collectn-name-3.

7. The sort-keys assume the following format:

```
[ASC[ENDING] | DESC[ENDING] | INCREASING | DECREASING] field-name-1
```

8. The number of sort-keys you may specify is unlimited. If you exceed the system capacity, you will be notified that the dynamic memory is exhausted.

9. The specification of the word BY is optional, and is only offered to maintain an English-like structure.

10. Every field-name specified as a sort-key must be legal for the CURRENT collection. This means the field must occur in the record definition.

11. If the specifications of ASCENDING, INCREASING, DESCENDING, or DECREASING are omitted **for the first sort-key**, DATATRIEVE assumes you wish this field sorted in ASCENDING sequence.

12. If the sort direction specification is omitted **on subsequent sort-keys**, DATATRIEVE assumes you want the last-specified sorting-direction to prevail.

13. When more than one field-name is specified, the collection is sorted so that the first field-name is treated as the major sort key, and each successive key becomes more and more minor.

Requirements

If you specify CURRENT, there must be an established CURRENT collection. If you name a collection for collectn-name-3, that named collection must be established.

Prompts & Responses

No prompting messages appear, unless you introduced a value expression of the form

```
*.prompt-name-1
```

Effects

1. The collection of records is formed in the order in which the records are found. You should not infer that there will be any order to the collection unless you include the optional SORTED BY clause.

2. If you specified the FIRST n clause, you obtain a subset of the collection as the final act **after** any restricting and sorting have occurred.

3. If the number specified for n in the FIRST n clause exceeds the number of records that can be processed to meet the selection criteria, the maximum number will be chosen.

4. The collection cursor is left null.

5. If all the elements in the expression are specified, the sequence of events is as follows:

   a. Restrict the collection according to the WITH clause.

   b. Sort the collection per the SORTED BY specifications.

   c. Extract the FIRST n of the results.

   d. Name this collection according to collectn-name-2.

   If a given rse omits any of the clauses, that particular step disappears, but the progressive sequence still follows as above.

4.6 SELECTING A PARTICULAR RECORD IN A COLLECTION -- SELECT

Once you have established a collection, the ability to manipulate the collection cursor through the established collection becomes a prime consideration. The SELECT command provides this capability.

SELECT

Purpose

This command specifies that you want a particular record made available for your purposes from either the CURRENT collection or a named collection. You succeed in moving the collection cursor to a particular point in the collection. The record you move to becomes the CURRENT record, and the one immediately after it becomes the NEXT record. Remember, a record may be the CURRENT record without necessarily being in the collection that is the CURRENT collection.

Format

```
SELECT  [ FIRST     ]
        [ NEXT      ]
        [ LAST      ]  [collectn-name-1]
        [ value-exp-1 ]
```

1. You should supply a record number specifier such as FIRST or LAST. If you omit this specifier, it is assumed that you wish the NEXT record in the collection.

2. The specification for value-exp-1 must be a value expression that produces a positive integer, and it must not exceed the total number of records in the collection. This restriction prohibits the use of alphabetic character strings and necessitates caution with alphanumeric character strings.

3. You may optionally wish to use collectn-name-1 to name a specific collection for the selection process; otherwise, it is assumed you wish to reference the CURRENT collection.

Requirements

You must have already established either the named collection or a CURRENT collection.

Restriction

You should not use the SELECT command in any compound statements. Compound statements are created with the statements in Section 4.13.

Prompts & Responses

None.

Effects

The effects of the command are highly dependent on the record number specifier given and the state of the collection and the position of its cursor at the outset. Table 4-4 summarizes the different effects.

Notice particularly that any time you attempt a SELECT on an empty collection, you will get an error message.

Because there can be multiple collection cursors, SELECTing a record from a collection does not make that collection CURRENT if it was not already so.

Examples

The following example shows two ways to select the very next record from the CURRENT collection.

```
DTR>SELECT NEXT; (RET)                    (1)

DTR>SELECT; (RET)                         (2)
```

In Method (2), the record number specifier was omitted and NEXT is used by default.

The following example illustrates how to select just the first record from the collection of small boats (less than 26 feet long).

```
DTR>SELECT FIRST SMALL-BOATS; (RET)
```

This example illustrates how to select the twentieth yacht record in the collection of medium-sized yachts.

```
DTR>SELECT 20 MEDIUM-YACHTS; (RET)
```

This example selects the last yacht record in the CURRENT collection.

```
DTR>SELECT LAST; (RET)
```

Hints

It is important to keep track of which collection is the CURRENT collection in case you want to omit the collection name in the SELECT command or others. If you have any doubts about which collection is current, you can always issue the SHOW CURRENT command. (See Section 4.3.)

When the collection cursor is null, the use of a SELECT NEXT command is equivalent in effect to a SELECT FIRST.

Table 4-4
Effects of the SELECT Command

Specification	Normal Effect	Empty Collection	Cursor at LAST
FIRST	Cursor points to first member	Cursor is null (Error)	Gets first member with no error
NEXT	Cursor advances forward one record		Cursor is unchanged, error message is given
LAST	Cursor points to last member		Cursor is unchanged
value-exp-1	Cursor moves to member numbered value-exp-1		Gets the value-exp-1 member, if it exists, no error

4.7 PUTTING RECORDS INTO AN ORDER -- SORT

You will find frequent need to order or reorder collections of records. The sorting facility supplied by the SORT command alone or the SORTED BY clause in other commands gives you this power. Collections can be sorted into ascending (increasing) or descending (decreasing) sequence based on the standard ASCII collating sequence.

SORT

Purpose

This command generally changes the order of a collection of records. You may decide you wish the collection ordered according to certain key data fields and each of these fields can be in either ascending or descending order.

Format

SORT [collection-name-1] BY sort-key-1 [,sort-key-2...]

where each sort-key assumes the following form:

[**ASC** [ENDING] | **DESC** [ENDING] | **INCREASING** | **DECREASING**] field-name-1

1. If you omit a collection-name, it is assumed that you want the sort to occur on the CURRENT collection.

2. If you provide a collection-name, it must be the name of an established collection.

3. The number of sort-keys you may specify is unlimited. If you exceed the system capacity, you will be notified that the dynamic memory is exhausted.

4. The specification of the word BY is optional, and is only offered to maintain an English-like structure.

5. Every field-name specified must be legal for the collection being processed. This means it must occur in the record definition.

6. If the specifications of ASCENDING, INCREASING, DESCENDING, or DECREASING are omitted **for the first sort-key**, DATATRIEVE assumes you want this field sorted in ASCENDING sequence.

7. If the sort direction specification is omitted **on subsequent sort-keys**, DATATRIEVE assumes you want the last-specified sorting-direction to prevail.

8. When more than one field-name is specified, the collection is sorted so that the first field-name is treated as the major sort key, and each successive key becomes more and more minor.

Requirements

You must have already performed a successful FIND on this collection. Section 4.4 describes the FIND command.

Prompts & Responses

None.

Effects

1. The records in the collection are sorted into the order specified or implied.

2. The collection cursor is null at the conclusion of this command.

3. Any remaining pointers to records previously removed by an ERASE command (Section 4.10) are permanently erased.

Examples

The following example sorts all the yacht records in the 26-foot yacht collection into alphabetical order based on the manufacturer name first and then on the model number.

```
DTR>SORT YACHTS-26 BY ASC MANUFACTURER, MODEL; (RET)
```

The following example illustrates how to sort the records in the CURRENT collection into order according to the yachts having the highest price.

```
DTR>SORT BY DESCENDING PRICE; (RET)
```

Hints

If you encounter a problem in sorting, most likely it will be in exceeding the memory space. If this should happen, you may want to consider any or all of the following approaches:

1. FINISH any domains you can (Section 4.12.2).

2. RELEASE any collections you won't need (Section 4.12.1).

3. READY domains that were readied for WRITE or MODIFY access to permit read-only access since this mode uses less buffer space (Section 4.1).

4.8 LOOKING AT THE RECORD CONTENTS -- PRINT

By this time, the beginning user who used a terminal to practice the preceding commands is probably frustrated by being unable to "look at" the results of the commands entered. The PRINT command permits you to display certain parts of the **collections.**

PRINT

4.8.1 Listing Record Contents

Purpose

This command provides a printout of the record contents. You may request to see one or more fields in either a single record or a complete collection of records. Since these printouts can be formatted to a certain extent, they are quite sufficient for informal purposes. However, you may want to consider the capabilities of the Report Writer for more elaborately formatted printouts. See Chapter 5 for a discussion of the Report Writer.

Format

```
                                              { file-spec-1        }
PRINT [ALL] [print-list-1] [OF rse] [ON      { *.prompt-name-1    }]
```

where the record-selection-expression (rse) assumes the following form:

```
[ALL    ]                              { CURRENT           }
[FIRST n] [collectn-name-2 IN]        { collectn-name-3   } [WITH conditn]
                                       { domain-name-1     }

[SORTED BY key-1[,key-2...]]
```

If necessary, review the descriptions of record selection expressions in Section 4.5.3.

1. If you specify ALL before the print-list, you will obtain a printout of all the records in the group you designate.

2. If you omit ALL, the printout will include only the items designated for the CURRENT record of the designated group.

3. The optional print-list can specify names, value expressions, literals, and horizontal and vertical spacing characters. The print-list assumes the form of any combination of the optional elements shown in Table 4-5. Print-list elements are separated by commas. Note that "n" (as shown in the table) represents a decimal integer of your selection. For more details see Section 4.8.2.

4. The optional OF rse clause refers to the same record selection expression seen in Section 4.5.3. This powerful expression lets you print records from named collections rather than restricting you to the CURRENT collection and permits sorting them, refining them, and/or even subsetting them -- all in one command. If the OF rse clause is used, it completely overrides the specification of ALL.

5. If you specify the optional ON file-spec-1 clause, file-spec-1 must be the name of a file or device specification. If you omit the file specification,

the printout appears on your terminal device. Note your other alternative is to use the value expression *.prompt-name (described in Section 4.5.1) to prompt a later entry of a file or device specification.

6. Note that all the command elements after the command name PRINT are apparently optional. It is fine to specify only the word PRINT if the CURRENT collection has a CURRENT record; you will get the printout for that one record. However, if there is no CURRENT record in the CURRENT collection and if you omit all the optional command elements, you will obtain an error message and a printout of **every field in every record** in the CURRENT record collection.

   Again, be warned that all optional command elements, if specified, must be given in the sequence shown.

Requirements

The collection cursor must not be null or else you must request ALL.

Prompts & Responses

No prompting messages appear.

Effects

1. The printout will conform to the examples provided in Figure 4-1. Note that the default column headers consist of the field-names as specified in the record definition.

2. The output of the command is directed to the terminal device, unless you provide an alternate file specification with the ON file-spec-1 clause.

3. The values of the fields are printed in the appropriate columns.

4. The collection cursor is left unchanged.

Examples

More examples occur in Figure 4-2.

Hints

You may wish to use the CTRL O (^O) character combination to halt a lengthy printout that you realize is in error or should not have started at this time.

If a printout is moving over your video terminal screen faster than you can read it, you may want to use the CTRL S (^S) to cause a temporary interruption and the CTRL Q (^Q) to resume. However, a better choice may be to direct the printout to another file for hard copy review.

You may abbreviate the PRINT command in just two possible ways. You can eliminate the words ALL and OF when you have no print-list, want ALL records, and want to specify either a domain name or FIRST n. Thus, the following exceptions to the syntax above will be accepted:

```
DTR>PRINT FIRST 5 MEDIUM-YACHTS WITH PRICE > 5000; (RET)

DTR>PRINT YACHTS WITH PRICE<100000 SORTED BY DESC PRICE; (RET)
```

Table 4-5
Print-list Elements

Element	Function	Default
field-name-1 [modifiers]	Specifies a class of data items to be printed from the record. Modifiers are explained below.	All the fields in the record
value-exp-1 [modifiers]	A value to be inserted. (See Section 4.5.1.) Modifiers are explained below.	None
SPACE [n]	Causes n horizontal spaces between entries on each line.	1 space
TAB [n]	Introduces as many tab characters into the print line as specified by n.	If n is omitted, a single tab is assumed
COL n	Advances across the horizontal line to column n. Note: n must not be less than 1 or more than the page width. If n is less than present column, a line is skipped.	None
SKIP [n]	Moves n blank lines ahead to start a new line. Printing starts in column 1, unless another element modifies the position further.	Single spacing

(continued on next page)

Table 4-5 (Cont.)
Print-list Elements

Element	Function	Default
NEW-PAGE	Begins a new print page. Printing starts in column 1, unless another element modifies the position further.	None
Modifiers		
("header-1"[/"header-2"...])	Specifies a character-string (Section 3.8) to be printed on successive lines over the immediately preceding field-name or value-expression. Specification of a hyphen for the header indicates no header is desired.	Headers are composed of the field-name definitions from the record definitions
USING edit-string	Imposes the characteristics of this edit-string on the preceding field or value expression. Edit-strings must observe COBOL rules (Appendix E).	Uses the edit-string in the record definition, if any. Otherwise considers the field's PICTURE string.

NOTE

DATATRIEVE-11 restricts the column width to 132 characters. It automatically sets tabs at every eighth column. Thus, if the print position is column 10 and DATATRIEVE encounters a TAB 2 specification, the new position will be column 24.

4.8.2 Formatting the Printout

In the description of the PRINT command you learned that you could specify a print-list. Table 4-5 summarizes the possible elements you may choose for your print-list. These elements may be mixed in many combinations. All are optional. Sometimes DATATRIEVE assumes a default value when you omit an element in the print-list. Print-lists are processed in left-to-right order.

Figure 4-2 provides some examples of the PRINT command and the actual printouts that were produced from them.

```
DTR>!SET UP FOR A DEMONSTRATION OF VARIOUS FORMS OF THE PRINT COMMAND
DTR>READY YACHTS SHARED READ;
DTR>FIND YACHTS WITH RIG="SLOOP" AND PRICE>20000 SORTED BY LOA;
[8 RECORDS FOUND]
DTR>SELECT 4;
DTR>!
DTR>!*****************************************
DTR>!PRINT ALL THE FIELDS OF JUST THE 4TH RECORD OF THE CURRENT COLLECTION
DTR>!OF SLOOPS PRICED OVER $20,000
DTR>!(IT'S THE SELECTED RECORD)
DTR>!
DTR>PRINT
                                LENGTH
                                 OVER
MANUFACTURER    MODEL      RIG    ALL    WEIGHT BEAM  PRICE

 BAYFIELD     30/32       SLOOP   32      9,500  10  $32,875

DTR>!*****************************************
DTR>!THE NEXT 3 EXAMPLES JUST PRINT THE VALUE EXPRESSIONS IN THEIR PRINT-LISTS
DTR>!
DTR>PRINT "DATATRIEVE-11 MAKES"," REPORT WRITING ","EASY"
DATATRIEVE-11 MAKES  REPORT WRITING  EASY

DTR>PRINT "2 + 2 = 5"
2 + 2 = 5

DTR>PRINT 2+2
  4

DTR>!*****************************************
DTR>!PRINT THE MANUFACTURER AND MODEL FIELDS OF ALL THE RECORDS IN THE CURRENT
DTR>!COLLECTION
DTR>!
DTR>PRINT ALL COL 10,MANUFACTURER,COL 30,MODEL;

          MANUFACTURER           MODEL

           CAPE DORY             28
           SABRE                 28
           RYDER                 S. CROSS
           BAYFIELD              30/32
           CHALLENGER            32
           CARIBBEAN             35
           CHALLENGER            35
           CHRIS-CRAF            CARIBBEAN
```

Figure 4-2 Sample PRINT Command Output

```
DTR>!****************************************
DTR>!PRINT ALL FIELDS OF ALL KETCHES PRICED OVER $30,000,
DTR>!SORTED BY LENGTH
DTR>!
DTR>PRINT YACHTS WITH RIG = "KETCH" AND PRICE > 30000 SORTED BY LOA

                                LENGTH
                                 OVER
MANUFACTURER    MODEL      RIG   ALL   WEIGHT BEAM  PRICE

 I. TRADER   37          KETCH   36    18,600  12  $39,500
 ALBERG      37 MK II    KETCH   37    20,000  12  $36,951
 IRWIN       37 MARK II  KETCH   37    20,000  11  $36,950

DTR>!****************************************
DTR>!PRINT ALL FIELDS OF ALL RECORDS IN THE DOMAIN OF YACHTS
DTR>!
DTR>PRINT YACHTS

                                LENGTH
                                 OVER
MANUFACTURER    MODEL      RIG   ALL   WEIGHT BEAM  PRICE

 ALBERG      37 MK II    KETCH   37    20,000  12  $36,951
 ALBIN       79          SLOOP   26     4,200  10  $17,900
 ALBIN       VEGA        SLOOP   27     5,070  08  $18,600
 AMERICAN    26          SLOOP   26     4,000  08   $9,895
 AMERICAN    26-MS       M/S     26     5,500  08  $18,895
 BAYFIELD    30/32       SLOOP   32     9,500  10  $32,875
 CAPE DORY   28          SLOOP   28     9,000  09  $21,990
 CAPE DORY   TYPHOON     SLOOP   19     1,900  06   $4,295
 CAPE DORY   25          SLOOP   25     4,000  07   $8,995
 CARIBBEAN   35          SLOOP   35    18,000  11  $37,850
 CHALLENGER  32          SLOOP   32    12,800  11  $31,835
 CHALLENGER  35          SLOOP   35    14,800  12  $39,215
 CHRIS-CRAF  CARIBBEAN   SLOOP   35    18,000  11  $37,850
 CLIPPER     CM 30       SLOOP   30     3,800  08   $9,500
 CLIPPER     CM 32       SLOOP   32     4,500  08  $12,950
 EASTWARD    HO          M/S     24     7,000  09  $15,900
 GRAMPIAN    26          SLOOP   26     5,600  08  $11,495
 GRAMPIAN    28          SLOOP   28     6,900  10  $14,475
 GRAMPIAN    34          KETCH   33    12,000  10  $29,675
 HUNTER      27          SLOOP   27     6,500  09  $14,999
 I. TRADER   37          KETCH   36    18,600  12  $39,500
 IRWIN       37 MARK II  KETCH   37    20,000  11  $36,950
 IRWIN       25          SLOOP   25     5,400  12  $10,950
 ISLANDER    28          SLOOP   28     5,994  10  $15,908
 ISLANDER    BAHAMA      SLOOP   24     4,200  08   $6,500
 LINDSEY     39          M/S     39    14,500  12  $35,900
 RYDER       S. CROSS    SLOOP   31    13,600  00  $32,500
 SABRE       28          SLOOP   28     7,400  09  $22,000
 SALT        19          SLOOP   25     2,600  07   $6,590
 TANZER      26          SLOOP   26     4,350  09  $11,750
 TANZER      28          SLOOP   28     6,800  10  $17,500
 VENTURE     21          SLOOP   21     1,500  07   $2,823
 VENTURE     222         SLOOP   22     2,000  07   $3,564
 WESTERLY    CENTAUR     SLOOP   26     6,700  08  $15,245
```

Figure 4-2 (Cont.) Sample PRINT Command Output

```
DTR>!************************************************
DTR>!PRINT ALL THE FIELDS OF THE FIRST 3 RECORDS IN THE GROUP OF YACHTS THAT ARE
DTR>!SHORTER THAN 26 FT.
DTR>!
DTR>PRINT FIRST 3 SMALL-YACHTS IN YACHTS WITH LOA < 26

                                LENGTH
                                 OVER
MANUFACTURER    MODEL      RIG    ALL    WEIGHT BEAM  PRICE

 CAPE DORY    TYPHOON     SLOOP   19      1,900  06   $4,295
 CAPE DORY    25          SLOOP   25      4,000  07   $8,995
 EASTWARD     HO          M/S     24      7,000  09  $15,900

DTR>!************************************************
DTR>!PRINT JUST THE PRICE FOR THE SAME GROUP BUT EDIT IT DIFFERENTLY
DTR>!
DTR>PRINT PRICE USING $ZZ,ZZZ.00 OF FIRST 3 SMALL-YACHTS IN YACHTS WITH LOA < 26;

  PRICE

$ 4,295.00
$ 8,995.00
$15,900.00

DTR>!************************************************
DTR>!PRINT THE CURRENTLY SELECTED RECORD IN THE CURRENT COLLECTION
DTR>!
DTR>PRINT

                                LENGTH
                                 OVER
MANUFACTURER    MODEL      RIG    ALL    WEIGHT BEAM  PRICE

 BAYFIELD     30/32       SLOOP   32      9,500  10  $32,875

DTR>!WERE YOU SURPRISED? NONE OF THE PRINT COMMANDS HAVE REALLY ESTABLISHED
DTR>!COLLECTIONS OR SELECTED ANY RECORDS, SO THE CURRENT RECORD IS STILL THE 4TH
DTR>!RECORD IN THE COLLECTION OF SLOOPS FORMED BY THE VERY FIRST 3 COMMANDS
DTR>!************************************************
DTR>!PRINT THE PRICE, MANUFACTURER AND MODEL OF YACHTS COSTING OVER
DTR>!$35,000.  DOUBLE SPACE THE PRINTOUT.  ^START IN COLUMN 16, GIVING
DTR>!THE PRICE FIELD THE HEADING 'IT'S ONLY MONEY'.  PRICE WILL BE
DTR>!ZERO SUPPRESSED, BUT NO DOLLAR SIGN IS DESIRED.
DTR>!
DTR>PRINT SKIP 2,TAB 2,PRICE('IT'S ONLY'/'MONEY') USING -
DTR>ZZ,ZZZ, COL 34,MANUFACTURER,MODEL OF YACHTS WITH -
DTR>PRICE>35000;

               IT'S ONLY
                 MONEY          MANUFACTURER   MODEL


                36,951           ALBERG       37 MK II


                37,850           CARIBBEAN    35


                39,215           CHALLENGER   35
```

Figure 4-2 (Cont.) Sample PRINT Command Output

4.9 CHANGING RECORD CONTENTS IN THE FILES -- MODIFY

One of the most powerful capabilities of DATATRIEVE is the ability to change the data as it exists in the RMS-11 files. With all the commands previously described, all actions occurred on copies of portions of the files. At no time were you able to affect the file itself. However, with the MODIFY command, you will be able to change data values in specific records. Certain safeguards are, however, inherent in the system. For example, you will only be able to use the MODIFY command on domains you successfully readied in WRITE or MODIFY mode. (See Section 4.1.)

MODIFY

Purpose

This command can change from one to all the data fields in:

- any given record or
- all the records in a collection.

Format

MODIFY [**ALL**] [fld-name-1 [,fld-name-2...]] [**OF** rse]

where the record selection expression (rse) assumes the form:

[**ALL** / **FIRST** n] [collectn-name-2 **IN**] {CURRENT / collectn-name-3 / domain-name-1} [**WITH** conditn]

[**SORTED** BY key-1[,key-2...]]

If necessary, review the descriptions of record selection expressions (Section 4.5.3) and sort-keys (Section 4.7).

1. If you specify ALL before fld-name-1, then the changes you request will be made to **every record** in the group you designate. If you omit ALL before fld-name-1, then your changes will be made only to the CURRENT record in either the named or CURRENT collection.

2. If you supply a field name, it must name a field that belongs to the record type associated with either the CURRENT or named collection.

3. If you omit the field names, DATATRIEVE will prompt you to specify a change value for **every field** in the record.

Requirements

You must have already performed a successful READY on the source domain for the records, and it must be readied for WRITE or MODIFY access. Section 4.1 describes the READY command.

Unless you are specifying ALL, the collection cursor must not be null; it must point to a record as the result of a successful SELECT command. Section 4.6 describes the SELECT command.

Restriction

You cannot modify the primary key field in any indexed sequential record.

Prompts & Responses

Once your command has been accepted, DATATRIEVE prompts you with a request to specify the new value for each of the field names. If all field names are being changed, then they will be requested in the order of the record definition.

DATATRIEVE prompts with the message

PLEASE SUPPLY VALUE FOR field-name:

It then waits for you to enter a value after the colon. Note that DATATRIEVE prompts for each elementary data item in the field names you supplied, or issues all the field names one by one.

If at any time in the prompting cycle, you want to escape, you can enter a CTRL Z (^Z). This aborts the MODIFY command, and prints the message

EXECUTION TERMINATED BY OPERATOR

Effects

1. In **both** the collection **and** the RMS-11 file each field name specified is changed to the new value supplied.

2. The collection cursor is left unchanged.

3. Additional effects may occur as prescribed by the record selection expression. (See Section 4.5).

Example

The following example changes two fields in the four records in the collection of yachts built by Seaworthy.

```
DTR>MODIFY ALL DISP, PRICE OF SEAWORTHY-YACHTS; RET

PLEASE SUPPLY VALUE FOR DISP:  5500 RET

PLEASE SUPPLY VALUE FOR PRICE:  22500 RET
```

Hints

The values supplied should correspond to the data characteristics established for the field in the record definition. For example, the field length of the new value should not exceed the defined length. Explicit checking is not performed, but certain violations will be recognized if DATATRIEVE uncovers an incompatibility while moving the new data into the area reserved for the old data.

You will not be permitted to skip over a data item by responding with just a carriage return (RET). If you do, you will be prompted again to supply the same field. However, you can leave a data item empty or zero-valued by entering a single space before the carriage return.

4.10 REMOVING ONE OR MORE RECORDS -- ERASE

During routine operations, it is not uncommon to determine that one or more records are invalid or have outlived their usefulness and should be removed from a file. This sort of file maintenance function is generally performed by a program at the installation. However, DATATRIEVE-11 offers the ability to perform "simple" removals of certain records. However, the word "simple" is critical here. When a record is removed by DATATRIEVE, there is no audit trail left. Because no historical details from the record are retained, serious accounting problems can result if ERASE is used indiscriminately. This is one reason why the use of passwords and access modes is so important. However, abuses can still occur at the hands of qualified users. Hopefully, when you use the ERASE command, you will do so with the proper amount of caution and respect for the data integrity.

ERASE

Purpose

This command removes one or more records from the files.

Format

ERASE [**ALL** [**OF** rse]]

where the record-selection-expression (rse) assumes the following form:

[**ALL** / **FIRST** n] [collectn-name-2 **IN**] { CURRENT / collectn-name-3 / domain-name-1 } [**WITH** conditn]

[**SORTED** BY key-1 [,key-2...]]

If necessary, review the descriptions of record selection expressions (Section 4.5.3) and sort-keys (Section 4.7).

1. If you supply a record selection expression, it must refer to a collection in a source domain that you have already readied in WRITE access mode.

Requirements

You must have already peformed a successful READY on the source domain for WRITE access. Section 4.1 describes the READY command.

Restriction

You cannot erase sequential RMS-11 records.

Prompts & Responses

None.

Effects

1. If you omit the optional elements, you will erase only the CURRENT record of the CURRENT collection, if any.

2. If you specify only ALL, you will erase every member of the CURRENT collection.

3. If you specify the optional rse, you will erase every record of the collection you identify.

4. The collection cursor becomes null.

Examples

The following example illustrates how you could erase all records in the CURRENT collection.

```
DTR>ERASE ALL; (RET)
```

The next example illustrates how you could erase just the CURRENT record in the CURRENT collection.

```
DTR>ERASE; (RET)
```

The last example illustrates how to erase all the records in the named collection SMALL-BOATS.

```
DTR>ERASE ALL OF SMALL-BOATS; (RET)
```

Hints

You must know which is the CURRENT collection and which is the CURRENT record before you attempt any ERASE command. If in doubt, use the SHOW CURRENT command (Section 4.3) and/or the PRINT command (Section 4.8) to ensure that you know what you will be erasing.

If you request a printout that includes erased records, DATATRIEVE omits the deleted records. However, if you SELECT a record that has been erased, DATATRIEVE notifies you that the record has been removed. This notification is possible since a record pointer remains after the deletion occurs.

It is only through the SORT command or SORTED BY clause (Sections 4.7 and 4.4, respectively) that pointers to records previously erased are removed.

4.11 ADDING A NEW RECORD TO THE FILE -- STORE

Certain qualified users may be permitted to add new records to existing files. This is a common file maintenance function in any installation, but there are some advantages (with attendant disadvantages) to using DATATRIEVE for this function. The DATATRIEVE user can rapidly add a record and supply data for the fields, avoiding the need to use a special file maintenance program. However, opening records with DATATRIEVE includes the risk that correct procedures may not be followed. Furthermore, data will not be subjected to extensive edits and logical crosschecks such as a specialized program could impose. For example, if it is known that sailboats can only have lengths between 8 and 70 feet, you would logically assume an error has been made if someone specifies 2 or 99 feet for the LOA (length-over-all field). This is the sort of checking a specialized program could do that the more casual DATATRIEVE user could omit.

Likewise, suppose management has recently dictated that no more boats manufactured by SMITT-JONES should be accepted. Now, even if the maintenance program has been updated to include a test for this prohibition, the unwitting DATATRIEVE user could insert new records and undermine the policy decision.

Using DATATRIEVE to add records to files is the quick and dirty method, in every sense of the expression. Please be aware of the potential problems that this feature can cause.

STORE

Purpose

This command creates a new record in the domain you specify. DATATRIEVE accepts your specification of the fields you want to enter or prompts you to specify data for each field defined for the record.

Format

STORE domain-name-1 [USING statement-1] [VERIFY USING statement-2]

1. You must supply a domain-name, and it must name a domain you have already readied for WRITE or EXTEND access (Section 4.1).

2. If you want to provide a statement for statement-1, in general it will assume the following form:

```
BEGIN fld-name-1 = value-1
[;fld-name-2 = value-2...]
;END
```

   where its keyword USING, if specified, must be on the same line as the keyword STORE; the remaining elements, separated by semicolons, may be on one or more successive lines. Follow the usual conventions regarding hyphens and carriage returns for line continuations. You may want to read about the BEGIN-END Block and assignment statements that are described in Section 4.13.

3. If you specify a VERIFY clause, no data is stored until the conditions of statement-2 are completed. Thus, this provides an opportunity to ensure that data is correct before it becomes part of the file.

4. The optional word USING is permitted for English-like usage, but serves no other purpose.

Requirements

You must have already performed a successful READY on this domain for either WRITE or EXTEND access. Section 4.1 describes the READY command.

Restriction

You cannot STORE new records into RMS-11 relative files.

Prompts & Responses

Unless you specify a statement giving field-names and desired values, you will be prompted to supply data for **each** of the defined fields in the record. The prompting message assumes the following form:

PLEASE SUPPLY VALUE FOR field-name:

where DATATRIEVE replaces "field-name" in the message

with the name of the field being sought. DATATRIEVE waits for your response, and if accepted, proceeds to the next sequential field in the record definition, until all fields have been filled.

Effects

1. The fields are added to the record in the order in which they are provided. If any piece of data is inappropriate for its definiton, assumptions are made or error messages issued, and the command continues on to the next defined field. If a numeric field was oversize and contained numeric data, the high-order digits were truncated and an error message issued. If an alphanumeric field contained excessive string data, truncation occurred without an error message.

2. The collection cursor remains unchanged. The newly added record is not in any collection. If you want to print this record, you must first establish a collection that contains it, and then select it.

Examples

The following example stores a new yacht record in the domain YACHTS. Observe that this choice of format causes DATATRIEVE to prompt you for the data items.

```
DTR> STORE YACHTS; RET

PLEASE SUPPLY VALUE FOR MANUFACTURER:  DONOVAN RET
PLEASE SUPPLY VALUE FOR MODEL:  34LK RET
PLEASE SUPPLY VALUE FOR RIG:  SLOOP RET
PLEASE SUPPLY VALUE FOR LENGTH-OVER-ALL:  34 RET
PLEASE SUPPLY VALUE FOR DISPLACEMENT:  4000 RET
PLEASE SUPPLY VALUE FOR BEAM:  12 RET
PLEASE SUPPLY VALUE FOR PRICE:  32000 RET
```

If you had merely wanted to open a new record with only the manufacturer, model, and type of rig for the data items, you could have done the following:

```
DTR> STORE YACHTS BEGIN RET
DTR> MANUFACTURER="DONOVAN";  MODEL="34LK"; RET
DTR> RIG="SLOOP";  END; RET
```

The next example first obtains data for a new yacht record, and then performs a series of IF statements (Section 4.13.5) to test for undesirable data values. If any are found, the STORE function does not occur. If the record passes these tests, the record is printed and the user is given the opportunity to review and approve it in its entirety prior to storing it.

```
DTR>STORE YACHTS VERIFY USING (RET)
DTR>  BEGIN (RET)
DTR>    IF BEAM EQ 0 THEN ABORT "BAD BEAM" (RET)
DTR>    IF DISP EQ 0 THEN ABORT "BAD DISPLACEMENT" (RET)
DTR>    IF NOT LOA BETWEEN 20 AND 60 THEN ABORT (RET)
DTR>       "BAD LENGTH" (RET)
DTR>    PRINT (RET)
DTR>    DISPLAY "CONFIRM WITH Y IF OK" (RET)
DTR>    IF *.CONFIRMATION NE "Y" THEN (RET)
DTR>    ABORT "STORE ABORTED" (RET)
DTR>END (RET)
```

In this case, because the PRINT command occurs within the VERIFY operation, the record of interest (which is not the CURRENT one), is the one displayed.

Hints

If you do not use the VERIFY option, make a practice of printing out each new record upon completion of a successful STORE command. This way, you may detect incorrect or omitted data items. Corrections can be made by using the MODIFY command (Section 4.9), once you have selected (Section 4.6) the new record.

As in the MODIFY command, explicit checking is not performed, but certain violations will be recognized if DATATRIEVE uncovers an incompatibility while moving new data into its reserved area.

When in full prompt mode, you will not be permitted to skip over a data item by responding with just a carriage return ((RET)). If you do, you will be prompted again to supply the same field. However, you can leave a data item empty or zero-valued by entering a single space before the carriage return.

A final hint is to consider the use of the REPEAT statement (Section 4.13.3) in conjunction with the STORE command whenever you must create a series of records in a given session. For example, observe how the following combination of the two commands saves operator input time:

```
DTR>REPEAT 3 STORE YACHTS BEGIN (RET)
DTR>MANUFACTURER="DONOVAN"; (RET)
DTR>MODEL=*.MODELNAME; (RET)
DTR>RIG="SLOOP"; (RET)
DTR>END; (RET)
PLEASE SUPPLY VALUE FOR MODELNAME:  35LK (RET)
PLEASE SUPPLY VALUE FOR MODELNAME:  36LK (RET)
PLEASE SUPPLY VALUE FOR MODELNAME:  37LK (RET)
DTR>
```

In the example above, you created three new records, having three fields with nonzero or nonblank values. Two of the fields were common to all three records; therefore, with the use of the REPEAT statement and the value expression of the form *.prompt-name, you created a situation where the only data input was for the field that varied from record to record. Had ten or more records been involved, the savings would seem even more significant.

4.12 SOME SPACE CONSIDERATIONS

Since DATATRIEVE keeps control information and record buffers for active collections in memory, it is important to keep the memory overhead as low as possible. Two commands facilitate this: RELEASE and FINISH. Invoke RELEASE whenever you are sure you have finished using a named collection. FINISH serves a similar function for domains.

RELEASE

4.12.1 When You're Done With a Collection -- RELEASE

Purpose

This command effectively removes a named collection from the system to free space for other uses. Note that it is the collection kept in temporary storage that is lost, but the records and domains from which it originated are unaffected. If necessary, the released collection can always be reestablished later.

Format

RELEASE collectn-name-1 **[**,collectn-name-2...**]**

1. You must supply at least one legitimate collection name. CURRENT is a legitimate collection name for an unnamed collection, if it is the CURRENT one.

2. You may optionally wish to name additional collections for release. The collections need not be formed from the same source domain.

Requirements

There should be one or more named collections in your domains.

Prompts & Responses

None.

Effects

1. The collections are removed in the order in which they are named in the command.

2. If the command fails while looking for subsequent collections, you will find that the collections up to the problem collection have been lost. However, the problem collection and any following collections have not been touched.

Examples

The following example releases the SMALL-BOATS collection.

```
DTR>RELEASE SMALL-BOATS; (RET)
```

The following example illustrates how to release three collections:

```
DTR>RELEASE SMALL-BOATS,MEDIUM-YACHTS,LARGE-YACHTS; (RET)
```

Hints

Cultivate the habit of releasing collections as soon as you know you have finished with them.

Note that an unnamed CURRENT collection is always automatically released by the next successful FIND command.

The FINISH command (Section 4.12.2) RELEASEs all collections belonging to the domains it operates on. Thus, it is redundant to specify a RELEASE command followed by a FINISH command involving the same domain.

Likewise, if you intend to EXIT immediately after this RELEASE command, it is unnecessary to use the RELEASE command. EXIT performs the RELEASE function for you.

FINISH

4.12.2 When You're Done With a Domain -- FINISH

Purpose

This command deactivates one or more domains to free space and/or to permit other users to gain access. You do not remove the domain or its records, you merely relinquish any previous claims to it made through an earlier READY command in this session.

Format

FINISH [domain-name-1 [,domain-name-2...]]

1. If you do not specify any domain names, all your READY domains are deactivated.

Requirements

None.

Prompts & Responses

None.

Effects

1. The named domains that were READY are deactivated so that they are no longer READY and cannot be the subject of any FIND commands.
2. If you name a domain that never was READY, you receive an error notification.
3. Any collections that existed for the domain are RELEASEd.

Examples

The following example finishes the domain YACHTS.

```
DTR>FINISH YACHTS; (RET)
```

The following example illustrates how to finish all ready domains.

```
DTR>FINISH; (RET)
```

Hints

If you accidentally FINISH a domain you want READY, you can simply READY it again.

If you intend to EXIT immediately after this FINISH command, it is unnecessary to use the FINISH command. EXIT performs a FINISH for you.

4.13 COMBINING DATATRIEVE-11 COMMANDS

There are a few additional statements you can learn that will permit you to combine DATATRIEVE-11 commands in powerful ways. These capabilities may have more significance to the designers of procedures (Chapter 6), but are available to all DATATRIEVE-11 users. They are therefore brought to your attention.

These statements employ some elements you have already learned: the record selection expression and the simple forms of the FIND, SORT, MODIFY and PRINT commands. The statements will be explained first and then some examples of how to combine them effectively will be given. You have actually already seen three of these, the BEGIN-END Block, the assignment statement, and the REPEAT statement, all in explanations of the STORE command; but they will be introduced more formally now.

BEGIN-END Block

4.13.1 BEGIN-END Block

Purpose

This statement groups the enclosed statements into a single compound statement. This facility is employed in the STORE command (Section 4.11).

Format

BEGIN statement-1 [; statement-2...] **END**

1. You must supply at least one assignment statement (Section 4.13.2).

Requirements

Conform to the requirements of the STORE and assignment statements. Section 4.11 describes the STORE command. Section 4.13.2 describes the assignment statement.

Prompts & Responses

None.

Effect

1. The statements are executed in the sequence entered.

Examples

The following example shows the use of a BEGIN-END Block with the STORE command. The effect will be to store a new record in the domain YACHTS for a 24-foot yacht built by Dufour with a model number of 24.

```
DTR>STORE YACHTS USING (RET)
DTR>BEGIN (RET)
DTR>    BUILDER="DUFOUR" (RET)
DTR>    MODEL="24" (RET)
DTR>    LOA=24; (RET)
DTR>END; (RET)
```

Assignment Statement

4.13.2 Assignment Statement

Purpose

The assignment statement provides an alternative method of establishing field values in the STORE command. The value is supplied by the statement itself, either explicitly or by reference to another field name. This differs from the technique of the MODIFY command (Section 4.9) since in that case you merely identified the changing field(s) by name, and then DATATRIEVE requested the new data item-by-item.

Formats

field-name-1 = value-exp-1

1. Field-name-1 must be defined in the record definitions for the source domain.
2. Value-exp-1 must be a legal value expression.

field-name-2 = field-name-3

1. Field-name-2 and field-name-3 must each be group data items.
2. These groups must have at least one identically named elementary data item. This name may be the defined name or the substitutable QUERY-NAME.

Requirements

The record must exist in a source domain that you have readied for WRITE or EXTEND access.

Prompts & Responses

None, unless value-exp-1 assumes the form of *.promptname or **.promptname.

Effects

1. The fields are added to the record in the order in which they are provided. If any piece of data is inappropriate for its definition, assumptions are made or error messages issued, and the command continues on to the next defined field. If a numeric field was oversize and contained numeric data, the high-order digits were truncated, and an error message issued. If an alphanumeric field contained excessive string data, truncation occurred without an error message.
2. The collection cursor is left unchanged.

Example

The following example uses three assignment statements in a BEGIN-END Block within a STORE command to place a value of $9000 in the price field in the new record.

```
DTR>STORE YACHTS BEGIN (RET)
DTR>    BUILDER="SEAWORTHY" (RET)
DTR>    MODEL="SW I" (RET)
DTR>    PRICE = 9000; END; (RET)
```

The second format of the assignment statement is used primarily to restructure data. This topic is described separately for the Data Administrator in Section 7.3.

Hints

Note that you must use the equals sign symbol (=) not the Boolean operator EQ.

No editing is performed on the data you supply.

The final example in the Hints section for the STORE command (Section 4.11) illustrated the use of *.prompt-name as the value expression in the assignment statement. Consider now, the effect of **.prompt-name in a similar situation. The value expression using the double asterisks has special meaning in a situation where repetition is planned. The value expression **.prompt-name effectively asks for this value **just one time** when inside a REPEAT or FOR loop. Once the value is supplied, it is used for its purpose repetitively and no further requests are made of the user for this item. For example, although the model number is requested three times in the example below, the beam is only requested once.

```
DTR>REPEAT 3 STORE YACHTS BEGIN (RET)
DTR>MANUFACTURER="DONOVAN"; (RET)
DTR>MODEL=*.MODELNAME; (RET)
DTR>RIG="SLOOP"; (RET)
DTR>BEAM=**.BEAMSIZE; (RET)
DTR>END; (RET)
PLEASE SUPPLY VALUE FOR MODELNAME: 35LK (RET)
PLEASE SUPPLY VALUE FOR BEAMSIZE: 12 (RET)
PLEASE SUPPLY VALUE FOR MODELNAME: 36LK (RET)
PLEASE SUPPLY VALUE FOR MODELNAME: 37LK (RET)
DTR>
```

REPEAT

4.13.3 REPEAT Statement

Purpose

This statement causes the immediately following statement to be executed a given number of times before execution progresses on to the next command. This clause can save you a great deal of tiresome repetition.

Format

REPEAT value-exp-1 statement-1

1. You must supply a value expression for the number of times to repeat the action (Section 4.5.1).

Requirements

None.

Prompts & Responses

None, unless value-exp-1 assumes the form of *.promptname or **.promptname.

Effect

1. The subsequent statement is executed as many times as dictated by the value supplied.

Example

The following example adds 20 new yacht records to the domain.

```
DTR>REPEAT 20 STORE YACHTS (RET)
```

Hints

Remember that with the combination of REPEAT and STORE you can propagate a simple error through a great deal of data.

4.13.4 THEN Statement

Purpose

This connecting word joins commands or statements so that execution of the second statement immediately follows the first.

Format

statement-1 **THEN** statement-2

1. You must supply two statements. You may choose any of the commands, used singly or in combinations for these statements, with two exceptions. Do not use the FIND command with SELECT, and do not use ABORT (Section 4.13.6) for statement-1.

Requirements

Conform to the requirements of the statements.

Restrictions

Do not use the combination FIND THEN SELECT. Do not specify an ABORT statement for statement-2.

Prompts & Responses

As provided by the statements being executed.

Effects

1. The first statement executes, followed by the second one.
2. Other effects depend on the choice of statements.

Example

The following example directs DATATRIEVE-11 to print the type and price of the CURRENT yacht and then modify the price.

```
DTR>PRINT TYPE, PRICE THEN MODIFY PRICE; (RET)

MANUFACTURER    MODEL    PRICE

AMERICAN          26     $9,898
PLEASE SUPPLY VALUE FOR PRICE:  10,898 (RET)
```

Hint

You may issue a number of successive THEN statements if meaningful. For example, you could enter the following commands:

```
DTR>READY YACHTS EXTEND (RET)
DTR>STORE YACHTS USING (RET)
DTR>     BUILDER = "DONOVAN" THEN (RET)
DTR>     MODEL = "DV24"; (RET)
```

4.13.5 IF-THEN-ELSE Statement

Purpose

This statement provides a limited choice of actions that may be followed based on existing conditions.

Format

IF condition THEN statement-1 [ELSE statement-2]

1. You must supply a condition in the form of a Boolean expression (Section 4.5.2).

2. You must supply an action to be taken, in the form of statement-1. Statement-1 may be a simple or compound command.

3. You may optionally want to specify a second possible action in the form of statement-2. Statement-2 may be a simple or compound command.

Requirements

Conform to the requirements of the statement(s) specified.

Prompts & Responses

As dictated by the statement(s) specified.

Effects

1. The condition is tested and if and only if it is true, statement-1 is executed. If the condition proves false, any optional statements following the word THEN will be executed.

2. The execution proceeds to the next DATATRIEVE-11 command.

Example

The following example tests if LOA is greater than 99. If it is, the user wishes to MODIFY the field. Otherwise, the execution can proceed to the next command.

```
DTR>IF LOA>99 THEN MODIFY LOA; RET
```

Hint

You may nest your IF statements. For example, the following sequence would classify a yacht by size and recognize any yacht that was oversize. It employs the DISPLAY statement described in Section 4.13.7.

```
DTR>IF LOA BETWEEN 0 AND 26 THEN DISPLAY "SMALL-YACHT" ELSE (RET)
DTR>IF LOA BETWEEN 27 AND 32 THEN DISPLAY "MEDIUM-YACHT" ELSE (RET)
DTR>IF LOA BETWEEN 33 AND 99 THEN DISPLAY "LARGE-YACHT" ELSE (RET)
DTR>DISPLAY "THIS YACHT IS WRONG LENGTH" (RET)
```

ABORT

4.13.6 ABORT Statement

Purpose

The ABORT statement provides a means of terminating command execution in a compound statement. Note that it only affects the one command or statement under consideration; it does not affect the DATATRIEVE-11 session as a whole.

Format

ABORT value-exp-1

1. Value-exp-1 must be a value expression. (See Section 4.5.1.)

Requirements

None.

Prompts & Responses

None.

Effects

1. If the ABORT action takes place, whatever value you specified for value-exp-1 will be displayed on your terminal device, followed by the notification

   EXECUTION TERMINATED BY "ABORT" COMMAND

2. The collection cursor is left unchanged.

3. You are prompted to enter the next DATATRIEVE command.

Example

The following example shows the use of the ABORT statement in combination with an IF-THEN-ELSE statement, for a record whose model number is not 345T.

```
DTR>IF MODEL EQ "345T" THEN MODIFY PRICE ELSE (RET)
DTR>ABORT "NOT MODEL 345T"; (RET)
ABORT: NOT MODEL 345T
EXECUTION TERMINATED BY "ABORT" COMMAND
DTR>
```

Hint

In the example below the concatenation character (|) is used to construct a value-expression that provides a more informative message.

```
DTR>IF MODEL EQ "345T" THEN MODIFY PRICE ELSE (RET)
DTR>ABORT "MODEL NO. IS: "|MODEL (RET)
```

DISPLAY

4.13.7 DISPLAY Statement

Purpose

This statement causes DATATRIEVE to immediately print the value specified. This facility lets you conduct brief checks on certain results while some operation is in progress. It is clearly not intended for writing large amounts of data; the PRINT command or Report Writer facilities are better choices for that. However, the DISPLAY statement can be a handy tool for displaying limited quantities of data, especially when you recall the range of items qualifying as value expressions (Table 4-2).

Format

DISPLAY value-exp-1

1. You must supply a value expression. Refer to Section 4.5.1.

Requirements

None.

Prompts & Responses

None, unless value-exp-1 assumes the form of *.promptname or **.promptname.

Effect

1. The value of the value expression is displayed on your terminal.

Example

The following example shows the DISPLAY statement used with an IF-THEN-ELSE statement.

```
DTR>IF PRICE>35000 THEN DISPLAY (RET)
DTR>"OVER-BUDGET" ELSE PRINT BUILDER; (RET)

MANUFACTURER

 AMERICAN
```

Hints

You could use the DISPLAY statement in a REPEAT-STORE compound statement to send an intermediate message to the user. For example, in the illustrated sequence

below, the reminder to ENTER NEXT RECORD precedes each request for data and sort of documents the beginning of each new record.

```
DTR>REPEAT 3 DISPLAY "ENTER NEXT RECORD" THEN (RET)
DTR>STORE YACHTS USING (RET)
DTR>BUILDER = "DUFOUR" (RET)
```

You may want to consider the use of the concatenation character (¦) to create a more informative message. See the Hint section of the ABORT command (Section 4.13.6).

FOR

4.13.8 FOR Statement

Purpose

This statement specifies that the substatement that follows must be performed on the group of records identified by the record selection expression.

Format

FOR rse-1 statement-1

1. You must supply a record selection expression. See Section 4.5.

2. The substatement statement-1 may be a command or two or more commands linked by the word THEN (Section 4.13.4).

Requirements

The domain identified by the rse must be READY.

You must obey the requirements of any command you invoke as statement-1.

Prompts & Responses

The prompts and responses will be dictated by your choice of command(s) for statement-1.

Effects

1. On READY domains, there is no prerequisite that FIND (unless CURRENT is specified in the rse) and/or SELECT be executed. The collection will be temporarily "found" **(but not established)** for you and processed. This part of the processing will not affect the cursor either.

2. The final effects will depend on your choice of command(s) for statement-1.

Example

The following example identifies the records of yachts manufactured by Pearson for modification of the price.

```
DTR>FOR YACHTS WITH (RET)
DTR>BUILDER EQ "PEARSON" PRINT MODEL, PRICE THEN (RET)
DTR>MODIFY PRICE; (RET)

 MODEL     PRICE

   26      $11,000
PLEASE SUPPLY VALUE FOR PRICE: 11500 (RET)

   26W     $10,000
PLEASE SUPPLY VALUE FOR PRICE: 10500 (RET)
DTR>
```

HELP

4.14 GETTING ASSISTANCE WITH THE HELP COMMAND

Purpose

This command requests assistance with one or more commands. Two forms of help are available, since the commands can be thought of in two categories: simple and advanced. You only enter the realm of the advanced form of any command when you employ the optional elements such as the record selection expression.

Format

HELP [**ADVANCED**] [commnd-name-1 [,commnd-name-2...]]

1. If you specify one or more command-names, they must each be legitimate DATATRIEVE commands.

2. You may optionally specify that you wish the more advanced type of assistance by specifying the word ADVANCED.

Requirements

None.

Prompts & Responses

No prompting messages appear, but you will be informed how to use the command or commands you specified.

Effects

None.

Examples

The following example requests help with the simple forms of the FIND and SELECT commands.

```
DTR> HELP FIND, SELECT; (RET)
```

The following example illustrates how to obtain assistance with the more complex forms of the FIND and MODIFY commands.

```
DTR> HELP ADVANCED FIND, MODIFY; (RET)
```

CHAPTER 5

COMMANDS FOR REPORT WRITING

This chapter presents the commands that are part of the DATATRIEVE-11 facility for report writing. These commands are entered in a group or block that you can call the RW sequence. Before learning how to construct an RW sequence, you should review the parts of a report.

5.1 PARTS OF A REPORT

The Report Writer allows you extensive flexibility in designing a report. The flexibility, however, means that a generalized report may have a large number of components. For example, the list of possible components includes report headers, column headers, report sections, detail lines, summary lines, report groups, and report events. Because we will refer to each of these parts by name, it is important for you to understand what they are. Figure 5-1 illustrates a typical report and its parts.

The **REPORT HEADER** is the line or lines that may be printed at the top of your report, usually as a title. The report header may include the report name, date, and page number or any combination of these three items. Nearly all reports will contain a report header on the first page; at your option, you may request that the header be printed on subsequent pages or subsequent sections.

The **COLUMN HEADER** consists of one or more lines printed as the headers for the columns in the report. The column header may include field names, DATATRIEVE keywords, or other headers that you have specified.

A **REPORT SECTION** is just what it sounds like: a section of a report. A report section begins on a new page, numbered one. You might want to divide the report into sections when you plan to separate it and send the various sections to different people.

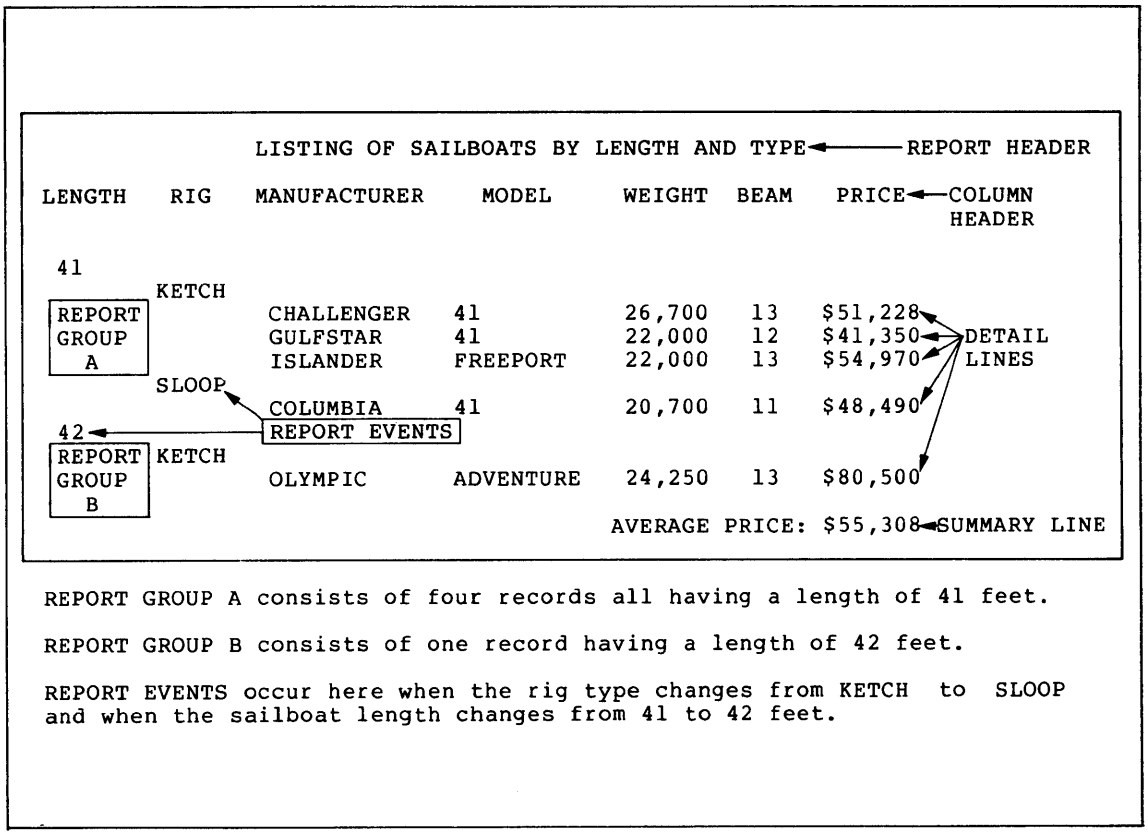
One or more lines in the report that are produced from a data record in the file are collectively called the **DETAIL LINE.** The detail line need not be straight from the file; the RW allows you to add literal strings and arithmetic expressions. For example, you could add an arithmetic expression such as PRICE/DISP to calculate the cost per pound for each yacht. The resulting value would become part of the detail line in the generated report but not part of the data record.

A number of successive input records with a single value for a particular field constitutes a **REPORT GROUP** (sometimes just called a group). The end of a report group is reached when the next record has a different value for the field in question. Report groups are often a useful key for formatting indentation and spacing. You can create subgroups by specifying more than one sort-key.

The beginning or end of a report group or the top or bottom of a page or report is called a **REPORT EVENT.** You can request that various procedures take place when a report event occurs. For example, at the bottom of a report group, you might want to print totals or subtotals for one or more fields.

In addition to invoking special procedures, a report event can trigger a **SUMMARY LINE.** Summary lines can include summary information (for example, any of the statistical functions TOTAL, COUNT, AVERAGE, MAX, or MIN) for a report, page, or report group, but they are not limited to such information. A summary line can also contain header information for groups, pages, or reports, or perhaps start a new page or section.

In the sections that follow you will learn Report Writer statements that specify these parts of a report. After all the statements have been discussed individually, Figure 5-2 illustrates some RW sequences that combine the statements.



REPORT GROUP A consists of four records all having a length of 41 feet.

REPORT GROUP B consists of one record having a length of 42 feet.

REPORT EVENTS occur here when the rig type changes from KETCH to SLOOP and when the sailboat length changes from 41 to 42 feet.

Figure 5-1 Parts of a Report

REPORT

5.2 INVOKING RW -- REPORT

Purpose

This statement invokes the Report Writer facility of DATATRIEVE-11. You may obtain a report on any collection or domain that is already established.

Format

REPORT [rse] [**ON** file-spec-1]

where the record selection expression (rse) assumes the form:

[**ALL** | **FIRST** n] [collectn-name-2 **IN**] {CURRENT | collectn-name-3 | domain-name-1} [**WITH** conditn]

[**SORTED** BY key-1[,key-2...]]

Note that n represents any decimal integer.

If necessary, review the descriptions of record selection expressions in Section 4.5.3.

1. If you specify a record selection expression, you will obtain a report on the records in the collection or domain you designate.

2. If you omit the record selection expression, the report will include the CURRENT collection data only.

3. If you specify the optional [ON file-spec-1] clause, file-spec-1 must be the name of a file or device specification. If you do not designate a special file specification, the report appears on your terminal. Note that you can use the value expression *.prompt-name (described in Section 4.5.1) to obtain a file or device specification.

Requirements

You must have already established a collection if you specify CURRENT or a collection name.

Prompts & Responses

None.

Effects

1. The report reflects only the CURRENT collection data unless you supply a collection name in the record selection expression.

2. The output of the report is directed to the terminal unless you provide an alternate file specification in the ON file-spec-1 clause.

3. The Report Writer treats all the subsequent input as RW statements until you issue the REPORT END statement.

4. The collection cursor is left unchanged.

Examples

The following example initiates a report for the CURRENT collection to be printed on the terminal.

```
DTR>REPORT; (RET)
```

The following example initiates a report for the MEDIUM-YACHTS collection with the specified characteristics to be printed on line printer unit 2.

```
DTR>REPORT FIRST 7 MEDIUM-YACHTS WITH PRICE< (RET)
DTR>40000 SORTED BY LOA ON LP2:; (RET)
```

Hints

The ability to print the report on your terminal may be especially useful for preparing a "trial run" of a report. This way you can check the spacing and format before you print the report on the line printer or other device.

While there is no provision in the REPORT command syntax for specifying that you wish multiple copies of your report, you could always use the following technique to obtain them. First generate a copy of the report into a named area (for example, REPORT ON REP1.SAV), and then use an operating system command to queue multiple listings of that file from the area.

REPORT END

5.3 TERMINATING THE REPORT -- REPORT END

Purpose

The REPORT END statement is the last statement in an RW sequence. It signals DATATRIEVE that subsequent commands are not to be treated as part of the RW sequence. Upon recognition of the REPORT END statement, DATATRIEVE begins to check the complete RW sequence for syntax errors. If none is found, execution of the report request begins. Otherwise, error messages appear on your terminal and you must correct the errors before your request can be processed.

Format

REPORT END

Requirements

You must have previously issued a REPORT statement, a SET REPORT-NAME statement, and at least one PRINT or AT statement.

Prompts & Responses

None.

Effects

1. DATATRIEVE scans all the RW statements for syntax errors. If none is found, the report you requested will be printed on the specified device. If any errors are present, you must correct them before a report can be printed.

Example

The following example ends a sequence of RW commands.

```
DTR>REPORT END; (RET)
```

SET

5.4 SPECIFYING REPORT PARAMETERS -- SET

Purpose

This statement defines values for one or more parameters throughout the report. You can set the following items:

- Report name
- Maximum number of lines per report
- Maximum number of pages per report
- Page numbering or no page numbering
- Date or no date
- Number of lines per output page
- Number of columns per output page

Format

SET parameter-1 [,parameter-2...]

where the parameters are chosen from the following list:

```
REPORT-NAME = report-name
MAX-LINES = integer-1
MAX-PAGES = integer-2
NUMBER
NO-NUMBER
DATE = ["string-1"]
NO-DATE
LINES-PAGE = integer-3
COLUMNS-PAGE = integer-4
```

1. REPORT-NAME must consist of one or more name segments, separated by slashes. The name segments must be literals (see Section 3.8). DATATRIEVE-11 centers the segments on successive lines. **REPORT-NAME must be set once in every RW sequence.**

2. Integer-1 denotes the maximum number of lines that the report may include. If the number of lines (including blank lines and lines automatically skipped) exceeds this number, DATATRIEVE-11 automatically terminates the report. If you omit this parameter, no checking is done, and the report may run until terminated by outside intervention.

3. Integer-2 indicates the maximum number of output pages that the report may span. DATATRIEVE-11 automatically ends the report after printing this number of pages. If you omit this parameter, no checking is done and the report may run until terminated by outside intervention.

4. NUMBER and NO-NUMBER, respectively, allow you to include or exclude page numbers in the report.

5. DATE and NO-DATE, respectively, allow you to include or exclude the date of your report. The DATE parameter has an optional string-1 clause. If you omit this clause, DATATRIEVE-11 uses the current system date in the format dd-mmm-yy, as in 18-AUG-77. By including the string-1 clause, you can specify a particular date in any format you want. For example, DATE="OCTOBER 31, 1977" or DATE="HALLOWEEN" are two acceptable forms.

6. Integer-3 represents the number of lines to be printed on each output page. If you omit this specification, DATATRIEVE-11 automatically uses a value of 60.

7. Integer-4 denotes the number of columns to be printed across the output page. If you omit this value, DATATRIEVE-11 automatically uses a value of 132. The maximum value of integer-4 is 132.

Requirements

None.

Prompts & Responses

None.

Effects

1. DATATRIEVE-11 sets values for all the parameters included in the SET statement. These values pertain to the entire report.

2. If certain parameters are **not** set, DATATRIEVE-11 adopts the default values shown in Table 5-1.

Examples

The following example sets a report name.

```
DTR>SET REPORT-NAME = "YACHT SALES"/ (RET)
DTR>"IN CALENDAR YEAR 1977"/ (RET)
DTR>"STAR-KEY MARINA, INC." (RET)
```

The second example sets a value of 58 lines per page and 72 columns per page.

```
DTR>SET LINES-PAGE = 58, COLUMNS-PAGE = 72; (RET)
```

Hint

If you need to set several parameters in the same report, you can use more than one SET statement. The additional SET statements may not, however, be used to change parameters that have already been set.

Table 5-1
Default Report Parameters

Parameter	Default Value
REPORT-NAME	None. Required
MAX-LINES	None. Not required.
MAX-PAGES	None. Not required
NUMBER	Pages will be numbered.
DATE	Current date in computer is used.
LINES-PAGE	60
COLUMNS-PAGE	132

PRINT

5.5 FORMATTING DETAIL LINES -- PRINT

Purpose

This statement specifies the content and format of a detail line in the report.

Format

PRINT detail-item-1 [,detail-item-1...]

1. Detail-item-1 and detail-item-2 are the same print list elements that were defined in Table 4-5. See Section 4.8 for a complete description of the PRINT command.

2. You may specify only one PRINT statement to specify the detail line format in each report specification.

Requirements

None.

Prompts & Responses

None.

Effect

1. The RW produces a detail line for each record in the collection or domain on which the report is based.

Example

The following example produces a detail line that starts with nine spaces, and then lists the yacht displacement and price as recorded in the data record. Finally, the price of the boat is divided by its weight, and the result is printed in the last column under the heading "PRICE PER POUND."

```
DTR>PRINT SPACE 9, DISP, PRICE, (RET)
DTR>PRICE/DISP ("PRICE"/"PER"/"POUND") USING (RET)
DTR>$ZZ,ZZZ.99; (RET)
```

5.6 FORMATTING SUMMARY LINES -- AT

Purpose

This statement specifies the content and format of a summary line in the report. A summary line may appear at the top or bottom of a page, report, or field. It may include a value, the result of an arithmetic calculation, or a report header. The summary line may also specify special spacing instructions, a new page, or a new section of the report.

Format

```
AT { TOP    } OF { field-name-1 } PRINT  summry-item-1 [,summry-item-2...]
   { BOTTOM }    { PAGE        }
                 { REPORT      }
```

1. Field-name-1 must name a field that belongs to the record type associated with the collection on which you are reporting.

2. Table 5-2 lists all the legal values for summary items. Note that since a modified PRINT statement is part of the AT statement, many of the legal options of the PRINT statement may be used in the summary items as well.

3. The value expression is evaluated using the first or last record of the group, the page, or the collection, depending on the summary line specification.

4. If you specify TOTAL, COUNT, AVERAGE, MAX, or MIN, RW computes the quantity for the field, page, or report. These specifications may only be part of an AT BOTTOM statement; you cannot summarize a column, page, or report in this manner at the top.

5. NEW-PAGE signals two report events:

   ```
   BOTTOM OF PAGE
   TOP OF PAGE
   ```

   Any procedures specified for the top or bottom of the page will be executed whenever a NEW-PAGE is specified.

6. NEW-SECTION signals three report events:

   ```
   BOTTOM OF PAGE
   TOP OF REPORT
   TOP OF PAGE
   ```

   After each new section is started, RW resets the page numbers to 1.

7. Unless you explicitly specify otherwise, RW automatically performs the following statement each time it starts a new page:

   AT TOP OF PAGE PRINT REPORT-HEADER, COLUMN-HEADER;

8. If you sorted the collection before invoking RW, RW maintains the order imposed by the sort.

   If you specify more than one AT TOP or AT BOTTOM statement, and the values of one or more fields change between successive input records, RW does not execute the AT TOP and AT BOTTOM statements in the order in which you typed them. Instead, it executes them in the order in which you specified the sort-keys before invoking the Report Writer.

Requirements

None.

Prompts & Responses

None.

Effect

1. If you specify more than one summary item, the procedures the summary items imply are executed in the order in which they are implied. For example, items specifying AT TOP result in printing at the top of the named item, while those carrying AT BOTTOM notations are saved and printed at the end of the report group.

Examples

The following example identifies the report event of a change in the length-over-all field, and causes the new LOA value to be printed under the column header LENGTH at the beginning of each report event.

```
DTR>AT TOP OF LOA PRINT LOA("LENGTH"); RET
```

The next example prints both the report name and column headers at the beginning of each new page.

```
DTR>AT TOP OF PAGE PRINT REPORT-HEADER, COLUMN-HEADER RET
```

Table 5-2
Summary Items

Item	Function
COL n	Advances across the horizontal line to column n. Note: n must not be less than one or more than the page width. If n is less than the present column, a line is skipped.
SPACE [n]	Causes n horizontal spaces between entries on each line.
SKIP [n]	Moves n blank lines anead to start a new line. Printing starts in column one, unless another element modifies the position further.
TOTAL field-1	Causes a total to be printed for the field at the bottom of the page or report.
COUNT	Causes the number of items in the page, field, or report to be printed.
AVERAGE field-1	Causes the average of all the values for field-1 to be printed.
MAX field-1	Causes the maximum value in the field to be printed.
MIN field-1	Causes the minimum value in the field to be printed.
NEW-PAGE	Forces a new page at this point in the report. Note that this phrase activates any TOP OF PAGE statements that you have specified.
NEW-SECTION	Causes the beginning of a new section of the report. Page numbers are restarted at one.
REPORT-HEADER	Causes the printing of the report header, as specified via the SET REPORT-NAME command.
COLUMN-HEADER	Causes the printing of the column headers.
value-expression [modifiers]	Causes a value to be inserted in the report. See Table 4-5.

NOTE

The TAB specification, although permitted in the PRINT command, is not permitted here.

```
DTR>READY YACHTS WRITE
DTR>FIND YACHTS WITH LOA > 33
[7 RECORDS FOUND]
DTR>REPORT;
DTR>SET REPORT-NAME='TESTING';
DTR>SET COLUMNS-PAGE = 50
DTR>PRINT PRICE,BUILDER,RIG;
DTR>REPORT END;


                    TESTING              28-OCT-77
                                         PAGE 1


 PRICE             MANUFACTURER            RIG

$36,951             ALBERG               KETCH
$37,850             CARIBBEAN            SLOOP
$39,215             CHALLENGER           SLOOP
$37,850             CHRIS-CRAF           SLOOP
$39,500             I. TRADER            KETCH
$36,950             IRWIN                KETCH
$35,900             LINDSEY              M/S


DTR>READY YACHTS
DTR>REPORT YACHTS WITH LOA >36 SORTED BY BUILDER, LOA
DTR>SET REPORT-NAME = 'YACHT SALES'/
DTR>'IN CALENDAR YEAR 1977'/
DTR>'STAR-KEY MARINA, INC.'
DTR>SET LINES-PAGE = 58, COLUMNS-PAGE = 72;
DTR>PRINT SPACE 9, DISP, PRICE,
DTR>PRICE/DISP ('PRICE'/'PER'/'POUND') USING
DTR>$ZZ,ZZZ.99;
DTR>AT TOP OF LOA PRINT LOA ('LENGTH')
DTR>AT TOP OF PAGE PRINT REPORT-HEADER,-
DTR>COLUMN-HEADER;
DTR>REPORT END;

                              YACHT SALES
                         IN CALENDAR YEAR 1977                28-OCT-77
                         STAR-KEY MARINA, INC.                PAGE 1


                                                               PRICE
                                                                PER
LENGTH    WEIGHT                 PRICE                         POUND


 37
         20,000                 $36,951                    $       1.00
         20,000                 $36,950                    $       1.00
 39
         14,500                 $35,900                    $       2.00
```

Figure 5-2 Sample Report Writer Sequences

```
DTR>READY YACHTS
DTR>FIND YACHT-TYPES IN YACHTS SORTED BY ASC RIG,
DTR>BUILDER, LOA;
[35 RECORDS FOUND]
DTR>SELECT FIRST;
DTR>REPORT YACHT-TYPES;
DTR>SET REPORT-NAME="YACHTS GROUPED BY TYPE";
DTR>SET COLUMNS-PAGE = 72
DTR>AT TOP OF RIG PRINT RIG
DTR>PRINT BUILDER, LOA, DISP, BEAM, PRICE;
DTR>AT BOTTOM OF RIG PRINT COUNT
DTR>REPORT END;
```

YACHTS GROUPED BY TYPE 28-OCT-77
PAGE 1

RIG	MANUFACTURER	LENGTH OVER ALL	WEIGHT	BEAM	PRICE	
KETCH						
	ALBERG	37	20,000	12	$36,951	
	GRAMPIAN	33	12,000	10	$29,675	
	I. TRADER	36	18,600	12	$39,500	
	IRWIN	37	20,000	11	$36,950	
						4
M/S						
	AMERICAN	26	5,500	08	$18,895	
	EASTWARD	24	7,000	09	$15,900	
	LINDSEY	39	14,500	12	$35,900	
						3
SLOOP						
	ALBIN	26	4,200	10	$17,900	
	ALBIN	27	5,070	08	$18,600	
	AMERICAN	26	4,000	08	$9,895	
	BAYFIELD	32	9,500	10	$32,875	
	CAPE DORY	19	1,900	06	$4,295	
	CAPE DORY	25	4,000	07	$8,995	
	CAPE DORY	28	9,000	09	$21,990	
	CARIBBEAN	35	18,000	11	$37,850	
	CHALLENGER	32	12,800	11	$31,835	
	CHALLENGER	35	14,800	12	$39,215	
	CHRIS-CRAF	35	18,000	11	$37,850	
	CLIPPER	30	3,800	08	$9,500	
	CLIPPER	32	4,500	08	$12,950	
	GRAMPIAN	26	5,600	08	$11,495	
	GRAMPIAN	28	6,900	10	$14,475	
	HUNTER	27	6,500	09	$14,999	
	IRWIN	25	5,400	12	$10,950	
	ISLANDER	24	4,200	08	$6,500	
	ISLANDER	28	5,994	10	$15,908	
	RYDER	31	13,600	00	$32,500	
	SABRE	28	7,400	09	$22,000	
	SALT	25	2,600	07	$6,590	
	TANZER	26	4,350	09	$11,750	
	TANZER	28	6,800	10	$17,500	
	VENTURE	21	1,500	07	$2,823	
	VENTURE	22	2,000	07	$3,564	
	WESTERLY	26	6,700	08	$15,245	
	WINDPOWER	16	650	07	$3,500	
						28

Figure 5-2 (Cont.) Sample Report Writer Sequences

```
DTR>!THE RW SEQUENCE THAT CREATED FIGURE 5-1.
DTR>!
DTR>READY YACHTS
DTR>REPORT YACHTS SORTED BY ASC LOA, RIG
DTR>SET REPORT-NAME="LISTING OF SAILBOATS BY LENGTH AND TYPE"
DTR>SET LINES-PAGE=15
DTR>SET COLUMNS-PAGE=70
DTR>AT TOP OF LOA PRINT LOA("LENGTH");
DTR>AT TOP OF RIG PRINT RIG("RIG");
DTR>AT BOTTOM OF REPORT PRINT COL 30,-
DTR>"AVERAGE PRICE: ",AVERAGE PRICE USING $ZZ,Z99 ;
DTR>PRINT TYPE, DISP, BEAM USING Z9, PRICE;
DTR>REPORT END;
```

Figure 5-2 (Cont.) Sample Report Writer Sequences

CHAPTER 6

PROCEDURES

6.1 INTRODUCTION TO PROCEDURES

Procedures are groups of DATATRIEVE statements and commands that you want to name so that you can cause them to execute without having to reenter them each time they are needed. Whether or not you become involved in defining procedures, you will probably want to invoke some of them as your use of DATATRIEVE expands. The use of procedures can save you a great deal of time at the keyboard. Since the sequence of commands has been "debugged" prior to its entry in the Data Dictionary, you can avoid some of the common errors of writing DATATRIEVE commands and statements extemporaneously.

6.2 INVOKING A PROCEDURE

Once a procedure has been defined and placed in the Data Dictionary, you may invoke it by name any time you need it. The technique is to issue a colon (:) immediately prior to the procedure name. If you are not sure of the procedure name, you can issue a SHOW PROCEDURES command (Section 4.3).

To illustrate this technique, suppose you know there is a procedure called PRICE-INCR in your Data Dictionary. Once you ready the domain YACHTS for writing, you can invoke PRICE-INCR to display both the current value of the price field and the value of the price if increased by 10 percent. The procedure asks you to supply the value of the price, so that you can modify the old value to the newly calculated figure. It repeats this process for every yacht with a price greater than zero. This procedure will produce the same results that you would have achieved by issuing the following sequence of DATATRIEVE commands. (As you would suspect, these commands comprise the procedure definition.)

```
DTR>FOR YACHTS WITH PRICE > 0 (RET)
DTR>PRINT "PRICE: ",PRICE," 10% INCREASE: ",PRICE*1.1 USING (RET)
DTR>$$$,$$$ THEN MODIFY PRICE; (RET)
DTR>FINISH; (RET)
```

Invoking this sequence of DATATRIEVE-11 commands is as simple as calling its name. You simply precede the name by a colon. You may follow it with an optional semicolon if you wish.

For example, using the defined procedure PRICE-INCR we could cause a 10 percent price increase in all the yacht records as follows:

```
DTR>READY YACHTS WRITE; (RET)
DTR>:PRICE-INCR (RET)
     PRICE

PRICE:  $36,951 10% INCREASE:  $40646.10

PLEASE SUPPLY VALUE FOR PRICE:  40646 (RET)
    .
    .
    .
and so forth.
```

You would be informed of the current and calculated price and requested to supply a value for PRICE for each of the records in the domain.

CHAPTER 7

FOR THE DATA ADMINISTRATOR

This chapter covers some topics of special interest to the Data Administrator. The average user will not need to study this information. From this chapter the Data Administrator can learn how to define records, domains, and procedures, how to maintain these definitions, how to use the data protection features for greater installation security, and finally, how to restructure data in existing files into other formats.

7.1 DATA DICTIONARY TOPICS

Before any user can use DATATRIEVE-11 on the records and domains, an individual at the installation must first define the records and domains. These definitions are retained in the Data Dictionary along with the procedure definitions.

In the following sections you will learn how to use the DEFINE command to create the record, domain, and procedure definitions. You will also learn how to remove the definitions from the Data Dictionary.

An Editing Technique

A general suggestion is offered here since you may find yourself wanting to edit your definitions and there is no editing facility that operates on the Data Dictionary; (DATATRIEVE-11 requires that you delete a definition and then redefine it). You might consider instead setting all your definitions up in indirect command files. These files may be edited at will with your editor. When you are satisfied with the definition, you enter DATATRIEVE-11 and simply respond to the DTR> prompt with

```
@file-name-1 (RET)
```

This technique was employed during DATATRIEVE-11 installation.

7.1.1 Defining the Records--DEFINE RECORD

You must define each field in each record, its size and general characteristics. While COBOL programmers will find the format of the definitions to be familiar, you should not assume that the RMS-11 files need to be generated by COBOL programs or that any other knowledge of the COBOL language is required. This is simply a convenient format for describing the data items and their relationships to one another in each record. Even if you have never studied COBOL, you should be able to use the elements comfortably once you see a few examples. The pertinent COBOL rules have been extracted and summarized. They are presented in Appendix E.

DEFINE RECORD

Purpose

This command creates a complete record definition in the Data Dictionary. This definition will permit references to the record and its fields. It is necessary, however, that the record also be defined as part of a defined domain. This action can take place either before or after the record definition, provided it occurs before a user attempts the first reference to the record. Section 7.1.2 describes how to define a domain.

Format

DEFINE RECORD record-name-1 USING data-def-1 [data-def-2...] ;

1. You must specify a record name. (See Section 3.7.)

2. Each record must consist of at least one data item (field) requiring a data definition. Additional fields will require additional data definitions. The definitions must be given in the same sequential order as the fields exist in the record.

3. Data definitions will assume the format of COBOL data definitions, employing the technique of a PICTURE clause to describe the type of data (for example, numeric and alphanumeric) its length, and permitted contents. This approach also supports the use of level-numbers to describe subfields within major fields. Each data definition assumes the following format:

   level-no fld-nme-1 [**PIC** IS char-string
   [**USAGE** IS usage] [**SIGN** IS qualifiers]]

   You will find descriptions of the optional PIC, USAGE, and SIGN clauses in Appendix E.

   You may also intersperse optional clauses such as

   [**QUERY-NAME** IS name].
   or
   [**QUERY-HEADER** IS "header-1" [/"header-2"...]].
   or
   [**EDIT-STRING** IS edit-string].

   These clauses designate abbreviations for the field name or abbreviations for the header to be used when printing the field on reports, or they specify an editing pattern to control the format of the data when it is output. The EDIT-STRING clause is described in Appendix E.

4. Observe that each data definition elementary data item terminates with a mandatory period.

5. The semicolon is required for command termination. You may also separate the optional clauses by semicolons, if you wish. To avoid any possible misinterpretation about the period that ends a data

definition line and the period that represents a decimal point in a character string, particularly when followed by a semicolon, you should adopt the following practice:

Place the final semicolon that terminates the record definition on a line by itself.

Requirements

You must not use any of the keywords in your definitions as either field names or record names. These keywords are described in Section 3.11. You also must not duplicate previously defined record, domain, or procedure names.

Restriction

The DEFINE RECORD command is not permitted in a procedure definition.

Prompts & Responses

None.

Effects

1. Provided you have chosen a unique name for the record, your new definition will be placed in the Data Dictionary.

2. A password table (Section 7.2) will be created for this resource and your UIC will be entered with full access privileges (RWEMC).

3. If the record name you choose is not unique, your command will be rejected. (To modify an existing record definition, you must first delete the old version with the DELETE command (Section 7.1.4), then issue a new DEFINE RECORD command--unless you have followed the suggested editing technique of Section 7.1 and used indirect command files.)

Example

The following example defines the yacht record as used in a number of examples throughout this manual.

```
DTR>DEFINE RECORD YACHT USING (RET)
DTR>01 BOAT. (RET)
DTR>    03 TYPE. (RET)
DTR>       06 MANUFACTURER PIC X(10) (RET)
DTR>          QUERY-NAME IS BUILDER. (RET)
DTR>       06 MODEL PIC X(10). (RET)
DTR>    03 SPECIFICATIONS (RET)
DTR>       QUERY-NAME SPECS. (RET)
DTR>       06 RIG PIC X(6). (RET)
DTR>       06 LENGTH-OVER-ALL PIC XXX (RET)
DTR>          QUERY-NAME IS LOA. (RET)
DTR>       06 DISPLACEMENT PIC 99999 (RET)
DTR>          QUERY-HEADER IS "WEIGHT" (RET)
DTR>          EDIT-STRING IS ZZ,ZZ9 (RET)
DTR>          QUERY-NAME IS DISP. (RET)
DTR>       06 BEAM PIC 99. (RET)
DTR>       06 PRICE PIC 99999 (RET)
DTR>          EDIT-STRING IS $$,$$$. (RET)
DTR> ; (RET)
```

Hints

The SHOW record-name command (Section 4.3) will allow you to confirm that a record definition has been created as you planned. This is a good practice because it is important to keep up-to-date copies of the record definitions, in case they should be lost or damaged during operation.

7.1.2 Defining the Domains--DEFINE DOMAIN

For each domain, there can be only one record type as defined by the DEFINE RECORD command described in Section 7.1.1. However, different domains may reference the same record type, if desired. In this section you will learn how to define a domain.

DEFINE DOMAIN

Purpose

This command defines a domain of one kind of record. Without this definition, which is kept in the Data Dictionary, no references can be made to the records.

Format

DEFINE DOMAIN domain-nme-1 **USING** rec-nme-1 ON rms-file-spec-1 ;

1. You must supply a domain-name, and it must be unique.

2. You must also provide a record name for a record that has been or will be defined to exist in this domain.

3. You must provide an RMS-11 file specification that identifies the device and file name and extension where the domain will reside, or at least exist, during operations.

4. You must not use any of the keywords (Section 3.11) for your domain or record names. Furthermore, you must not duplicate previously defined domain, record, or procedure names when establishing the new domain name. The record name must uniquely identify its record definition.

5. You must terminate the command with a semicolon.

Requirements

Use only unique names for domains and records.

Restriction

You must not use the DEFINE DOMAIN command in a procedure definition.

Prompts & Responses

None.

Effects

1. Provided you have chosen a unique domain name for the domain, your new definition will be placed in the Data Dictionary.

2. A password table (Section 7.2) will be created for this resource and your UIC will be entered with full access privileges (RWEMC).

3. If the domain name you choose is not unique, your command will be rejected. (To modify an existing domain definition, you must first delete the old version with the DELETE command (Section 7.1.4), and then issue a new DEFINE DOMAIN command. See also the alternate method suggested in Section 7.1.)

Example

The following example defines the domain YACHTS, using the record called YACHT, from the file YACHT.DAT. Observe that the domain name is the plural form, YACHTS, while the record name is singular. This makes the two names unique, although the difference is somewhat subtle.

```
DTR>DEFINE DOMAIN YACHTS USING YACHT ON (RET)
DTR>YACHT.DAT; (RET)
```

Hint

The SHOW domain-name command (Section 4.3) will confirm that a domain definition has been created as you planned.

7.1.3 Defining a Procedure -- DEFINE PROCEDURE

Before any user can invoke a procedure, it must be defined and catalogued. One command is used to do this: DEFINE PROCEDURE.

DEFINE PROCEDURE

Purpose

This command creates a procedure definition that is automatically catalogued in the Data Dictionary. This procedure definition becomes available immediately to you and any other user. It remains available until it is deleted by means of the DELETE command.

Format

DEFINE PROCEDURE procedure-name-1

. } DATATRIEVE-11 commands and statements

END-PROCEDURE ;

1. You must supply a procedure name, which must conform to the rules for names in Section 3.7.

2. You may specify one or more DATATRIEVE commands or statements, optionally separated by semicolons.

3. The end of the procedure definition is signalled by an END-PROCEDURE statement.

Requirements

None.

Restrictions

You must not include any of the following commands that require access to the Data Dictionary: READY, SHOW, DEFINE RECORD, DEFINE DOMAIN, DEFINEP, DELETE, DELETEP, or SHOWP.

You cannot nest DEFINE PROCEDURE commands. In other words, do not attempt to create a procedure within a procedure.

Prompts & Responses

None.

Effects

1. The procedure, consisting of all the commands and statements you specify, is automatically catalogued by name in the Data Dictionary.

2. Any comments you specify in the definition are not catalogued.

3. A password table (Section 7.2) will be created for this resource and your UIC will be entered with full access privileges (RWEMC).

Examples

The following example shows how to create a procedure definition that will produce a report of yacht types for the YACHTS domain every time the procedure is invoked.

```
DTR>DEFINE PROCEDURE YACHT-TYPE-REPORT (RET)
DTR>REPORT YACHTS SORTED BY ASC RIG, BUILDER, LOA (RET)
DTR>SET REPORT-NAME="YACHTS GROUPED BY TYPE"; (RET)
DTR>PRINT RIG, BUILDER, LOA, DISP, BEAM, PRICE; (RET)
DTR>AT BOTTOM OF RIG PRINT COUNT (RET)
DTR>REPORT END; (RET)
DTR>FINISH; (RET)
DTR>END-PROCEDURE; (RET)
```

Hints

Test your procedures as DATATRIEVE-11 commands before you create and catalogue the formal definitions. This permits you to debug them first. Otherwise, you may create a definition, try it, find it doesn't work, and be faced with deleting it so that you may reenter the entire piece of code again. You cannot simply change a line of code in the definition. If a definition is in error, it must be first removed, then reentered.

See also the editing technique suggested in Section 7.1 as an alternative.

7.1.4 Deleting a Previous Definition -- DELETE

Before you can redefine any domain, record, or procedure, you are required to delete the existing definition. This requirement helps ensure that definitions are not mistakenly corrupted, and lends support to the requirement that only unique names may be used.

DELETE

Purpose

This command deletes a previous definition of either a domain, procedure, or record from the Data Dictionary. Once the item is removed, no user can successfully reference it.

Format

```
DELETE { domain-name-1    }
       { record-name-1    } [(password-str-1)];
       { procedure-name-1 }
```

1. You must supply your choice of either a domain name, a record name, or a procedure name.

2. You must terminate the command with a semicolon.

3. You must possess C(ontrol) access privileges for the named resource. This will be determined from your UIC or the optional password you may supply. See Section 7.2 for more information on passwords.

Requirements

The name specified for the domain, record, or procedure must match a name defined in the Data Dictionary.

You must possess C(ontrol) access privileges.

Restriction

You must not include the DELETE command in a procedure definition.

Prompts & Responses

None.

Effects

1. The **definition** of the record, domain, or procedure is removed from the Data Dictionary. When dealing with records and domains, note that only the definition is removed. There is no effect on the physical file: neither the file nor its records are deleted. Only the ability to reference the data through DATATRIEVE-11 is lost, at least until such time as new definitions are provided.

2. The password table for the named resource is also removed from the Data Dictionary.

Examples

The following example deletes the definition of the domain called YACHTS, assuming that your UIC possesses C privileges.

```
DTR>DELETE YACHTS; (RET)
```

The following example illustrates how to remove the definition of the yacht record from the Data Dictionary using a password that has been accorded C privileges.

```
DTR>DELETE YACHT (SUNNY-DAY); RET
```

The last example illustrates how you might delete the definition of the procedure called PRICE-PER-POUND.

```
DTR>DELETE PRICE-PER-POUND; RET
```

Hints

If critical definitions are accidentally deleted with this command, you can replace them with the DEFINE command.

However, this brings up another advantage of using indirect command files for all your definitions, as suggested in Section 7.1. Restoring a lost definition can be as easy as

```
DTR>@PRODF1 RET
```

if the definition is stored in a command file such as PRODF1.CMD, in this case.

7.2 DATA PROTECTION FEATURES

The protection of data within DATATRIEVE-11 is accomplished through two independent mechanisms: the protection systems of RMS-11 and those within DATATRIEVE-11 itself. The DATATRIEVE-11 user can regulate access to domains, records, and procedures through access requirements recorded with the definitions in the Data Dictionary.

Before you can learn the commands that direct and control the security features of DATATRIEVE-11, you must learn some basic concepts of the design.

7.2.1 Two Types of Lock

This security system employs two different types of lock:

PW (Passwords) -- character strings of up to ten characters that the user must employ with certain commands (for example, SUNNY-DAY and FAIRWINDS).

UIC† account numbers -- numbers known to the operating system that DATATRIEVE-11 can check, that the user need not specify. Thus, this is the default lock-type that is tested when required (for example, [305,305] and [*,*]).

These locks and keys are maintained on a domain, record, and procedure basis. That is, **each resource has its own password table** identifying the password strings or user account numbers that may have access to the resource. Furthermore, this table tracks the type of access that may be granted the user whose key satisfies the key and lock-type requirements.

7.2.2 Five Types of Privileges

The types of access are called the user's **privileges.** Thus, even though you may be allowed to modify the YACHTS domain, your co-worker Steve may only be permitted to read the records in the same domain. The types of privileges are defined in Table 7-1.

7.2.3 Password Table Structure

Figure 7-1 illustrates the password table structure using a possible example for the YACHTS domain. Each sequential line is a **password table entry.**

There will never be a vacancy in the sequential table entries. DATATRIEVE-11 processing of the DEFINEP and DELETEP commands ensures this.

---

† UIC is the User Identification Code specification that refers to the user's disk directory and is expressed syntactically as [m,n] where m and n may be integers or an asterisk (*).

Table 7-1
Privilege Codes

Privilege Code	Meaning
R	R(ead)-- User can only retrieve this resource.
E	E(xtend)--User can only add records to this resource.
M	M(odify)--User can retrieve or change records in this resource.
W	W(rite)-- User can do all of the above (retrieve, change, and add), and can also erase records.
C	C(ontrol)--User can issue the data protection commands: DELETEP, DEFINEP, and SHOWP, as well as DELETE.

There can be from zero to five privileges granted per lock-type and key entry. However, since a W privilege grants R, E, and M privileges anyway, there will seldom be more than four privileges per entry.

The number of entries in each table is unrestricted and all entries are dynamically allocated.

Sequence Number	Lock Type	Key	Privileges
1	PW	FAIRWINDS	CW
2	PW	SUNNY-DAY	W
3	UIC	[304,303]	"6"
4	PW	STAR-KEY	R
5	UIC	[305,305]	CREM
6	UIC	[301,314]	C
7	UIC	[*,305]	REM

Figure 7-1. Example of Password Table Structure

7.2.4 Creating and Maintaining Password Tables

The password tables are automatically started and given a single entry at resource definition time. This entry identifies the creating user (by UIC) and permits the creator the full set of privileges (REMWC), for this resource.

One of the Data Administrator's major responsibilities is to define the additional password table entries for each resource. These additional entries are created by means of the DEFINEP command. They are maintained by the DELETEP and DEFINEP commands. Since the authorized Data Administrator also needs to obtain printouts of the tables for review, the SHOWP command is designed for this purpose.

Entries are created or deleted one at a time, except when the entire table is erased. The password table for any resource is erased whenever that resource is deleted by means of the DELETE command. (See Section 7.1.4.)

These table creation and maintenance commands are so important to the protection of the data that only users with the C(ontrol) privilege will be allowed to invoke them.

7.2.5 Password Table Processing

Whenever there is a question of access to a domain, record, or procedure, the password table for that resource is examined. Access privileges are checked for any of the following commands:

- READY
- SHOW domain-name
- SHOW record-name
- SHOW procedure-name
- DEFINEP
- DELETEP
- SHOWP
- DELETE

Password tables are processed in a standard fashion that has important implications for the table entry designer. The rules are summarized below and illustrated by the flowchart in Figure 7-2.

1. Users who are considered "privileged" by TRAX because they have received UIC values less than 8, are automatically given the C(ontrol) access privilege. They may receive additional privileges as a result of the next two steps that otherwise apply to less privileged users.

2. Table entries are searched sequentially looking for the first hit. Thus, if a particular user qualifies for access under a number of possible lock-type and key combinations, the access privileges granted will be the first one(s) encountered.

3. The comparison of keys is directed by the lock-type specification **in the table** (not by the user's password specification or default UIC). If the table entry calls for a PW (password) lock-type, the user's command is checked for inclusion of a password that matches. If the table entry calls for a UIC, the user's UIC is automatically checked.

The combined effect of these three rules is that even if the user has a password and specifies it with the command, it will serve no purpose if the user qualifies for access under a UIC type of table entry that DATATRIEVE-11 encounters first.

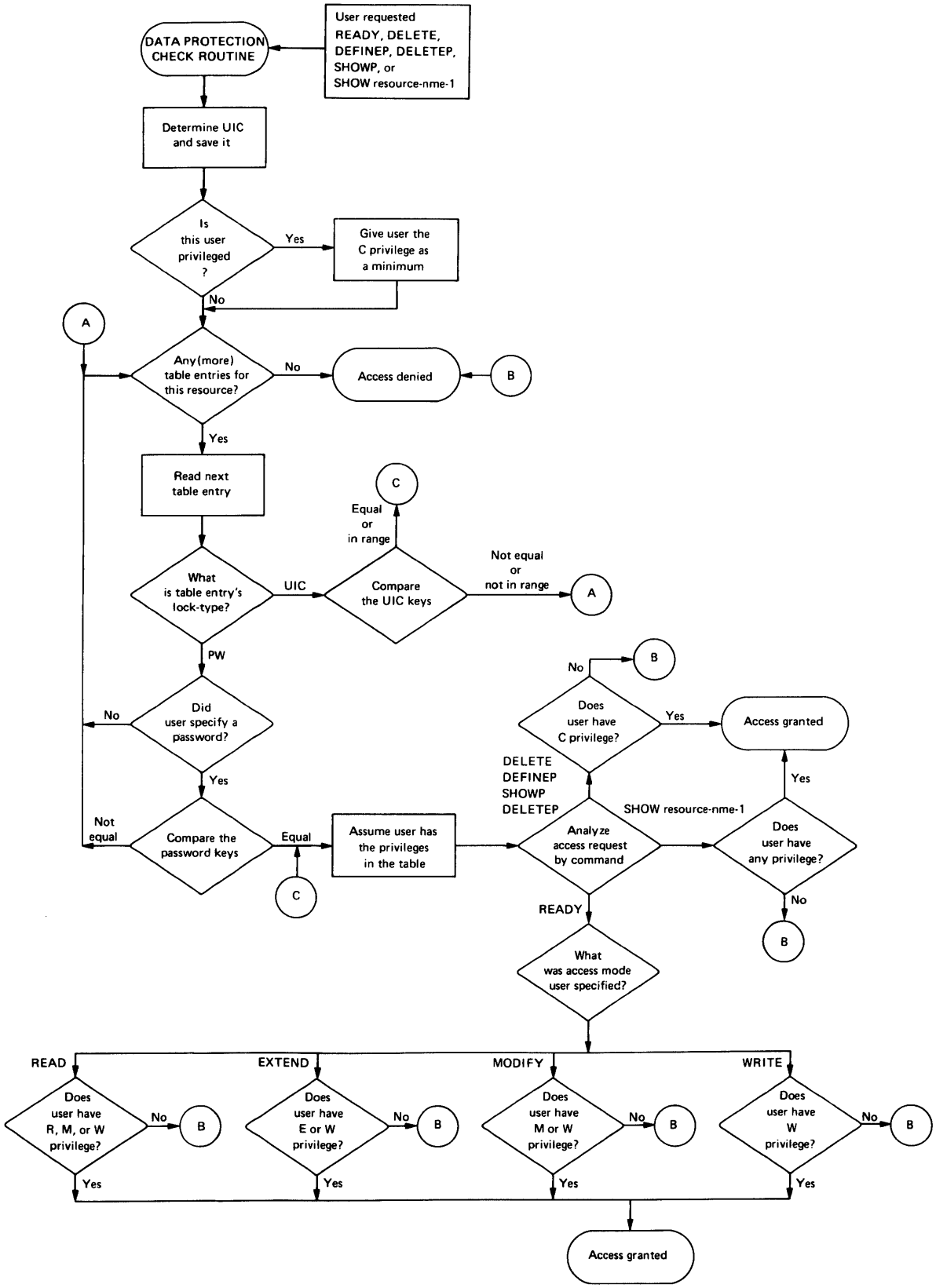


Figure 7-2 Flowchart of Password Processing Logic

The following examples test a few user attempts to gain access to the YACHTS domain against the sample table in Figure 7-1.

Case 1

The user of account [304,303] issues a SHOW YACHTS command. No password is given. The first possible hit occurs on the third table entry and user [304,303] is denied access privileges. (However, if this user knew either of the two passwords FAIRWINDS or SUNNY-DAY, access would have been granted. This would probably defeat the intent of the designer who intended to lock that account out.)

Case 2

The user of account [301,305] issues a READY YACHTS WRITE command. No password is given. The first possible hit occurs on the seventh table entry. However, W is not one of the privileges granted in the table, so the request is denied.

Case 3

The user of account [301,314] issues a READY YACHTS (STAR-KEY) command. The default access requested is SHARED READ. There is a hit on the fourth entry because the password matches and this user receives retrieval privileges.

7.2.6 Coding the Table Entries

Table 7-2 summarizes the types of privileges required to use certain important commands. The table may help you construct and then mentally check your password table designs prior to putting them into effect. As you code each entry you may find it helpful to think in the following terms:

```
...  to enable this user to issue these commands,
          ...  give these privileges...
```

Where the table shows two privileges that afford the use of the same command, you may choose to give either one. However, keep in mind that the W privilege is more all-encompassing. Don't give W privileges to the user you want to be able to MODIFY but not ERASE, because although W permits MODIFY commands, it also permits ERASE commands. Instead, give M privileges. For this reason, X's under W in Table 7-2 are shown in parentheses.

Work your way down the table, and then throw away any duplicate privileges. Whenever you have WR, WM, or WE combinations, you may simplify them to just W. Likewise, MR is redundant; use just M.

Table 7-2
Privilege Requirements By Command

Command	Permits Also	Privilege Required (choice of 1)				
		C	W	M	E	R
DEFINEP	--	X				
DELETEP	--	X				
SHOWP	--	X				
DELETE	--	X				
SHOW resource-nme-1	--	X	(X)	X	X	X
READY WRITE	ERASE MODIFY, STORE, REPEAT STORE, and retrieval commands†		X			
READY MODIFY	MODIFY and retrieval commands†		(X)	X		
READY EXTEND	STORE, REPEAT STORE		(X)		X	
READY READ	retrieval commands†		(X)	X		X

† The retrieval commands are FIND, SELECT, SHOW CURRENT, SORT, and PRINT. Parentheses indicate give W privileges with care because W permits a number of other capabilities also.

7.2.7 Design Guidelines for Password Tables

The following guidelines are offered as suggestions. They are **not** restrictions.

1. Place any UICs that will be denied privileges among the low-numbered entries. This will avert successful use of passwords to gain access.

2. Place password entries with maximum privileges (such as C and W) next. This could avoid users intended to gain the most powers from matching the less restrictive requirements and gaining fewer privileges instead.

3. Place the least restrictive entries, such as any UICs containing an asterisk or any PWs having commonly known passwords, near the end of the table. As a general rule, you should grant these users the fewest privileges.

4. If you will grant any privileges to **all** users through the UIC designation [*,*], place it as the very last entry. It is meaningless to have entries that follow this one since they will never be reached. It is dangerous for this form to precede the vital entry with C privileges because then no one can print, change, or even delete the table, unless they receive C privileges due to having project codes less than 8.

SUMMARY NOTE

While these data protection features are relatively sophisticated and powerful, they are not impervious to systematic attacks by determined outsiders. They can prevent browsing and discourage a number of accidental errors by unqualified users. They can best be applied in the overall context of careful data management by the installation at large.

To summarize, if the installation as a whole follows tight security regulations, these features will augment those practices.

7.2.8 Adding Entries to the Password Table -- DEFINEP

The user who defines a resource is often the only one to use it so there is no requirement to add entries to the password table. However, whenever it can be anticipated that a resource will be shared, consideration should be given to protecting the resource through additional password table entries. The command that permits the addition of entries is DEFINEP.

DEFINEP

Purpose

This command allows you to add one entry to the password table for whatever resource you name. The resource may be a domain, record, or procedure.

Format

DEFINEP resource-nme-1 [(password-str-1)] seq-number, lock-type-1, key-1, privilege-str-1

1. You must supply a resource name that identifies either a defined domain, record name, or procedure.

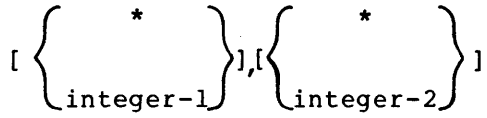
2. Password-str-1 is your personal password enclosed in parentheses. If you do not provide a password, your UIC will be used to check that you have C privileges.

3. You must also supply a sequence number that identifies the table entry in the password table that you will be creating. This sequence number must be an integer.

4. You must specify the lock-type. Valid lock-types are either PW or UIC.

5. You must specify the key that unlocks the lock-type. That is, if the lock-type is given as PW, you must identify the valid password. Passwords are character strings of one to ten characters.

   If the lock-type is given as UIC, then you must identify which UIC will be permitted access. The UIC format is:

   The square brackets in this case are **required** elements. They differ from syntax brackets in that they are part of the command input line.

6. You must specify the privileges to be granted. Choose from any or all the characters in Table 7-1. If you wish this particular user to receive no privileges, you may specify "", where the quotes are required.

Requirements

You must have C(ontrol) access privileges.

The named resource must have been previously defined.

Restriction

You must not include the DEFINEP command in a procedure definition.

Prompts & Responses

None, unless you specify an asterisk (*) for the optional password. This will cause the prompt:

PLEASE SUPPLY PASSWORD:

to appear. In this prompt mode your password specification is not echoed on your terminal -- for greater password security.

Effects

1. If you are deemed to have C(ontrol) privileges, a table entry will be created in the password table of the resource you name. This table resides in the Data Dictionary.

2. If the sequence number you specify either:

   a. already exists -- the new entry takes its place and it and all higher numbered entries are moved ahead by one, or

   b. is greater than the last existing one plus one --the number is ignored and the entry becomes the next sequential one. For example, there are five entries when you specify a sequence number of nine. Your new entry becomes the sixth one, not the ninth.

Examples

The following example illustrates how to add the fourth password table entry for the YACHTS domain table.

```
DTR>DEFINEP YACHTS (FAIRWINDS) 4,PW,STAR-KEY,R (RET)
```

The following example illustrates how to create the sixth password table entry for the same YACHTS domain table.

```
DTR>DEFINEP YACHTS (FAIRWINDS) 6,UIC,[302,314],C (RET)
```

Hint

To avoid errors, always obtain a current copy of the password table prior to making any additions. (See Section 7.2.10, SHOWP.) Due to the manner of table processing, the sequence of a particular entry can greatly affect the efficacy of the rest of the entries.

7.2.9 Deleting a Password Table Entry -- DELETEP

While maintaining a password table, you may need to delete one or more entries. The DELETEP command permits this. However, if your goal is to delete the entire table, you should use the proper form of the DELETE command instead.

DELETEP

Purpose

This command deletes one table entry from the password table of the named resource. The resource may be a domain, record, or procedure.

Format

DELETEP resource-nme-1 [(password-str-1)] seq-number

1. You must supply a resource name that is either a defined domain, record, or procedure.

2. Password-str-1 is your personal password enclosed in parentheses. If you do not provide a password, your UIC will be used to check that you have C(ontrol) privileges.

3. You must supply a sequence number that identifies the specific table entry you wish to delete from its password table. This sequence number must be an integer.

Requirements

1. You must have C(ontrol) access privileges.

2. The named resource must have been previously defined.

Restriction

You must not include the DELETEP command in a procedure definition.

Prompts & Responses

None, unless you specify an asterisk (*) for the optional password. This will cause the prompt:

PLEASE SUPPLY PASSWORD:

to appear. In this prompt mode your password specification is not echoed on your terminal -- for greater password security.

Effects

1. If you are deemed to have C(ontrol) privileges, the table entry will be reviewed for deletion.

2. Deletion of the table entry will occur under **both** of the following conditions:

   a. it exists, and

   b. it is **not** the only remaining table entry that can grant C(ontrol) privileges.

Examples

The following example illustrates how to delete the fourth password table entry in the YACHTS domain table.

```
DTR>DELETEP YACHTS (FAIRWINDS) 4 RET
```

To delete the sixth password table entry:

```
DTR>DELETEP YACHTS (FAIRWINDS) 6 RET
```

Hints

To avoid errors, always obtain a current copy of the password table prior to making any deletions. (See Section 7.2.10.) Always check that the entry you wish to delete has not been moved to a different sequence number.

You can only delete one entry at a time. To delete the entire table, use the DELETE command and name the resource.

7.2.10 Printing a Password Table -- SHOWP

A special command is provided to enable users with C privileges to view the password table for any given named resource. It is important that this be done before every change. Printed copies of the table should be either very carefully guarded or destroyed when not in use.

SHOWP

Purpose

This command allows you to obtain a printout of a password table on your terminal device.

Format

SHOWP resource-nme-1 [(password-str-1)]

1. You must supply a resource name that is either a defined domain, record, or procedure.

2. Password-str-1 is your personal password enclosed in parentheses. If you do not provide a password, your UIC will be used to check that you have C(ontrol) privileges.

Requirements

1. You must have C(ontrol) access privileges.

2. The named resource must have been previously defined.

Restriction

You must not include the SHOWP command in a procedure definition.

Prompts & Responses

None, unless you specify an asterisk (*) for the optional password. This will cause the following prompt to appear:

PLEASE SUPPLY PASSWORD:

In this prompt mode your password specification is not echoed on your terminal -- for greater password protection.

Effects

If you have C(ontrol) privileges, and the named resource exists, its password table is printed in its entirety on your terminal device.

Example

The following example illustrates how to print the password table for the YACHTS domain, assuming the password FAIRWINDS has been accorded C(ontrol) privileges.

```
DTR>SHOWP YACHTS (FAIRWINDS) (RET)
```

Hint

If your terminal provides hard copy, take care not to walk away and leave the printout open to public inspection. Given a copy of the table and this manual, most users could gain access to the resource.

7.3 RESTRUCTURING DATA

It is possible to restructure data with DATATRIEVE-11. In this section you will learn a possible technique. In the context of this manual, "restructuring" refers to selecting data fields from the records in one or more files (or subsets of those files) to create records of an altered format in a new file. The new record, as you will see in the example, may also include other data fields that were not part of the original file(s).

The general technique consists of providing record and domain definitions for the new file through the DEFINE RECORD and DEFINE DOMAIN commands. Once you ready all the domains, and the domain that will receive the data must be readied for WRITE or EXTEND access, you simply issue a combination of FOR and STORE to cause the transfer of data. The example below will illustrate this.

In this example the familiar YACHT record is providing the bulk of the data. Assume that for each YACHT record in existence now a new, shorter record is desired that will contain the same builder, model, and price data, but will provide a new field for the quantity. This new field will be called the NUMBER-IN-STOCK. None of the other data items, such as rig, beam, length-over-all, and weight, is desired. The DATATRIEVE-11 record and domain definitions could be formulated as shown below. (It is also necessary that you define the NEW-YACHTS file through RMS-11 and your operating system. The file may be defined as either sequential or indexed sequential, but not relative.)

```
DEFINE RECORD NEW-YACHT USING
   01 BOAT.
      05 BUILDER PIC X(15).
      05 MODEL PIC X(10).
      05 PRICE PIC 9(5).
      05 NUMBER-IN-STOCK PIC 9(3).
;

DEFINE DOMAIN NEW-YACHTS USING NEW-YACHT ON SUMMARY.YCT;
```

Observe that the BUILDER field in the new record has been defined to include five more characters than the original record definition (Section 7.1.1). However, the PRICE and MODEL fields are identical in both record definitions.

The following command sequence will cause the selected data in each record in the YACHTS domain to be copied into a new record in the NEW-YACHTS domain, in an altered format.

```
READY YACHTS
READY NEW-YACHTS WRITE
FOR YACHTS
    STORE NEW-YACHTS USING
          BOAT = BOAT
EXIT
```

Note that the specification of BOAT, the 01-level number item forces all the subordinate-level number items to be included in the STORE function. Only items having matching names in the record definitions (for example, BUILDER, MODEL, and PRICE) are processed. Observe that the match for BUILDER occurs by virtue of the QUERY-NAME. As this implies, matches are not restricted to the primary name of the field. Since the new BUILDER field is larger, it will be padded (in this case, with blanks at the right). Furthermore, since no field in the old definition can be found to match NUMBER-IN-STOCK, that data item will be set to zero in each of the new records. Figure 7-3

illustrates this process graphically.

Had you wished instead to create new records for a **subset** of the old records, you could have used an appropriate record selection expression in the FOR statement. For example, if you knew that this branch of the marina would only carry the less costly boats (priced under $35,000), you could have modified the FOR statement as follows:

```
FOR YACHTS WITH PRICE < 35000
```

The result of such a clause would be to create a subset of the original file of records.

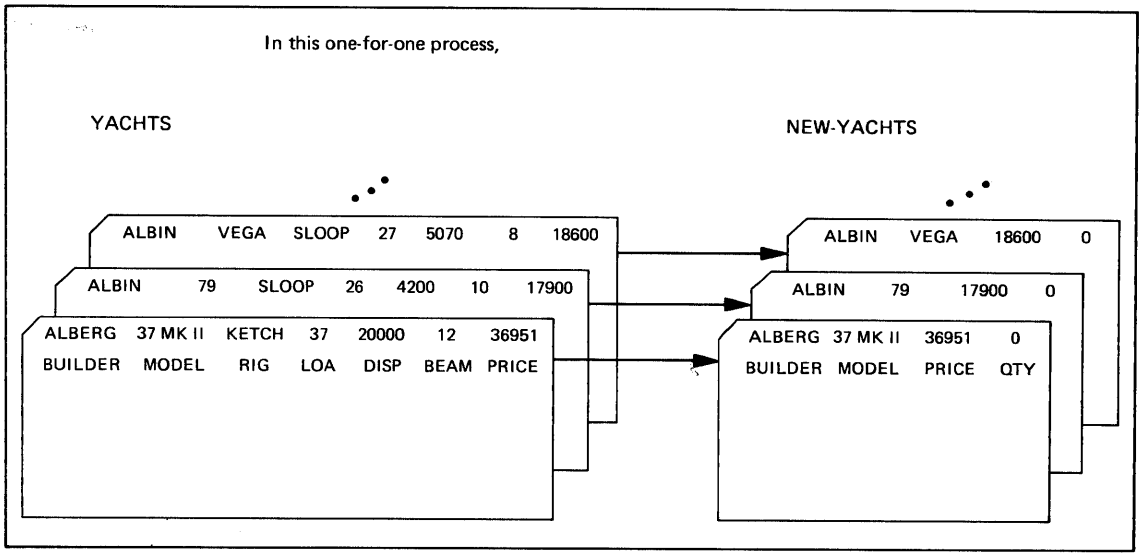


Figure 7-3 Illustrating a Simple Case of Restructuring

7.4 RESTRICTIONS

Because DATATRIEVE-11 runs only in the Support Environment, you cannot update any files that are in use in the Transaction Processor Environment.

APPENDIX A

SAMPLE DATATRIEVE-11 SESSION

The following session was run under a TRAX Operating System.

```
DCL> RUN DTR
12:25:33
Datatrieve, DEC Query and Report System
Version: V01.00, 22-NOV-77
Type HELP for help
DTR>!SAMPLE DATATRIEVE SESSION
DTR>!
DTR>DEFINE RECORD BOOK USING
DTR> 01 BOOK-TITLE.
DTR>     03  TITLE PIC X(22).
DTR>     03  FIRST-AUTHOR.
DTR>         06 LAST-NAME-AUTH1 PIC X(15).
DTR>         06 FIRST-NAME-AUTH1 PIC X.
DTR>     03  PUBLISHER PIC X(15).
DTR>     03  CATEGORY PIC 99.
DTR>     03  PRINT-DATE PIC 99.
DTR>     03  LOCATION PIC X.
DTR>     03  LOANED-TO PIC X(10).
DTR>     03  TYPE PIC X.
DTR>;
DTR>!
DTR>SHOW BOOK;
RECORD BOOK
 USING
01 BOOK-TITLE.
   03  TITLE PIC X(22).
   03  FIRST-AUTHOR.
       06 LAST-NAME-AUTH1 PIC X(15).
       06 FIRST-NAME-AUTH1 PIC X.
   03  PUBLISHER PIC X(15).
   03  CATEGORY PIC 99.
   03  PRINT-DATE PIC 99.
   03  LOCATION PIC X.
   03  LOANED-TO PIC X(10).
   03  TYPE PIC X.
;
DTR>!
DTR>!DEFINING THE DOMAIN
DTR>!
DTR>DEFINE DOMAIN BOOKS USING BOOK ON BOOK.DAT;
DTR>!
DTR>SHOW BOOKS;
DOMAIN BOOKS
 USING BOOK ON BOOK.DAT;
```

SAMPLE DATATRIEVE-11 SESSION

```
DTR>!ADD FOUR NEW BOOK RECORDS
DTR>!
DTR>READY BOOKS WRITE;
DTR>REPEAT 4 STORE BOOKS;
Please supply value for TITLE: SUPERMONEY
Please supply value for LAST-NAME-AUTH1: SMITH
Please supply value for FIRST-NAME-AUTH1: A
Please supply value for PUBLISHER: RANDOM HOUSE
Please supply value for CATEGORY: 15
Please supply value for PRINT-DATE: 72
Please supply value for LOCATION: I
Please supply value for LOANED-TO:
Please supply value for TYPE: H
Please supply value for TITLE: CURTAIN
Please supply value for LAST-NAME-AUTH1: CHRISTIE
Please supply value for FIRST-NAME-AUTH1: A
Please supply value for PUBLISHER: POCKET BOOKS
Please supply value for CATEGORY: 13
Please supply value for PRINT-DATE: 76
Please supply value for LOCATION: I
Please supply value for LOANED-TO:
Please supply value for TYPE: P
Please supply value for TITLE: THE ART OF LOVING
Please supply value for LAST-NAME-AUTH1: FROMM
Please supply value for FIRST-NAME-AUTH1: E
Please supply value for PUBLISHER: HARPER & ROW
Please supply value for CATEGORY: 11
Please supply value for PRINT-DATE: 62
Please supply value for LOCATION: I
Please supply value for LOANED-TO:
Please supply value for TYPE: P
Please supply value for TITLE: HEART OF DARKNESS
Please supply value for LAST-NAME-AUTH1: CONRAD
Please supply value for FIRST-NAME-AUTH1: J
Please supply value for PUBLISHER: W.W. NORTON
Please supply value for CATEGORY: 99
Please supply value for PRINT-DATE: 63
Please supply value for LOCATION: I
Please supply value for LOANED-TO:
Please supply value for TYPE: P
DTR>!
DTR>FIND COMPSCI IN BOOKS WITH CATEGORY EQUAL 09;
[3 records found]
DTR>SELECT 1
DTR>PRINT SKIP 2,"THESE BOOKS ARE IN THE COMPUTER SCIENCE CATEGORY",SKIP 2 THEN FOR
DTR>COMPSCI PRINT TITLE, FIRST-AUTHOR,PUBLISHER,PRINT-DATE,-
DTR>LOCATION("LOC"/"I=IN"/"O=OUT"),TYPE("TYPE"/"P=PAPER"/"H=HARD")


THESE BOOKS ARE IN THE COMPUTER SCIENCE CATEGORY

                              LAST        FIRST                           LOC    TYPE
                              NAME        NAME                     PRINT I=IN  P=PAPER
        TITLE                 AUTH1       AUTH1     PUBLISHER      DATE  O=OUT H=HARD

DIGITAL LOGIC CIRCUITS LIBES                S    HAYDEN BOOK CO.  75     O       P
INTRO. TO DATABASE SYS DATE                 C    ADDISON-WESLEY   76     I       H
BASIC DIGITAL ELECTRON RYAN                 R    TAB BOOKS        75     O       P

DTR>!
DTR>!DEFINE A REPORT WRITER SEQUENCE THAT WILL IDENTIFY BOOKS ON LOAN
DTR>!
DTR>DEFINE PROCEDURE ON-LOAN-REPORT
DTR>REPORT BOOKS WITH LOCATION EQ "O" SORTED BY LOANED-TO, LAST-NAME-AUTH1
DTR>SET REPORT-NAME = "BOOKS ON LOAN"
DTR>SET COLUMNS-PAGE = 80
DTR>PRINT LOANED-TO ("BORROWED"/"BY"),LAST-NAME-AUTH1,TITLE,CATEGORY,TYPE;
DTR>REPORT END;
DTR>END-PROCEDURE;
```

SAMPLE DATATRIEVE-11 SESSION

```
DTR>!DEFINE A REPORT WRITER SEQUENCE TO LIST ALL THE PAPERBACKS
DTR>!
DTR>DEFINE PROCEDURE PAPERBACK-REPORT
DTR>REPORT BOOKS WITH TYPE = "P" SORTED BY ASC FIRST-AUTHOR;
DTR>SET REPORT-NAME = "PAPERBACKS", COLUMNS-PAGE = 80;
DTR>PRINT LAST-NAME-AUTH1, TITLE, LOCATION, CATEGORY;
DTR>AT BOTTOM OF REPORT PRINT SKIP 2,COL 25,"TOTAL PAPERBACKS:",COUNT,SKIP 2,-
DTR>COL 25,"AVERAGE AGE OF BOOKS: ",(77 - AVERAGE PRINT-DATE)
DTR>REPORT END
DTR>END-PROCEDURE;
DTR>!
DTR>!DEFINE A SECOND RECORD TYPE THAT WILL BE USED TO DECODE EACH
DTR>!CATEGORY NUMBER
DTR>!
DTR>DEFINE RECORD CAT-DECODER USING
DTR> 01  BOOK-TITLE.
DTR>     03 TITLE PIC X(22).
DTR>     03 CATGRY PIC X(15).
DTR> ;
DTR>!
DTR>DEFINE DOMAIN CAT-DECODE USING CAT-DECODER ON CATGRY.DAT;
DTR>!
DTR>!DEFINE A PROCEDURE TO TRANSLATE THE NUMERICALLY CODED CATEGORIES TO
DTR>!ENGLISH EQUIVALENT
DTR>!
DTR>!
DTR>DEFINE PROCEDURE TRANSLATE-CATEGORY
DTR>FOR BOOKS
DTR>IF CATEGORY=05 THEN STORE CAT-DECODE BEGIN TITLE=TITLE;CATGRY = "ENGL";END ELSE
DTR>IF CATEGORY=09 THEN STORE CAT-DECODE BEGIN TITLE=TITLE;CATGRY = "CMP SC";END ELSE
DTR>IF CATEGORY=11 THEN STORE CAT-DECODE BEGIN TITLE=TITLE;CATGRY = "PSYCH";END ELSE
DTR>IF CATEGORY=13 THEN STORE CAT-DECODE BEGIN TITLE=TITLE;CATGRY = "FICT";END ELSE
DTR>IF CATEGORY=15 THEN STORE CAT-DECODE BEGIN TITLE=TITLE;CATGRY = "ECONOM";END ELSE
DTR>ABORT "CAN'T DECODE THIS CATEGORY -- "|CATEGORY
DTR>END-PROCEDURE;
DTR>!
DTR>!PRINT THE PAPERBACK AND ON-LOAN-REPORTS
DTR>!
DTR>READY BOOKS WRITE
DTR>FIND BOOKS
[12 records found]
DTR>SELECT 1
DTR>:PAPERBACK-REPORT;
```

PAPERBACKS 2-DEC-77
PAGE 1

LAST NAME AUTH1	TITLE	LOCATION	CATEGORY
CHRISTIE	CURTAIN	I	13
CONRAD	HEART OF DARKNESS	I	99
FROMM	THE ART OF LOVING	I	11
ILG	CHILD BEHAVIOR	O	11
LIBES	DIGITAL LOGIC CIRCUITS	O	09
RYAN	BASIC DIGITAL ELECTRON	O	09
SHEEHY	PASSAGES	O	11

TOTAL PAPERBACKS:

AVERAGE AGE OF BOOKS:

SAMPLE DATATRIEVE-11 SESSION

```
DTR>:ON-LOAN-REPORT

                                BOOKS ON LOAN                          2-DEC-77
                                                                       PAGE 1


 BORROWED BY          LAST NAME AUTH1          TITLE                  CATEGORY   TYPE

B. BORROWE       HOGINS              THE STRUCTURE OF WRITI          05          H
E. SHOCKER       RYAN                BASIC DIGITAL ELECTRON          09          P
H. KANTWRI       HODGES              HARBRACE COLLEGE HNDBK          05          H
J. WURSTKI       ILG                 CHILD BEHAVIOR                  11          P
N. BLUEAGE       SHEEHY              PASSAGES                        11          P
T. TOOKIT        LIBES               DIGITAL LOGIC CIRCUITS          09          P
DTR>!
DTR>!TRANSLATE EACH CATEGORY AND SAVE IT IN THE CAT-DECODER RECORD
DTR>!
DTR>READY CAT-DECODE WRITE
DTR>FIND BOOKS
[12 records found]
DTR>!
DTR>:TRANSLATE-CATEGORY
ABORT: CAN'T DECODE THIS CATEGORY -- 99
Execution terminated by "ABORT" statement
DTR>!
DTR>!NOW PRINT THE NEW RECORDS JUST CREATED
DTR>!
DTR>FIND CAT-DECODE;SELECT 1;PRINT ALL OF CAT-DECODE;
[11 records found]


        TITLE              CATGRY

DIGITAL LOGIC CIRCUITS CMP SC
INTRO. TO DATABASE SYS CMP SC
THE STRUCTURE OF WRITI ENGL
ROGET'S THESAURUS      ENGL
CHILD BEHAVIOR         PSYCH
BASIC DIGITAL ELECTRON CMP SC
HARBRACE COLLEGE HNDBK ENGL
PASSAGES               PSYCH
SUPERMONEY             ECONOM
CURTAIN                FICT
THE ART OF LOVING      PSYCH

DTR>!BEFORE LEAVING THIS SESSION, DELETE ALL THESE TEMPORARY DEFINITIONS
DTR>!
DTR>DELETE BOOK;
DTR>DELETE BOOKS;
DTR>DELETE ON-LOAN-REPORT;
DTR>DELETE PAPERBACK-REPORT;
DTR>DELETE TRANSLATE-CATEGORY;
DTR>DELETE CAT-DECODER;
DTR>DELETE CAT-DECODE;
DTR>EXIT
>
```

APPENDIX B

SUMMARY OF DATATRIEVE-11 COMMANDS AND STATEMENTS

SUMMARY OF COMMANDS

```
DEFINE DOMAIN domain-name-1 USING record-name-1 ON rms-file-spec-1 ;

DEFINE PROCEDURE procedure-name-1
        .
        .   } DATATRIEVE statements and commands
        .
     END-PROCEDURE;

DEFINE RECORD record-name-1 USING data-def-1 [data-def-2...] ;

DEFINEP resource-nme-1 [(password-str-1)] seq-number,lock-type-1,key-1,
     privilege-str-1

DELETE { domain-name-1     } [(password-str-1)] ;
       { record-name-1     }
       { procedure-name-1  }

DELETEP resource-nme-1 [(password-str-1)] seq-number

ERASE  [ALL  [OF rse]]

EXIT

FIND domain-name-1 [WITH condition]

FIND CURRENT [WITH condition]

FIND record-selection-expression

FINISH [domain-name-1 [,domain-name-2...]]

HELP [ADVANCED] [command-name-1[,command-name-2...]]

MODIFY [ALL] [field-name-1 [,field-name-2...]] [OF rse]

PRINT [ALL] [print-list-1] [OF rse] [ON { file-spec-1      }]
                                       { *.prompt-name-1   }

READY  domain-name-1 [(password-str-1)]  [ SHARED    ]  [ READ   ]
                                         [ PROTECTED ]  [ MODIFY ]
                                         [ EXCLUSIVE ]  [ WRITE  ]
                                                        [ EXTEND ]

RELEASE  collection-name-1  [,collection-name-2...]
```

SUMMARY OF COMMANDS (Cont.)

```
        ┌FIRST        ┐
        │NEXT         │
SELECT  │LAST         │      [collection-name-1]
        └value-exp-1  ┘
```

SHOW show-item-1 [,show-item-2...]

where the show items are chosen from the following list:

```
{ PROCEDURES                              }
{ DOMAINS                                 }
{ COLLECTIONS                             }
{ RECORDS                                 }
{ ALL                                     }
{ CURRENT                                 }
{ READY                                   }
{ procedure-name-1[(password-str-1)]      }
{ domain-name-1[(password-str-2)]         }
{ record-name-1[(password-str-3)]         }
{ collection-name-1                       }
```

SHOWP resource-1 [(password-str-1)]

SORT [collection-name-1] BY sort-key-1 [,sort-key-2...]

where the sort-keys assume the following form:

```
┌ASC[ENDING]  ┐
│DESC[ENDING] │  field-name-1
│INCREASING   │
└DECREASING   ┘
```

STORE domain-name-1 [USING statement-1] [VERIFY USING statement-2]

SUMMARY OF STATEMENTS

field-name-1 = value-exp-1

field-name-1 = field-name-2

ABORT value-exp-1

BEGIN statement-1 [;statement-2...] **END**

DISPLAY value-exp-1

FOR rse-1 statement-1

IF condition THEN statement-1 [**ELSE** statement-2]

REPEAT value-exp-1 statement-1

statement-1 **THEN** statement-2

REPORT WRITER STATEMENTS

```
      {TOP   }    {fld-name-1}
AT    {BOTTOM} OF {PAGE      }  PRINT   summry-item-1 [,summry-item-2...]
                  {REPORT    }
```

(Table 5-2 lists all the summary items.)

```
PRINT detail-item-1 [,detail-item-2...]

REPORT [rse] [ON file-spec-1]

REPORT END

SET parameter-1 [,parameter-2...]
```

where the parameters are chosen from the following list:

```
REPORT-NAME = report-name
MAX-LINES = integer-1
MAX-PAGES = integer-2
NUMBER
NO-NUMBER
DATE = ["string-1"]
NO-DATE
LINES-PAGE = integer-3
COLUMNS-PAGE = integer-4
```

Subexpressions

where the record selection expression (rse) assumes the following form:

```
[ALL    ]                          { CURRENT           }
[FIRST n]  [collectn-name-2 IN]    { collectn-name-3   } [WITH conditn]
                                   { domain-name-1     }

           [SORTED BY key-1[,key-2...]]
```

where each sort-key is in the form:

```
[ASC [ENDING] ]
[DESC [ENDING]]   field-name-1
[INCREASING   ]
[DECREASING   ]
```

SUMMARY OF VALUE EXPRESSIONS

Item Format	Description	Remarks
field-name-1	Contents of this field in the CURRENT record[1] are evaluated and used.	Uses CURRENT record[1] of CURRENT collection, so both must exist and contain a field with this name.
collectn-name-1.field-name-2	Contents of this field in the CURRENT record[1] of the named collection are evaluated and used.	Be sure to supply a period (.) as a connector. The field named must exist in the records of the named collection. The collection must be established and have a CURRENT record[1].
"character-string"	The character string is used as given, unless the context forces it to be treated as a number[2].	Any characters may be used except the carriage return, line feed, or CTRL Z (^Z). To include a quotation mark, use two successive quotation marks.
integer-1	A string of digits interpreted as a decimal number.	
- value-exp-1	The resultant value is the negation of the value of value-exp-1.	Take care when using character-strings for value-exp-1: they are changed into numbers.
(value-exp-2)	Causes the evaluation of the enclosed value expression to occur prior to that of any other value expression with which it appears.	
value-exp-3 { + - * / } value-exp-4	Interpreted as the arithmetic result of the operation on the two values expressed by the value expressions.	The rules of precedence for arithmetic operations hold (unless you use parentheses to force certain operations to occur first): multiplications (*) and divisions (/) precede additions (+) and subtractions (-), working left-to-right.

(continued on next page)

SUMMARY OF VALUE EXPRESSIONS (Cont.)

Item Format	Description	Remarks
value-exp-5 ¦ value-exp-6	Creates a new character string by combining the two character strings given for value-exp-5 and value-exp-6.	Special caution should be used with numbers. The result maintains a left-to-right pattern. For example, "ABC"¦"DEF" produces "ABCDEF", "12"¦"034" produces "12034", but 12¦034 produces 1234.
{ MAX / MIN / AVERAGE / TOTAL } value-exp-7 [OF rse]	Computes[2] your choice of the maximum, minimum, average, or total, respectively of this value expression for all records (in the CURRENT collection, by default). If the OF clause is used, then the computation can occur on your choice of the named collection, the named domain, or the CURRENT collection, and they can be further restricted by the rest of the expression.	
{ *.prompt-name-1 / **.prompt-name-1 }	Permits the specification of the value at execution time. Has special importance in procedures. Prompt-name-1 should be the name you want DATATRIEVE-11 to use to prompt you to supply a value.	DATATRIEVE-11 prints the message PLEASE SUPPLY VALUE FOR prompt-name-1: and waits for your response before it processes the command in which this value expression occurs. (Examples occur in Sections 4.11 and 4.13.2.)
COUNT [OF rse]	Computes the number of records in the object of the record selection expression.	The default is the number of records in the CURRENT collection.

[1]The CURRENT record is the record the collection cursor identifies. The collection cursor can only be moved to a record by the SELECT command, which is discussed in Section 4.6.

[2]Value expressions are treated as numbers in the SELECT command, in the FIRST n clause of the record selection expression, and when subject to arithmetic operators (+, -, *, and /) or computations for maximum, minimum, total, and average. Thus, these are not appropriate places for the alphanumeric character string.

SUMMARY OF BOOLEAN EXPRESSIONS

	Operator Symbol	Format/Example	Meaning
Simple	>	A > B	True if A is greater than B
	GT	A GT B	
	GREATER-THAN	A GREATER-THAN B	
	GE	A GE B	True if A is greater than or equal to B
	GREATER-EQUAL	A GREATER-EQUAL B	
	<	A < B	True if A is less than B
	LT	A LT B	
	LESS-THAN	A LESS-THAN B	
	LE	A LE B	True if A is less than or equal in value to B
	LESS-EQUAL	A LESS-EQUAL B	
	EQ	A EQ B	True if A equals B
	EQUAL	A EQUAL B	
	=	A = B	
		A EQ B,C, [OR] D A EQUAL B,C, [OR] D A = B,C, [OR] D	True if A equals any of the three values, B, C, or D
	NOT-EQUAL	A NOT-EQUAL B	True if A does not equal B
	NE	A NE B	
	BT	A BT B [AND] C	True if the value of A falls in between the range of values of B through C, inclusive
	BETWEEN	A BETWEEN B [AND] C	
Compound	AND	BOOL-A AND BOOL-B **PRICE>10000 AND** (RET) **BUILDER EQ "SEAWORTHY"** (RET)	True if the values of the Boolean expressions BOOL-A and BOOL-B are both true
	NOT	NOT BOOL-A **NOT PRICE>10000** (RET)	True if the value of the Boolean expression BOOL-A is false
	OR	BOOL-A OR BOOL-B **MODEL="43K" OR MODEL="49K"** (RET)	True if either of the Boolean expressions BOOL-A or BOOL-B is true
	( )	(BOOL-A) **(BEAM<12 AND LOA BT 20 AND 35)** (RET)	True if BOOL-A is true. The parentheses force the evaluation of this Boolean first. Note that BOOL-A can be a compound Boolean, and this is a means of grouping Booleans in case the normal precedence rules* would not produce the desired effect.

Note

The character A above represents a field name, while B, C, and D are value expressions (as described in Section 4.5.1).

*Normal precedence rules require evaluation of expressions with NOT first, followed by expressions with AND, followed by expressions with OR.

Print-list Elements

Element	Function	Default
field-name-1 [modifiers]	Specifies a class of data items to be printed from the record. Modifiers are explained below.	All the fields in the record
value-exp-1 [modifiers]	A value to be inserted. See Section 4.5.1. Modifiers are explained below.	None
SPACE [n]	Causes n horizontal spaces between entries on each line.	1 space
TAB [n]	Introduces as many tab characters into the print line as specified by n.	If n is omitted, a single tab is assumed
COL n	Advances across the horizontal line to column n. Note: n must not be less than 1 or more than the page width.	None
SKIP [n]	Moves n blank lines ahead to start a new line. Printing starts in column 1, unless another element modifies the position further.	Single spacing
NEW-PAGE	Begins a new print page. Printing starts in column 1, unless another element modifies the position further.	None
	Modifiers	
("header-1"[/"header-2"...])	Specifies a character string See Section 3.8. to be printed on successive lines over the immediately preceding field-name or value-expression. Specification of a hyphen for the header indicates no header is desired.	Headers are composed of the field-name definitions from the record definitions
USING edit-string	Imposes the characteristics of this edit-string on the preceding field or value expression. Edit-strings must observe COBOL Rules (Appendix E).	Uses the edit-string in the record definition, if any. Otherwise considers the field's PICTURE string.

NOTE

DATATRIEVE-11 restricts the column width to 132 characters. It sets tabs automatically at every eighth column. Thus, if the print position is column 10 and DATATRIEVE encounters a TAB 2 specification, the new position will be column 24.

APPENDIX C

DATATRIEVE-11 KEYWORDS

ABORT	DESC	LESS-EQUAL	READ
ADVANCED	DESCENDING	LESS-THAN	READY
ALL	DISPLAY	LINES-PAGE	RECORD
AND	DOMAIN	LT	RECORDS
ASC	DOMAINS	MAX	RELEASE
ASCENDING	EDIT-STRING	MAX-LINES	REPEAT
AT	ELSE	MAX-PAGES	REPORT
AVERAGE	END	MIN	REPORT-HEADER
BEGIN	END-PROCEDURE	MODIFY	REPORT-NAME
BETWEEN	EQ	NE	SELECT
BOTTOM	EQUAL	NEW-PAGE	SEPARATE
BT	ERASE	NEW-SECTION	SET
BY	EXCLUSIVE	NEXT	SHARED
CHARACTER	EXIT	NO	SHOW
CLOSE	EXTEND	NO-DATE	SHOWP
COL	FILLER	NO-NUMBER	SIGN
COLLECTIONS	FIND	NOT	SKIP
COLUMN	FINISH	NOT-EQUAL	SORT
COLUMN-HEADER	FIRST	NUMBER	SORTED
COLUMNS-PAGE	FOR	OF	SPACE
COMP	GE	ON	STORE
COMP-1	GREATER-EQUAL	OPEN	TAB
COMP-2	GREATER-THAN	OR	THE
COMP-3	GT	PAGE	THEN
COMP-5	HELP	PIC	TOP
COMP-6	IF	PICTURE	TOTAL
COUNT	IN	PRINT	TRAILING
CURRENT	INCREASING	PROCEDURE	UIC
DATE	IS	PROCEDURES	USAGE
DECREASING	JUSTIFY	PROTECTED	USING
DEFINE	LAST	PW	VERIFY
DEFINEP	LE	QUERY-HEADER	WITH
DELETE	LEADING	QUERY-NAME	WRITE
DELETEP			

APPENDIX D

DATATRIEVE-11 MESSAGES (ALPHABETICALLY)

NOTES

Errors marked here by (SPR) should not occur. If you receive one of these messages, use a Software Performance Report to notify DIGITAL. Be sure to include all pertinent output.

A sequence of three hyphens (---) in these messages denotes a substitution will occur in the actual message that will identify troublesome items.

TEXT

"---" IS AN UNKNOWN NAME
"---" IS NEITHER A COLLECTION NOR A READIED DOMAIN
"---" CLAUSE NOT RECOGNIZED
"---" IS AN UNKNOWN SET OPTION
"---" IS NOT A KNOWN DICTIONARY ELEMENT TYPE
"---" IS NOT A PROCEDURE
"---" IS NOT THE NAME OF A READIED DOMAIN
"=" EXPECTED, "---" ENCOUNTERED
"AT ---" IS INNOVATIVE, BUT, ALAS, UNKNOWN
"AT" CONDITION PREVIOUSLY SPECIFIED
"SET ---" IS UNKNOWN
"THEN" EXPECTED, PROCEEDING ANYWAY
$CLOSE FAILED (SPR)
$CONNECT FAILED (SPR)
$CREATE FAILED
$DELETE FAILED (SPR)
$DISCONNECT FAILED (SPR)
$DISPLAY FAILED (SPR)
$FIND FAILED (SPR)
$FREE FAILED (SPR)
$FREE OPERATION ISSUED BUT NO BUCKET WAS LOCKED
$GET FAILED
$INIT NEVER ISSUED
$INITF FAILED (SPR)
$OPEN FAILED
$PUT FAILED (SPR)
$REWIND FAILED (SPR)
$UPDATE FAILED (SPR)
$WAIT ERROR ON WORK FILE
$WAIT FAILED (SPR)
--- IS NEITHER A DOMAIN NOR COLLECTION

```
--- IS NOT A DOMAIN NAME
--- IS NOT A VALID RECORD NAME
ACCESS DENIED TO DICTIONARY RESOURCE "---"
ACCESS TO FILE "---" DENIED BY OPERATING SYSTEM
ARITHMETIC OVERFLOW DURING CONVERSION
ASSIGNMENT TO DATA TYPE NOT IMPLEMENTED YET                          (SPR)
ATTEMPT TO ADD RECORD TO MIDDLE OF SEQUENTIAL FILE
ATTEMPT TO DELETE LAST PRIVILEGE ENTRY
ATTEMPT TO DIVIDE BY ZERO, RETURNING -1 AS VALUE
ATTEMPT TO EXTEND AN AREA CONTAINING AN UNUSED EXTENT
ATTEMPT TO RELEASE BAD BLOCK                                         (SPR)
ATTEMPT TO UPDATE KEY FIELD W/O CHANGE ATTRIBUTE
AUXILIARY OUTPUT FILE CLOSED
BAD "IS" NODE                                                        (SPR)
BAD ALLOCATION QUANTITY
BAD ALN FIELD IN ALLOCATION XAB
BAD AOP FIELD IN ALLOCATION XAB
BAD AREA NUMBER IN LAN FIELD IN XAB
BAD BUFFER ADDRESS                                                   (SPR)
BAD CHANNEL NUMBER (LCH) IN FAB
BAD COD FIELD IN XAB
BAD EDIT CHARACTER                                                   (SPR)
BAD ELEMENT ("---") IN QUERY-HEADER
BAD FILE NAME STRING
BAD IAD FIELD IN ALLOCATION XAB
BAD IAN FIELD IN XAB
BAD ISI FIELD IN TERMINAL RAB
BAD LOC FIELD OF XAB
BAD LOCK TYPE, DICTIONARY DAMAGED                                    (SPR)
BAD NAM FIELD IN FAB
BAD ORG FIELD OF FAB
BAD POS FIELD IN XAB
BAD PRINT COLUMN SPECIFIED IN REPORT LINE
BAD RAB
BAD RAC FIELD IN RAB
BAD RAT FIELD IN RAB
BAD RBF FIELD IN RAB
BAD RECORD FILE ADDRESS IN RAB
BAD RECORD FORMAT ON MAG TAPE
BAD RECORD IN SEQUENTIAL FILE
BAD RFM FIELD IN FAB
BAD ROP FIELD
BAD RRV RECORD IN INDEXED FILE; FILE MAY BE CORRUPT
BAD RSZ FIELD IN RAB DURING $PUT OR $UPDATE
BAD SHR FIELD IN FAB
BAD SIZ FIELD IN XAB
BAD UBF FIELD IN RAB
BAD UIC SPECIFICATION
BAD VALUE FOR COLUMNS-PAGE
BAD VOL FIELD IN XAB
BAD XAB FIELD IN FAB
BEGINNING OF FILE DETECTED DURING $SPACE
BLOCK LENGTH IN FAB OR RAB IS INCORRECT
BUCKET SIZE FIELD EXCEEDS MAXIMUM
BUCKET SIZE FIELD IN XAB EXCEEDS MAXIMUM
BUFFER IN USE                                                        (SPR)
CAN'T ACCESS DICTIONARY FOR WRITE
CAN'T CONNECT SECOND RAB TO SEQUENTIAL FILE
CAN'T ERASE RECORD FROM SEQUENTIAL FILE (DOMAIN "---")
CAN'T OPEN COMMAND FILE
CAN'T READY DICTIONARY; DICTIONARY POSSIBLY CORRUPT
CAN'T STORE RECORD IN RMS RELATIVE FILE (DOMAIN: ---)
CAN'T TAKE MIN, MAX, AVERAGE, OR TOTAL OF ZERO OBJECTS
```

```
CENTRAL STORAGE POOL EXHAUSTED
CLOSE FUNCTION FAILED (MUST BE RSTS/E)
COLLECTION NOT IN SYSTEM COLLECTION LIST                          (SPR)
COLUMN ASSIGNMENT NOT FOUND                                       (SPR)
COLUMNS-PAGE VALUE IS TOO LARGE
COMMAND "---" IS UNKNOWN
COMMAND FILE NESTING LIMIT EXCEEDED
COMMAND FILE SYNTAX ERROR
COMPARISON FOR DATA TYPE NOT YET IMPLEMENTED                      (SPR)
COMPILER STORAGE POOL EXHAUSTED, REQUEST ABANDONED
CONTROL ACCESS NOT ALLOWED
CONVERSION NOT YET IMPLEMENTED                                    (SPR)
COULDN'T FIND HASH TABLE NODE FOR REMOVAL                         (SPR)
DATA TYPE NOT YET IMPLEMENTED FOR SORT                            (SPR)
DEVICE FULL: CAN'T CREATE OR EXTEND FILE
DEVICE IS WRITE LOCKED
DEVICE NAME SYNTAX ERROR IN "---"
DEVICE NOT READY
DEVICE POSITIONING ERROR
DICTIONARY ELEMENT "---" IS ALREADY IN USE
DICTIONARY ELEMENT "---" NOT FOUND
DICTIONARY IS DAMAGED, PLEASE "QUIT" IMMEDIATELY!
DICTIONARY OBJECT IS TOO LARGE
DICTIONARY OPERATION ALREADY IN PROGRESS
DICTIONARY OPERATION IS IN PROGRESS
DICTIONARY RECORDS OUTSTANDING                                    (SPR)
DICTIONARY NAME SYNTAX ERROR IN "---"
DICTIONARY NOT FOUND FOR FILE "---"
DOMAIN --- HAS NOT BEEN DEFINED
DOMAIN --- NOT PROPERLY READIED FOR ERASE
DOMAIN --- NOT PROPERLY READIED FOR MODIFY
DOMAIN --- NOT PROPERLY READIED FOR STORE
DOMAIN NOT FOUND IN SYSTEM DOMAIN LIST                            (SPR)
DUPLICATE FIELD BREAK SPECIFIED FOR ---
DUPLICATE OR INVALID SIGN SPECIFIED
END OF FILE
END OF HEADER STRING NOT FOUND                                    (SPR)
END OF PRIMARY INPUT FILE, SESSION TERMINATING
EOF POSITIONING FAILED                                            (SPR)
ERROR IN FILE PROLOGUE: FILE IS CORRUPT
ERROR ON PRIMARY INPUT FILE
ERROR WHILE READYING FILE PROLOGUE
ERROR WHILE WRITING PROLOGUE
EXECUTION FAILED
EXECUTION TERMINATED BY "ABORT" COMMAND
EXECUTION TERMINATED BY OPERATOR
EXPANDED STRING AREA IN NAM BLOCK TOO SHORT
EXPECTED A NUMBER, ENCOUNTERED "---"
EXPECTED "'S" IN QUALIFIED SET NAME, ENCOUNTERED "---"
EXPECTED ";" AFTER DELETE, ENCOUNTERED "---"
EXPECTED "=" IN SET STATEMENT, ENCOUNTERED "---"
EXPECTED "IS" POINTER NODE                                        (SPR)
EXPECTED "LEADING" OR "TRAILING", ENCOUNTERED "---"
EXPECTED CL BLOCK                                                 (SPR)
EXPECTED CN BLOCK                                                 (SPR)
EXPECTED CN OR ST NODE                                            (SPR)
EXPECTED CO BLOCK                                                 (SPR)
EXPECTED CO OR OD BLOCK                                           (SPR)
EXPECTED COLLECTION NAME, ENCOUNTERED "---"
EXPECTED COMMA OR END OF STATEMENT, ENCOUNTERED "---"
EXPECTED DD BLOCK                                                 (SPR)
EXPECTED END OF DELETEP STATEMENT, ENCOUNTERED "---"
EXPECTED END OF REPORT STATEMENT, ENCOUNTERED "---"
EXPECTED END OF STATEMENT AFTER DOMAIN DEF., ENCOUNTERED "---"
```

EXPECTED END OF STATEMENT AFTER PROC. DEF., ENCOUNTERED "---"	
EXPECTED END OF STATEMENT AFTER RECORD DEF, ENCOUNTERED "---"	
EXPECTED END OF STATEMENT OR "COMMA", ENCOUNTERED "---"	
EXPECTED END OF STATEMENT, ENCOUNTERED "---"	
EXPECTED FD BLOCK	(SPR)
EXPECTED FIELD NAME BLOCK	(SPR)
EXPECTED FIELD NAME, ENCOUNTERED "---"	
EXPECTED FS BLOCK	(SPR)
EXPECTED HT BLOCK	(SPR)
EXPECTED IS OR CN BLOCK	(SPR)
EXPECTED IS PRINT OBJECT	(SPR)
EXPECTED IS, FD, OR CN BLOCK	(SPR)
EXPECTED LK BLOCK, DICTIONARY DAMAGED	(SPR)
EXPECTED LOCK TYPE (PW OR UIC), ENCOUNTERED "---"	
EXPECTED NAME NODE	(SPR)
EXPECTED NN BLOCK	(SPR)
EXPECTED NUMBER IN SET STATEMENT, ENCOUNTERED "---"	
EXPECTED NUMBER, ENCOUNTERED NON-DIGIT	
EXPECTED OCTAL NUMBER, ENCOUNTERED "---"	
EXPECTED PERIOD FOLLOWING FIELD DEFINITION, ENCOUNTERED "---"	
EXPECTED PRIVILEGE LIST, ENCOUNTERED "---"	
EXPECTED PT BLOCK, DICTIONARY DAMAGED	(SPR)
EXPECTED RD BLOCK	(SPR)
EXPECTED READY MODE, ENCOUNTERED "---"	
EXPECTED RELATIONAL OPERATOR (EQ, ETC.), ENCOUNTERED "---"	
EXPECTED RO BLOCK	(SPR)
EXPECTED RS BLOCK	(SPR)
EXPECTED ST BLOCK	(SPR)
EXPECTED STATEMENT, ENCOUNTERED "---"	
EXPECTED STATISTICAL BLOCK	(SPR)
EXPECTED VL BLOCK	(SPR)
FIELD "---" NOT INCLUDED IN SORT ORDER	
FIELD NAME "---" IS AMBIGUOUS	
FILE "---" ALREADY EXISTS	
FILE "---" IN INCOMPATIBLE USE BY ANOTHER USER	
FILE "---" NOT FOUND	
FILE COULD NOT BE SPOOLED, REISSUE COMMAND	
FILE EXPIRATION DATE NOT REACHED	
FILE EXTEND FAILURE	
FILE HEADER CONTAINS BAD DATE/TIME INFORMATION	
FILE NAME SYNTAX ERROR IN "---"	
FILE OPEN FAILED, DOMAIN NOT READIED	
FILE READ ERROR	
FILE TYPE SYNTAX ERROR IN "---"	
FILE WRITE ERROR	
FILES-11 ACP COULD NOT MARK FILE FOR DELETION	
FILES-11 ACP ENTER FUNCTION FAILED	
FILES-11 ACP FIND FUNCTION FAILED	
FILES-11 ACP REMOVE FUNCTION FAILED	
GET-PAGE FAILED	(SPR)
HASH TABLE ENTRY FAILED	(SPR)
HEADER SEGMENT NOT RECOGNIZED	(SPR)
HEADER STRING EXHAUSTED	(SPR)
I/O OPERATION PENDING FOR FILE	
ILLEGAL ASSIGNMENT TO GROUP DATA ITEM	
ILLEGAL COMPUTATION PICTURE ON FIELD "---"	
ILLEGAL DUPLICATE KEY VALUE	
ILLEGAL OPERATION	
ILLEGAL PICTURE ON ---	
ILLEGAL PLACEMENT OF MINUS SIGN, IGNORING SIGN	
ILLEGAL SIGN ON COMPUTATIONAL DATA ITEM	
INCONSISTENT RECORD LENGTH FOR SORT	(SPR)
INCORRECT "REPORT END" STATEMENT	

```
INDEX BUCKET CHECK-TYPE MISMATCH; FILE CORRUPT
INDEX NOT INITIALIZED
INDEX TREE ERROR; INDEXED FILE IS CORRUPT
INTERNAL DATA STRUCTURE IS CORRUPTED
INTERNAL INDEX DESCRIPTOR SPACE EXHAUSTED
INTERNAL RMS-11 ERROR (PLEASE NOTIFY DEC)                          (SPR)
INVALID AREA NUMBER IN DAN FIELD OF XAB
INVALID COLUMN HEADER
INVALID COLUMN HEADER ("---")
INVALID COMBINATION OF NO DUPLICATES AND CHANGE
INVALID DISK ADDRESS                                               (SPR)
INVALID FILE ID
INVALID FILE OPTIONS
INVALID IFI FIELD IN FAB
INVALID KEY OF REFERENCE IN RAB
INVALID OPERATION IN AST
INVALID OVERPUNCHED SIGN, ASSUMING +0 FOR DIGIT
INVALID REAL ADDRESS                                               (SPR)
INVALID REPORT STATEMENT ("---")
INVALID USAGE TYPE "---"
KEY --- DOES NOT MATCH ANY FIELD
KEY BUFFER ADDRESS IS ZERO
KEY SIZE IS ZERO
KEY SIZE ZERO OR NEGATIVE                                          (SPR)
LINE SKIP LIMIT EXCEEDED
LOCK TYPE NOT KNOWN, DICTIONARY DAMAGED                            (SPR)
LOGICAL CHANNEL IS BUSY
MAG TAPE IS NOT ANSI LABELED
MAX NUMBER OF KEYS EXCEEDED
MAXIMUM NUMBER OF HEADER LINES HAS BEEN EXCEEDED
MAXIMUM RECORD SIZE IS ZERO, AND RECORD IS FIXED OR RELATIVE
MISSING PICTURE OR EDIT STRING
MORE TABS SPECIFIED THAN EXIST FOR LINE WIDTH
MULTIPLE DECIMAL POINTS IN "---"
MULTIPLE DETAIL LINES SPECIFIED IN REPORT (LIMIT OF ONE)
NAME TOO LONG FOR DICTIONARY USAGE
NEGATIVE VALUE ASSIGNED TO UNSIGNED DATA ITEM (---)
NESTED DICTIONARY ACCESSES ARE NOT ALLOWED
NESTED DICTIONARY ACCESSES NOT PERMITTED
NO BUFFERS AVAILABLE, REQUEST ABANDONED
NO COLLECTION FOR SELECT
NO CONTEXT FOR ERASE
NO CONTEXT FOR MODIFY
NO CONTEXT FOR PRINT
NO CONTEXT HAS BEEN ESTABLISHED FOR FIELD NAME "---"
NO CURRENT RECORD
NO DATA ITEMS MATCHED FOR GROUP ASSIGNMENT
NO PICTURE SPECIFIED ON ELEMENTARY FIELD "---"
NO PRIMARY KEY SPECIFIED FOR INDEX FILE
NO RECORD SELECTED, PRINTING WHOLE COLLECTION
NO REPORT-NAME SPECIFIED
NO SELECTED RECORD FOR MODIFY
NO SORT ORDER FOR REPORT
NODE TYPE NOT RECOGNIZED                                           (SPR)
NODE TYPE NOT YET IMPLEMENTED                                      (SPR)
NON-DIGIT IN STRING "---", IGNORING CHARACTER(S)
NOT A RECOGNIZED NODE TYPE                                         (SPR)
NOT VALID FAB
NULL STATEMENT LIST                                                (SPR)
NUMERIC VALUE TOO GREAT FOR DATA TYPE (OVERFLOW)
OPEN FOR FILE "---" FAILED
OPERATING SYSTEM COULD NOT ACCESS FILE "---"
OPERATING SYSTEM COULD NOT CREATE FILE "---"
OPERATING SYSTEM ERROR DURING $CLOSE
```

OPERATION NOT DECLARED AT OPEN TIME	
OPERATION NOT SELECTED IN ORG$ MACRO	
OUT OF SEQUENCE $PUT FOR INDEXED FILE	
OVERFLOW DURING MULTIPLICATION	
PAGE LEFT LOCKED	(SPR)
PAGE NOT IN USE	(SPR)
PERMANENT STORAGE POOL EXHAUSTED, REQUEST ABANDONED	(SPR)
PRINT OBJECT TOO LARGE FOR LINE WIDTH	
PRIVATE BUFFER POOL ADDRESS NOT ON DOUBLE WORD BOUNDARY	
PRIVATE BUFFER POOL SIZE NOT MULTIPLE OF 4	
PRIVILEGE SEQUENCE NUMBERS START AT 1, THANK YOU	
PROTECTION TABLE ELEMENT DOESN'T EXIST	
RE-READY FAILED, DOMAIN "---" AUTOMATICALLY FINISHED	
READ ERROR ON FILE HEADER ATTRIBUTES	
READ ERROR ON WORK FILE	
RECORD "---" HAS NOT BEEN DEFINED	
RECORD HAS BEEN DELETED	
RECORD IDENTIFIED IS NON-POSITIVE	
RECORD IDENTIFIED IS OUT OF RANGE	
RECORD INDENTIFIED BY INDEX DOESN'T EXIST	
RECORD NOT IN CORE	(SPR)
RECORD NUMBER OUT OF RANGE FOR COLLECTION	
RECORD STREAM ALREADY ACTIVE	
RECORD TOO BIG	
RECORD TYPE ALREADY IN USE	
RELATIVE RECORD ALREADY EXISTS	
REPEAT COUNT IS NOT REASONABLE	
REPORT FORMATTING CLAUSE OUT OF CONTEXT	
REQUEST FOR BLOCK OF ZERO LENGTH	(SPR)
REQUESTED ACCESS TO "---" DENIED	
REQUESTED COLLECTION NAME "---" IS IN USE	
RETRN CALLED AFTER ERROR	(SPR)
RMS DYNAMIC MEMORY HAS BEEN EXHAUSTED	
RMS WON'T RELEASE BLOCK	(SPR)
RMS WON'T RELEASE BLOCK AFTER $CLOSE	(SPR)
RMS WON'T RETURN BLOCK TO SORT	(SPR)
RO BLOCK IS ALREADY CONNECTED	(SPR)
RUN TIME STORAGE POOL EXHAUSTED, REQUEST ABANDONED	
SEARCHING FOR SEMI-COLON	
SELECTED RECORD HAS BEEN PREVIOUSLY DELETED	
SORT ALREADY IN PROGRESS	(SPR)
SORT DEVICE INPUT ERROR	
SORT DEVICE OUTPUT ERROR	
SORT END OF FILE RECORD ERROR	(SPR)
SORT END OF STRING RECORD ERROR	(SPR)
SORT ERROR, COLLECTION RELEASED (SORRY)	(SPR)
SORT FOUND RECORD LARGER THAN EXPECTED	(SPR)
SORT INPUT OPEN FAILED	
SORT KEY ADDRESS ODD	(SPR)
SORT OUTPUT OPEN FAILED	
SORT RECORD ADDRESS ODD	(SPR)
SORT RECORD EXCEEDS MAXIMUM SIZE	(SPR)
SORT RECORD SIZE NOT POSITIVE	(SPR)
SORT SCRATCH RECORD TOO LARGE	(SPR)
SORT SUBROUTINES CALLED OUT OF ORDER	(SPR)
SORT WORK SPACE EXHAUSTED	
SPACE EXHAUSTED IN DICTIONARY ELEMENT	
SPLIT KEYS ARE NOT SUPPORTED, KEY IGNORED	
SPOOLER QUEUE OPERATION FAILED	(SPR)
SPOOLER REQUEST OPERATION FAILED, DON'T WORRY	
STACK SPACE EXHAUSTED	
STACK SPACE EXHAUSTED DURING ASYNCH OPERATION	
STATEMENT NOT IMPLEMENTED YET	(SPR)
SYMBOL "---" IS ALREADY IN USE; DOMAIN --- CANNOT BE READIED	

```
SYNTAX ERROR DURING DOMAIN DEFINITION
SYNTAX ERROR DURING RECORD DEFINITION
SYNTAX ERROR IN CREATEP COMMAND
SYNTAX ERROR IN FILE VERSION NUMBER
SYNTAX ERROR IN PASSWORD SPECIFICATION
SYSTEM DIRECTIVE ERROR
TARGET BUCKET LOCKED
TOO FEW SCRATCH FILES FOR SORT                                    (SPR)
TOO MANY ITEMS FOR SINGLE REPORT LINE
TOO MANY SCRATCH FILES FOR SORT                                   (SPR)
TRUNCATION DURING ASSIGNMENT
UNMATCHED PARENTHESIS IN ARITHMETIC EXPRESSION
UNMATCHED PARENTHESIS IN BOOLEAN EXPRESSION
UNMATCHED PARENTHESIS IN PICTURE STRING
UNRECOGNIZED NAME "---"
UNRECOGNIZED OPERATOR                                             (SPR)
UNRECOGNIZED OR UNIMPLEMENTED STATEMENT TYPE                      (SPR)
UNTERMINATED QUOTED STRING
VALUE TOO LARGE
WORK FILE $CONNECT FAILED                                         (SPR)
WORK FILE $WAIT FAILED                                            (SPR)
WORK FILE CREATE FAILED                                           (SPR)
WORK FILE PAGE NUMBER OUT OF RANGE                                (SPR)
WORK FILE SPACE EXHAUSTED
WORK FILE WRITE FAILED
WRITE ERROR ON FILE HEADER ATTRIBUTES
XAB'S NOT IN PROPER ORDER
[--- RECORDS FOUND]
[1 RECORD FOUND]
[LOOKING FOR "="]
[LOOKING FOR "BY" OR SORT LIST]
[LOOKING FOR "FIRST", DOMAIN NAME, OR COLLECTION NAME]
[LOOKING FOR "SET" OPTION]
[LOOKING FOR "SET" VALUE]
[LOOKING FOR A MATCHING PARENTHESIS]
[LOOKING FOR A VALUE EXPRESSION]
[LOOKING FOR ASSIGNMENT STATEMENT(S)]
[LOOKING FOR BOOLEAN EXPRESSION]
[LOOKING FOR COLLECTION OR DOMAIN NAME]
[LOOKING FOR COLUMN HEADER]
[LOOKING FOR CONSTANT]
[LOOKING FOR DOMAIN NAME]
[LOOKING FOR EDIT-STRING]
[LOOKING FOR FIELD NAME]
[LOOKING FOR FILE NAME FOR REPORT]
[LOOKING FOR FILE NAME]
[LOOKING FOR HEADER SEGMENT]
[LOOKING FOR MATCHING PARENTHESIS]
[LOOKING FOR NEXT ELEMENT IN LIST]
[LOOKING FOR RELATIONAL EXPRESSION]
[LOOKING FOR RELATIONAL OPERATOR (EQ, GT, ETC.)]
[LOOKING FOR SORT LIST]
[LOOKING FOR STATEMENT OR "END"]
[LOOKING FOR STATEMENT]
[LOOKING FOR UPPER VALUE OF BETWEEN]
[SYNTAX ERROR--FLUSHING INPUT]
```

APPENDIX E

COBOL RULES

DATATRIEVE-11 uses a subset of COBOL rules in defining records, PICTURE character strings, and editing strings. The relevant rules have been extracted from the **PDP-11 COBOL Language Reference Manual**, and are summarized below with some tutorial material. For additional assistance, refer to the **PDP-11 COBOL Language Reference Manual**.

The general format of a data-item definition in the record definition is given below. It consists of a number of elements that are covered in this Appendix.

```
data-item-name
     level-no  field-name-1 [PIC clause]
     [;]   [EDIT-STRING-clause]
     [;]   [USAGE-clause]
     [;]   [SIGN-clause].
```

PICTURE CLAUSE

The PICTURE clause defines the format of data as it will exist in the file. If no overriding edit-string clauses are specified, in either the record definition or PRINT command, then the PICTURE clause also controls the output format of the data.

The PICTURE clause assumes the following format:

PIC IS char-string

The char-string is the part of the PICTURE clause that describes the number and type of characters expected in the data. The char-string is restricted to the symbols described in Table E-1. A number of examples follow:

Example	Field May Consist Of:
PRICE PIC 99999	-- 5 digits
LENGTH-OVER-ALL PIC XXX	-- 3 alphabetic or numeric characters
MANUFACTURER PIC X(10)	--10 alphabetic or numeric characters
PRICE PIC S9(5)V99	--7 digits;2 are tenths and hundredths

Table E-1
PICTURE Character-String Symbols

Symbol	Meaning
9	Only digits (numbers 0 through 9) are permitted.
X	Alphanumeric characters are permitted.
S	The number carries a sign, but not necessarily in this location.
V	Assumed decimal point location.
(a number)	Repeat the preceding symbol this many times.

USAGE CLAUSE

The USAGE clause applies to numerical data fields and defines the numerical data representation. The USAGE clause assumes the following format:

USAGE IS usage-string

The usage-string may be any of the computational notations given in Table E-2.

Table E-2
USAGE Specifications

Specification	Meaning
COMP	BASIC-PLUS-2 integer
COMP-1	short form floating point
COMP-2	long form floating point
COMP-6	COBOL binary

Note that COMP-1 and COMP-2 are used in BASIC-PLUS-2 programs to represent single- and double-precision real numbers, respectively.

SIGN CLAUSE

The SIGN clause expresses the location of the sign in a numeric field.

SIGN IS { **LEADING** / **TRAILING** } [**SEPARATE**]

The permitted sign qualifiers are summarized in Table E-3.

Table E-3
Sign Qualifiers

Item	Meaning
LEADING	The sign character is leftmost in the data field
TRAILING	The sign character is rightmost in the data field
SEPARATE	The sign character is a separate character in the number, and must be in either a leading or trailing position, as specified.

Note that if the SIGN clause is not specified and the data field is numeric, it is assumed by default that the sign is TRAILING NON-SEPARATE. In other words, it is assumed the sign is represented by an "overpunch" in the rightmost digit.

EDIT-STRING CLAUSE

The EDIT-STRING clause, if present, specifies the format to be used to print a particular field. If the EDIT-STRING clause is not specified, the PIC clause is used to provide the printing format.

The edit-string in the record definition assumes the following format:

EDIT-STRING IS edit-string

Edit-strings are also allowed in the PRINT commands, as part of the USING EDIT-STRING clause. The edit-string may consist of any of the characters in Table E-4.

Table E-4
Edit-String Symbols

Item	Replacement
9	a digit
Z	a blank character (if digit is 0) or a significant digit
.	a decimal point
-	a blank character if sign is positive or a minus sign (-) if sign is negative
+	a minus sign if sign is negative or a plus sign (+) if sign is positive
0	zero (0)
B	a blank character (space)
/	a slash character (/) -- used in dates
*	an asterisk (*) if zero, or else a significant digit
,	a comma (,) if preceded by high-order significant digits, otherwise a blank character (space)
$	a floating dollar sign--immediately precedes the first significant digit
(a number)	repeat the immediately preceding symbol this many times
CR	two blank characters if data is positive or CR if negative
DB	two blank characters if data is positive or DB if negative

A

B

C

D

E

F

G

H

I

K

L

M

N

O

P

Q

R

S

T

U

V

W

READER'S COMMENTS

NOTE: This form is for document comments only. DIGITAL will use comments submitted on this form at the company's discretion. Problems with software should be reported on a Software Performance Report (SPR) form. If you require a written reply and are eligible to receive one under SPR service, submit your comments on an SPR form.

Did you find errors in this manual? If so, specify by page.

Did you find this manual understandable, usable, and well-organized? Please make suggestions for improvement.

Is there sufficient documentation on associated system programs required for use of the software described in this manual? If not, what material is missing and where should it be placed?

Please indicate the type of user/reader that you most nearly represent.

- ☐ Assembly language programmer
- ☐ Higher-level language programmer
- ☐ Occasional programmer (experienced)
- ☐ User with little programming experience
- ☐ Student programmer
- ☐ Non-programmer interested in computer concepts and capabilities

Name _____ Date _____

Organization _____

Street _____

City _____ State _____ Zip Code or Country _____

---------- Fold Here ----------

---------- Do Not Tear - Fold Here and Staple ----------

FIRST CLASS
PERMIT NO. 33
MAYNARD, MASS.

BUSINESS REPLY MAIL
NO POSTAGE STAMP NECESSARY IF MAILED IN THE UNITED STATES

Postage will be paid by:

Software Documentation
146 Main Street ML5-5/E39
Maynard, Massachusetts 01754