CONTENTS

Page

Page

SECTION 1
INTRODUCTION

The requirements of modern business include the ability to access large volumes of information, to locate particular pieces of information, and to organize, summarize, and update such information. A system providing these capabilities is called an information retrieval system. Tymshare RETRIEVE is a general-purpose information retrieval system providing all of these capabilities.

RETRIEVE has unlimited applications in both the business and technical worlds. For example, RETRIEVE can be used to maintain data bases of accounting data, personnel files, inventories, sales information, mailing lists, production control statistics, and stock transactions.

RETRIEVE is designed to access a uniformly formatted data base, designed by the user to suit his individual needs. Once the user has described a data base structure to RETRIEVE, it is automatically stored on a file, allowing subsequent access of that data base without redescribing the structure. All information entered into the data base is also stored automatically on a file which may be accessed subsequently. In fact, RETRIEVE remembers all modifications that the user makes during a RETRIEVE session and updates the data base file automatically.

RETRIEVE contains many outstanding features enabling the user to:

- Use a small set of powerful commands requiring no knowledge of programming.
- Create a data base directly from RETRIEVE or from a file that has been created on paper tape, in Tymshare's EDITOR, or with a program written in another language.
- Update a data base by changing selected items, deleting entire records, or adding new records.
- Sort a data base by several items.
- Generate from a data base individually designed reports, which may include column headings, totals, subtotals, and any values which can be computed from the items in a data base record. RETRIEVE remembers the report format and can repeatedly use it to generate reports without the user's reentering the report description.
- Merge two or more data bases to create a new data base.
- Create, access, and manipulate binary and scrambled data bases. Binary data bases allow extremely rapid processing; scrambled data bases provide information security.
- Store a set of RETRIEVE commands to be executed repeatedly on a command file, which can then be executed with one simple command.
- Allow the selection of records satisfying any given condition and the calculation of totals, averages, or mathematical expressions involving items in the data base.

ABOUT THIS MANUAL

Section 2 introduces the basic concepts and definitions necessary for using RETRIEVE. The section begins with an example using a few simple commands to introduce RETRIEVE, and then discusses the features introduced in detail. By first studying this section, the user can learn the basic information necessary to use RETRIEVE.

The next three sections discuss all of the RETRIEVE commands for creating and accessing data bases, retrieving information stored on a data base, and updating the data base.

Report generation using the powerful REPORT command is discussed in Section 6; Section 7 deals with command files.

Various features which facilitate easier use of RETRIEVE are presented in Section 8. Section 9 contains some sample RETRIEVE sessions executed on the Tymshare system. Section 10 is a RETRIEVE command summary.

SYMBOL CONVENTIONS

In all examples in this manual, everything typed by the user is underlined. The symbols for user-typed Carriage Return, Line Feed, and Alt Mode/Escape are:

Carriage Return: ↵

Line Feed: ↴

Alt Mode/Escape: ⊕

Control characters are denoted by a superscript c. For example, A^c denotes Control A. The method of typing a control character depends on the type of terminal being used. Consult the literature for your particular terminal.

When a general form of a RETRIEVE command is given in this manual, square brackets are used to denote optional parts of the command. For example,

APPEND [FROM *file name*]

means that the command has the two forms

APPEND

and

APPEND FROM *file name*

SECTION 2
BASIC CONCEPTS OF RETRIEVE

AN EXAMPLE

As an introduction to the basic concepts of RETRIEVE, a simple example is shown below. The user wants to use RETRIEVE to maintain personnel files so that he can quickly and accurately generate payroll information. The following example illustrates some of the features of RETRIEVE that he could use to do this. These features are explained in detail in the remainder of this section.

```
-RETRIEVE↵
```

RETRIEVE is called from the EXECUTIVE.

```
.CREATE↵
DATA BASE: PERSONNEL↵
NEW BASE, OK? YES↵
```

The CREATE command is used to create a new data base named PERSONNEL.

```
PLEASE TYPE IN THE STRUCTURE OF YOUR DATA BASE

FIELD  NAME,WIDTH,TYPE,DECIMAL PLACES

    1  EMPLOYEE,20,C↵
    2  SOC.SEC,11,C↵
    3  SALARY,6,N,2↵
    4  HRS,3,I↵
    5  PAY,7,N,2↵
    6  ↵
```

The user defines the structure of his data base. The first two items are character items, indicated by C. The third item is numeric, N; the fourth item is integer, I. The maximum width is established for each item.

```
EMPLOYEE              SOC.SEC      SALARY HRS PAY

ANDREWS KARL,469-20-9531,2.35,40,0↵
BRADFORD SUSAN,202-46-9277,4.90,40,0↵
FRENCH MARK,519-45-6218,7.20,40,0↵
NELSON DONALD,311,--61-2629,5.10,40,0↵
PALMER DAVID,357-48-3158,410,-1,0↵
PARKER MARY,351-04-8260,4.10,40,0↵
RODRIGUES MARIA,373-75-7302,198.70,-1,0↵
WINTON JOHN,421-98-7244,4.25,40,0↵
↵
```

He enters his data.

Control A, which echos as a back arrow, is used to correct an error.

This Carriage Return terminates data entry. RETRIEVE automatically stores the eight records entered on a file named PERSONNEL in the user's directory. The structure description is also saved automatically on a file named PERSONNEL'STR.x', where x is the letter of the current version of RETRIEVE.

```
8 RECORDS
```

```
•LIST↵
```

LIST prints the entire data base, including headings and record numbers. The record numbers are assigned automatically by RETRIEVE.

```
 RECNO  EMPLOYEE              SOC.SEC       SALARY  HRS      PAY

     1  ANDREWS KARL          469-20-9531     2.35   40      .00
     2  BRADFORD SUSAN        202-46-9277     4.90   40      .00
     3  FRENCH MARK           519-45-6218     7.20   40      .00
     4  NELSON DONALD         311-61-2629     5.10   40      .00
     5  PALMER DAVID          357-48-3158   410.00   -1      .00
     6  PARKER MARY           351-04-8260     4.10   40      .00
     7  RODRIGUES MARIA       373-75-7302   198.70   -1      .00
     8  WINTON JOHN           421-98-7244     4.25   40      .00

8 RECORDS
```

After each command, RETRIEVE prints the number of records affected.

```
•1,4 CHANGE HRS↵
```

The value of the field HRS is changed in records 1 and 4.

```
HRS : HRS

 40 : 48↵
 40 : 44↵

2 RECORDS
```

```
•CHANGE FOR 'WINTON' IN EMPLOYEE↵
```

The CHANGE command is used to modify an entire record.

```
EMPLOYEE              SOC.SEC       SALARY  HRS      PAY

WINTON JOHN           421-98-7244     4.25   40      .00
WINTON JOHN,421-98-7243,4.90,48,0↵

1 RECORDS
```

```
•DELETE FOR EMPLOYEE = 'FRENCH MARK'↵
```

A record is deleted.

```
1 RECORDS

PERSONNEL'OLD' CONTAINS A COPY OF YOUR OLD DATA BASE.
SHALL WE RETAIN IT? YES↵
```

```
•REPLACE PAY WITH SALARY FOR HRS=-1↵
```

The REPLACE command computes PAY for salaried employees, identified by –1 in the field HRS.

```
2 RECORDS

•REPLACE PAY WITH SALARY*HRS FOR HRS#-1↵
```

A new value of PAY is computed for hourly employees.

```
5 RECORDS

•PRINT EMPLOYEE, PAY↵
```

PRINT is used to list the items EMPLOYEE and PAY. Note that headings are printed.

```
EMPLOYEE                PAY

ANDREWS KARL          112.80
BRADFORD SUSAN        196.00
NELSON DONALD         224.40
PALMER DAVID          410.00
PARKER MARY           164.00
RODRIGUES MARIA       198.70
WINTON JOHN           235.20

7 RECORDS

•SUM PAY↵
```

The SUM command prints the total of all values of PAY.

```
        SUM   OF

    1541.10   PAY

7 RECORDS

•QUIT↵
```

QUIT returns the user to the EXECUTIVE. Note that the user did not need to save his data base. RETRIEVE automatically updates the data base file after any command which changes the data. Thus, all changes made are now stored on the file PERSONNEL.

```
-
```

THE RETRIEVE DATA BASE

RETRIEVE is a general-purpose information retrieval system allowing the user to store in one place a large volume of related data, or information, and to access and update this information as required. The information accessed and stored by such a system is called the *data base.* In our example, the data base consists of personnel information.

A data base is usually divided into *records;* each record consists of one or more *fields.* In the sample data base on page 3, each record consists of five fields: EMPLOYEE, SOC.SEC, SALARY, HRS, and PAY.

RETRIEVE is designed to process a *uniformly formatted data base.* Uniformly formatted means that each record contains the same number of fields, arranged in the same order. Thus, each record in our example must contain the five fields:

EMPLOYEE SOC.SEC SALARY HRS PAY

arranged in that order.

Designing the Structure of a Data Base

RETRIEVE requires that certain information about the structure of a new data base be supplied before that data base may be entered. Therefore, the user should first design his data base structure, deciding how many fields he wants in each record, and what these fields should be. A maximum of 98 fields in a record is allowed, with a maximum of 72 characters in each field. However, no symbolic record may be longer than 768 characters.[1] (The maximum is 256 words per record on a binary data base.) In our example, the user decided that his data base would consist of five fields: the employee name, social security number, salary, hours, and pay. After defining what the fields should be, each field should be assigned a name, maximum width, type (character or numeric), and, optionally, the number of decimal places.

Field Names

Field names must begin with a letter and can contain only letters, digits, periods, and the character @. They cannot contain spaces. For example,

Valid	Not Valid	
SOC.SEC	SOC. SEC	*Contains a space.*
SALES1	1SALES	*Does not start with a letter.*
ITEM.NO	ITEM#	*Contains the illegal character #.*

A field name may contain as many as 31 characters. The following words may not be used as field names since they have other meanings when used in RETRIEVE commands:

ALL	FOR	NOT	RESULTS
AND	FROM	ON	SCRAMBLED
BINARY	IF	OR	SYMBOLIC
BY	IN	RECNO	WITH

1 – Note that symbolic records longer than 256 characters cannot be listed or edited in EDITOR.

Field Width

The field width is the maximum number of characters which may be stored in the field. Items entered in this field may be less than the specified maximum, but none can be greater than the maximum. The choice of the width must be based on reasonable assumptions about the data to be stored in the field. For example, the field SOC.SEC was assigned a width of 11 characters, since social security numbers always contain nine digits and two hyphens.

The field EMPLOYEE in the data base PERSONNEL created above has a specified field width of 20 characters, but the first entry into this field, ANDREWS KARL, contains only 12 characters. RETRIEVE automatically supplies spaces to fill the field in the data base file. If an entry to the EMPLOYEE field contains 22 characters, the last two are ignored.

A field may be as wide as 72 characters, as long as the limit of 768 characters per record is not exceeded.

Field Types

A field may be of either character or numeric type. As their names imply, numeric fields must contain only numbers, while character fields may contain any combination of numbers, symbols, or letters.

Character fields are specified by the letter C during structure entry. Such fields may contain both numeric and non-numeric characters. However, arithmetic operations can never be performed on character items, even if the items contain only numeric characters.

There are two kinds of numeric fields in RETRIEVE, integer and non-integer. Integer fields may not contain decimal points. Arithmetic operations may be performed on numeric fields.[1]

Integer fields are indicated by the letter I during structure entry. They may be expressed as integer numbers (without decimal points) within the range –8388608 to 8388608 or as scientific notation in E format. For example, the following are acceptable entries in an integer field:

0
369
–10
3E2 *3E2 is equivalent to 300.*

The N field type may contain any number including integers. Values may contain the digits 0 through 9, one decimal point, and a sign (+ or –). In addition, values may be entered in E format. For example, the following are legal N field entries:

6
–3.54
721.2
4E2
7.5782E–3

Any E format number within the range –1E76 to +1E76 can be used in an N field. The general form of the E format is rEn, where r is a non-zero real number, written with or without a decimal point, and n is an integer. Both r and n may be negative. The r is the mantissa; n, the exponent. The exponent represents the power of ten by which the mantissa is to be multiplied. For example,

1 – See *Expressions, Conditions, and the FOR Modifier*, page 17.

5.1E4 = 51000 -3.21E2 = -321

6E-2 = .06 8E0 = 8

An E format number can be used in an I field only if the number is an actual integer. For example, the following E format numbers may be used in an I field:

6.7E3 (6700)

9.14E2 (914)

The numbers 6.14E1 and 7.1E-2 are not permitted in I fields, since they are equivalent to 61.4 and .071, which are not integers.

NOTE: Commas may not be included in any number. For example, the number 31,219,728 must be typed as 31219728.

When an N field is specified, the user optionally may specify the number of decimal places contained in the field. This feature is discussed under *Describing the Data Base*, page 9.

CALLING RETRIEVE

After the user has designed his data base, he is ready to call RETRIEVE and create his data base. RETRIEVE is called from the EXECUTIVE as follows:

-**RETRIEVE**↵

RETRIEVE responds with a period to indicate that it is ready to accept a command. Each RETRIEVE command must be terminated by a Carriage Return. If a command is too long to be entered on a single line, a Line Feed is used to continue the command on the following line. The maximum allowable command length is 256 characters.

THE CREATE COMMAND: DESCRIBING AND CREATING A NEW DATA BASE

The CREATE command allows the user both to describe the structure of a new data base and to enter the actual data records directly from the terminal. It may be used by typing

.**CREATE**↵

RETRIEVE responds by printing

DATA BASE:

The user chooses the name of his data base; in our sample, the name chosen is PERSONNEL. He enters the name, followed by a Carriage Return. In our example, we have

DATA BASE: PERSONNEL↵

The data base name may be any file name allowed by the Tymshare EXECUTIVE, excluding file names with comments.[1] RETRIEVE stores the data base on a file with that name.

If the name typed is a new data base name,[2] RETRIEVE responds by typing

NEW BASE,OK?

This question may be confirmed by typing YES or NO (or a left subset of YES or NO) followed by a Carriage Return. Typing NO aborts the command; typing YES causes RETRIEVE

1 – See the *Tymshare EXECUTIVE Reference Manual* for details about legal file names.

2 – If the name typed is not a new data base name, RETRIEVE responds as described on page 30.

to request the data base structure. After the structure is entered, RETRIEVE requests the data. In our example, the structure and data are entered as below. The entry of the structure and data is described after the example.

```
-RETRIEVE↵

•CREATE↵
DATA BASE: PERSONNEL↵
NEW BASE, OK? YES↵


PLEASE TYPE IN THE STRUCTURE OF YOUR DATA BASE

FIELD  NAME,WIDTH,TYPE,DECIMAL PLACES

     1  EMPLOYEE,20,C↵
     2  SOC.SEC,11,C↵
     3  SALARY,6,N,2↵
     4  HRS,3,I↵
     5  PAY,7,N,2↵
     6  ↵
```

The user enters the field name (EMPLOYEE), width (20), and type (C). These descriptions are separated by commas and terminated by a Carriage Return.

```
EMPLOYEE              SOC.SEC       SALARY HRS PAY

ANDREWS KARL,469-20-9531,2.35,40,0↵
BRADFORD SUSAN,202-46-9277,4.90,40,0↵
FRENCH MARK,519-45-6218,7.20,40,0↵
NELSON DONALD,311,--61-2629,5.10,40,0↵
PALMER DAVID,357-48-3158,410,-1,0↵
PARKER MARY,351-04-8260,4.10,40,0↵
RODRIGUES MARIA,373-75-7302,198.70,-1,0↵
WINTON JOHN,421-98-7244,4.25,40,0↵
↵

8 RECORDS

•
```

RETRIEVE prompts data entry by printing field names.

The user enters his data records. The items are separated by commas and the record terminated by a Carriage Return. The separating commas are not part of the fields.

Eight records were entered.

RETRIEVE prints a period to indicate its readiness to accept another command.

Describing the Data Base

RETRIEVE requests the structure of a data base during CREATE by printing

```
PLEASE TYPE IN THE STRUCTURE OF YOUR DATA BASE

FIELD  NAME,WIDTH,TYPE,DECIMAL PLACES

     1
```

It then waits for the user to describe the first field. The user enters the field name, width, and type, and the number of decimal places desired for non-integer numeric types, in that order. These descriptors may be separated by commas and terminated by a Carriage Return. After the Carriage Return, RETRIEVE prompts with the next field number. To terminate the structure input, a Carriage Return is typed immediately after the field number prompt. In our example, five fields are described as follows:

Field	Description	Meaning
1	EMPLOYEE,20,C	The field named EMPLOYEE has a maximum width of 20 characters; it is of type C (character).
2	SOC.SEC,11,C	Field 2 is named SOC.SEC; it is a character field of width 11.
3	SALARY,6,N,2	This field is named SALARY. It is numeric of width 6 (3 digits, a decimal point, and 2 digits following the decimal point).
4	HRS,3,I	The field named HRS is of width 3; it is of integer type.
5	PAY,7,N,2	The field PAY is a numeric field of width 7 with 2 decimal places.
6	↵	Structure input is terminated with a Carriage Return. Five fields have been described.

RETRIEVE automatically stores the structure information entered on a file in the user's directory. This file is named by appending the extension 'STR.*x*' to the data base name, where *x* is the letter of the current version of RETRIEVE. This file is used by RETRIEVE whenever the data base PERSONNEL is used, so that the user need not define the structure of his data base more than once.

When describing a numeric field, it is not necessary to enter the width and number of decimal places. Instead, the user may elect to use the standard values given in the following table:

Type	Width	Decimal Places
N	13	6
I	9	0

To use the standard values, the user enters only the field name and type. For example, if the user had wanted SALARY to have a standard width of 13 (12 digits and a decimal point) with 6 decimal places, he could have entered the field description as

3 SALARY,N↵

Similarly, to specify a standard width of 13 with 2 decimal places for PAY, he could have entered

5 PAY,N,2↵

Entering the Data Records

After the structure input is terminated during CREATE, RETRIEVE prompts data input by printing the field names in the order specified. It then returns the carriage and waits for the

user to enter his data records. To enter a record, the user types the field entries in the specified order. The items must be separated by commas, and the record terminated by a Carriage Return. In our example, the records are entered as follows:

```
EMPLOYEE                SOC.SEC       SALARY HRS PAY

ANDREWS KARL,469-20-9531,2.35,40,0↵
BRADFORD SUSAN,202-46-9277,4.90,40,0↵
FRENCH MARK,519-45-6218,7.20,40,0↵
NELSON DONALD,311,--61-2629,5.10,40,0↵
PALMER DAVID,357-48-3158,410,-1,0↵
PARKER MARY,351-04-8260,4.10,40,0↵
RODRIGUES MARIA,373-75-7302,198.70,-1,0↵
WINTON JOHN,421-98-7244,4.25,40,0↵
↵
```

Control A deletes a character and, on some terminals, prints a back arrow.

```
8 RECORDS
```

Note that the fields need not be entered in a rigid format, although they must be entered in the order specified in the data base structure. RETRIEVE automatically supplies spaces to fill the specified field width. Character fields are left justified; that is, trailing spaces are supplied to fill the field. Numeric fields are right justified (leading spaces are supplied). Fields in which the number of decimal places has been supplied are aligned by decimal point. Trailing zeros are added, if necessary, to ensure that a number is right justified.

After all records are entered, the user types a Carriage Return alone to terminate the command. RETRIEVE prints the number of records entered and then prints a period to indicate it is ready to accept another command.

RETRIEVE automatically stores the records entered on a file in the user's directory. This file is given the name specified as data base during the CREATE command (PERSONNEL in our example).

Since commas terminate field entries, they can be included in a field only by enclosing the field in single or double quote marks. For example, if the user had wanted to separate the last name of an employee from the first name in our example, he could have entered the records as follows:

'ANDREWS,KARL',469-20-9531,2,35,40,0

or

"ANDREWS,KARL",...

Leading and embedded spaces are counted as part of the field. For example, if the user had typed the first field in the second record as

BRADFORD SUSAN,

16 characters would have been read. The item would begin with two leading spaces. Note that the space embedded between BRADFORD and SUSAN is also counted.

If a record contains more items than the terminal width allows, a Line Feed may be used to continue the record entry on the following line.

Correcting Errors

Errors may be corrected while typing data records by using Control A, Control Q, and Control W. This feature is discussed fully in Section 8. In our example, the user deletes a comma in the fourth record entered by typing a Control A which deletes the preceding character:

NELSON DONALD,311,A^{C}← -61-2629,5.10,40,0↵

On some terminals, when a Control A is typed, RETRIEVE prints a back arrow (←) to indicate its use. The above record is accepted as:

NELSON DONALD,311-61-2629,5.10,40,0

Control Q is used to delete the entire current line being typed.

Control W deletes the previous word in the line being typed.

Error Messages

For the user's convenience, RETRIEVE indicates errors in the data base as it is being entered.

If a field entered contains an error such as a non-numeric character in a numeric field, it and all succeeding fields in the record are ignored. When the record is terminated, RETRIEVE requests the field in error by printing the field name followed by a comma and a space. The user should then retype the fields needed to complete the record.

If the record entered has too few fields, RETRIEVE requests the first field not entered just as in the preceding case.

Example

Suppose the user had typed record 5 of our example as follows:

```
PALMER DAVID, 357-48-3158, $410.00, -1, 0↵
```

He types a non-numeric character, $, in SALARY.

```
CONTENTS OF FIELD "SALARY" IS NOT A NUMBER

SALARY, 410.00, -1↵
PAY, 0↵
```

When he corrects the error, he forgets the last field in the record.

RETRIEVE prompts each time an error occurs. After correcting the errors, the user may continue typing records.

If a record entered has too many fields, the message TOO MANY FIELDS is printed. The record must be reentered.

If a field longer than its maximum width is entered, it is truncated, and a message is printed. Thus, if record 5 had been entered with an embedded space in SOC.SEC, as follows:

```
PALMER DAVID, 357-48- 3158, 410, -1, 0↵

REC.4 FIELD "SOC.SEC" IS TOO LONG. LAST 1 CHARS. IGNORED
```

the record retained in the data base would have been

```
PALMER DAVID          357-48- 315 410.00  -1      .00
```

The social security number can be corrected by using the CHANGE command, described on page 20.

DISPLAYING THE DATA BASE: LIST AND PRINT

Our example illustrates two commands for displaying all or part of the data base: LIST and PRINT.[1] Both of these commands can be used to print either entire records or selected fields of records.

The LIST command may be used to display the entire data base, as in our example:

```
•LIST↵

 RECNO  EMPLOYEE            SOC.SEC       SALARY  HRS     PAY

     1  ANDREWS KARL        469-20-9531     2.35   40     .00
     2  BRADFORD SUSAN      202-46-9277     4.90   40     .00
     3  FRENCH MARK         519-45-6218     7.20   40     .00
     4  NELSON DONALD       311-61-2629     5.10   40     .00
     5  PALMER DAVID        357-48-3158   410.00   -1     .00
     6  PARKER MARY         351-04-8260     4.10   40     .00
     7  RODRIGUES MARIA     373-75-7302   198.70   -1     .00
     8  WINTON JOHN         421-98-7244     4.25   40     .00

 8 RECORDS
```

Notice that record numbers (RECNO) as well as headings are printed by LIST. The record numbers are assigned automatically by RETRIEVE; they are not entered with the data records.[2]

NOTE: If the terminal being used is not wide enough to print the entire data base with the LIST command, the characters on the right overprint or print on the next line.

LIST may also be used to display selected fields in a data base, in the form

LIST *field list*

followed by a Carriage Return, where the field list consists of one or more field names, separated by commas. For example, the employee name, social security number, and salary for all records in our data base could be displayed as follows:

1 – A third command for displaying the data base, FAST, is also available. See *LIST, PRINT, and FAST*, page 49.

2 – See page 15 for further information about record numbers.

```
.LIST EMPLOYEE, SOC.SEC, SALARY↵

 RECNO  EMPLOYEE              SOC.SEC       SALARY

     1  ANDREWS KARL          469-20-9531     2.35
     2  BRADFORD SUSAN        202-46-9277     4.90
     3  FRENCH MARK           519-45-6218     7.20
     4  NELSON DONALD         311-61-2629     5.10
     5  PALMER DAVID          357-48-3158   410.00
     6  PARKER MARY           351-04-8260     4.10
     7  RODRIGUES MARIA       373-75-7302   198.70
     8  WINTON JOHN           421-98-7244     4.25

8 RECORDS
```

The PRINT command is the same as LIST except that it does not automatically print record numbers. Thus, in our example above:

```
.PRINT EMPLOYEE, SOC.SEC, SALARY↵

EMPLOYEE              SOC.SEC       SALARY

ANDREWS KARL          469-20-9531     2.35
BRADFORD SUSAN        202-46-9277     4.90
FRENCH MARK           519-45-6218     7.20
NELSON DONALD         311-61-2629     5.10
PALMER DAVID          357-48-3158   410.00
PARKER MARY           351-04-8260     4.10
RODRIGUES MARIA       373-75-7302   198.70
WINTON JOHN           421-98-7244     4.25

8 RECORDS
```

The command

```
.PRINT↵
```

prints the entire data base with headings but without record numbers.

RECORD NUMBER ADDRESSING

The forms of LIST and PRINT described above process all records in a data base. However, these commands can be modified to process any single record or group of records selected by the user through the use of record number addressing. In fact, most RETRIEVE commands can be used with record number addresses, as shown in the descriptions of each command given later in this manual.

The general form to use for record number addressing is

range list command

where the *range list* can be

1. The record number of a single line, such as 15.
2. A range of lines, consisting of a pair of numbers separated by a colon. For example, 5:10 specifies lines 5 through 10.
3. Any combination of 1 and 2 above, where the record numbers and ranges are separated by commas or spaces. For example,

Range List	Specifies
1,7:20,50:100,200	Records 1, 7 through 20, 50 through 100, and 200.
1,3,20	Records 1, 3, and 20.
5:15 200:250	Records 5 through 15 and records 200 through 250.

Note that record numbers in the range must be given in numerical order. Thus, 3:16 is allowed, but 16:3 is not. However, the ranges themselves need not be in order. For example,

.2,7,3:5 LIST↵

prints records 2, 7, and 3 through 5 in the specified order.

A dollar sign ($) may be used in a range list to represent the last record. For example,

.$ LIST↵

prints the last record in the data base.

The range 90:$ specifies record 90 through the last record.

Examples

```
•5 LIST↵

 RECNO EMPLOYEE              SOC•SEC       SALARY HRS      PAY

     5 PALMER DAVID          357-48-3158 410•00  -1      •00

1 RECORDS
```

```
•3:6 PRINT EMPLOYEE↵

EMPLOYEE

FRENCH MARK
NELSON DONALD
PALMER DAVID
PARKER MARY

4 RECORDS

•1,3:5,7 LIST EMPLOYEE,SOC.SEC,SALARY↵

 RECNO  EMPLOYEE              SOC.SEC       SALARY

     1  ANDREWS KARL          469-20-9531     2.35
     3  FRENCH MARK           519-45-6218     7.20
     4  NELSON DONALD         311-61-2629     5.10
     5  PALMER DAVID          357-48-3158   410.00
     7  RODRIGUES MARIA       373-75-7302   198.70

5 RECORDS
```

Record numbers are automatically assigned by RETRIEVE when the records are entered. After a DELETE, SORT, or MERGE command, these record numbers are changed to reflect the new order. The LIST command always prints the record numbers in the first column, and thus may be used to determine the record number of a particular record.

RECNO is the field name used by RETRIEVE to identify the record number. RECNO is a valid numeric field and may be used in the field list in certain RETRIEVE commands. For example,

```
•FAST EMPLOYEE,RECNO↵

ANDREWS KARL          1
BRADFORD SUSAN        2
FRENCH MARK           3
NELSON DONALD         4
PALMER DAVID          5
PARKER MARY           6
RODRIGUES MARIA       7
WINTON JOHN           8

8 RECORDS

•
```

RECNO cannot be used in the field list in the SORT, MERGE, CHANGE, MODIFY, or REPLACE commands.

EXPRESSIONS; CONDITIONS, AND THE FOR MODIFIER

In addition to record number addressing, command conditions are available for specifying the records on which a command is to operate. These conditions are used together with the FOR modifier, in the form

command **FOR** ***condition***

For example,

```
.LIST FOR HRS > 40↵
```

lists all records for which the field HRS contains a value greater than 40. In this example, the condition is

HRS > 40

In our example, the employee and salary for all salaried employees could be displayed as follows:

```
.PRINT EMPLOYEE, SALARY FOR HRS = -1↵
```

The condition is HRS = -1.

```
EMPLOYEE                SALARY

PALMER DAVID            410.00
RODRIGUES MARIA         198.70

2 RECORDS

.
```

Conditions are composed of combinations of expressions. Thus, to define conditions precisely, we must first define *expressions.* RETRIEVE allows the use of both numeric and character expressions in conditions.

A *numeric expression* can be any of the following:

1. A number in integer or decimal form, such as 3 or –9.2.
2. A numeric field name, such as SALARY in our example.
3. Any meaningful combination of 1 and 2 above using arithmetic operators, such as HRS–40.

RETRIEVE allows the following arithmetic operators:

Operator	Meaning
+	Addition
–	Subtraction (negation or unary minus)
*	Multiplication
/	Division
↑	Exponentiation

Examples of Numeric Expressions

Assuming HRS, SALARY, TOTAL, QTY1, QTY2, PRICE1, and PRICE2 are numeric field names, the following are numeric expressions.

0

HRS

-6 (example of negation or unary minus)

SALARY*40

TOTAL/12

QTY1*PRICE1 + QTY2*PRICE2

2↑3 (means 2^3, i.e., 2 × 2 × 2)

NOTE: No two arithmetic operators may be adjacent, except in the case that the second operator is a unary minus.

A *character expression* can be any of the following:

1. Any string enclosed in single or double quote marks, such as "ANDREWS KARL" or '53–6902'. (A string is simply a sequence of characters.)
2. Any character field name, such as EMPLOYEE in our example.
3. Any meaningful combination of the above with the operator +, used for string concatenation. The operator + used in a character expression signifies string concatenation, that is, joining two strings together. For example, 'SUN' + 'DAY' has the value SUNDAY.

Examples of Character Expressions

Assuming SOC.SEC, CITY, STATE, ZIP, LNAME, and FNAME are the names of character fields, the following are all character expressions:

'560–62–8009'

SOC.SEC

CITY + STATE + ZIP

LNAME+","+FNAME

A *condition* may consist of:

1. Numeric expressions related by any of the relational operators listed below, for example, HRS>40.
2. Character expressions related by any of the relational operators, for example, EMPLOYEE = 'FRENCH MARK'.

The relational operators, which may be used in either short or long form, are:

Short Form	Long Form
<	LESS THAN
>	GREATER THAN
=	EQUAL TO
#	UNEQUAL TO
<=	LESS THAN OR EQUAL TO
>=	GREATER THAN OR EQUAL TO

Any relational operator may be preceded by the operator NOT. For example, the following are equivalent:

<

NOT >=

LESS THAN

NOT GREATER THAN OR EQUAL TO

A condition may also consist of character expressions related by the operator IN. This operator specifies that one string is contained in another. For example, the condition

'WINTON' NOT IN EMPLOYEE

refers to all records which do not contain the string WINTON within the field EMPLOYEE.

More complicated conditions may be formed using the following logical words, listed in descending order of precedence.

Word	Meaning
NOT	NOT A is true if A is false.
AND	A AND B is true only if A and B are both true.
OR	A OR B is true if A or B or both A and B are true (inclusive OR).

Conditions in RETRIEVE are always preceded by the FOR modifier and always have a value of true or false.

Examples

```
FOR JOB.DESC # 'MGE' AND JOB.DESC # 'OFR'
FOR QTY LESS THAN 10 AND PRIORITY.CODE = 'A'
FOR PROFIT > PRICE/10 OR COST < .50 OR ON.HAND > QTY
```

Parentheses may be used in expressions and conditions to specify the order of operations. For example,

SALARY*40 + SALARY*2*(HRS-40)

In evaluating an expression or condition, the following rules govern the order in which operations are performed.

1. All operations within the innermost set of parentheses are performed first, then those within the next set, etc.
2. Without violating rule 1 above, operations are performed in the following order:
   - (unary minus or negation)
   ↑
   * and /
   + and - (subtraction)
   <, <=, >, >=, =, #
   IN
3. Without violating rules 1 and 2 above, operations are performed from left to right.

UPDATING THE DATA BASE

Our example illustrates three commands for updating the data base: CHANGE, DELETE, and REPLACE.

The CHANGE Command

This command allows the user to selectively change the values of the fields or records. It takes the form

[*range list*] CHANGE [*field list*] [FOR *condition*]

where everything in brackets is optional.[1] The *field list*, if specified, consists of the name of the field or fields to be changed, separated by commas. For example,

1,5,16:20 CHANGE QTY FOR PART.NO = 731

If a field list is not specified, CHANGE allows the user to update an entire record or records. When CHANGE is used in this manner, RETRIEVE prints headings as in the PRINT command, and displays the first record to be changed. Then it waits for the user to enter the new record. After the new record is accepted, the next record to be changed, if any, is displayed. RETRIEVE continues until all records specified in the command have been processed.

1 – CHANGE may also be used with the RESULTS TO and ALL TO modifiers. See *Destination Modifiers*, page 61, for details.

In our example, CHANGE was used to update one record as follows:

```
•CHANGE FOR 'WINTON' IN EMPLOYEE↵

EMPLOYEE                SOC.SEC        SALARY HRS      PAY

WINTON JOHN             421-98-7244    4.25   40       .00
WINTON JOHN,421-98-7243,4.90,48,0↵

1 RECORDS
```

The new field values are entered, separated by commas, just as during CREATE.

The command

```
.CHANGE↵
```

allows modification of each record in the data base. The records are printed one at a time in the same format as in the above example. After printing a record, RETRIEVE waits for the user to enter the updated record and then prints a blank line and the next record in the data base.

If a field list is present in CHANGE, a heading containing the specified field names, a colon, and the field names again, is printed. Then the old field values, followed by a colon but not a Carriage Return, are printed for each record to be changed. After printing the colon, RETRIEVE waits for the new field value or values to be entered. Thus, in our example:

```
•1,4 CHANGE HRS↵

HRS : HRS

 40 : 48↵
 40 : 44↵

2 RECORDS
```

If more than one field is to be changed, the new fields entered are separated by commas. To leave a field unchanged, a comma or Carriage Return is typed immediately after a comma or prompt. For example, we could change SALARY and HRS for records 4 through 6 in our example as follows:

```
.4:6 CHANGE SALARY,HRS↵

SALARY HRS : SALARY HRS

  5.10   44 : 5.50,43↵
410.00   -1 : 460,↵          Leaves HRS unchanged.
  4.10   40 : ,47↵           Leaves SALARY unchanged.

3 RECORDS
```

The DELETE Command

This command allows deletion of selected records. It takes the form

[*range list*] DELETE [FOR *condition*]

where everything in brackets is optional.

The unchanged data base is written on a backup file for security. This file may be retained or deleted as the user chooses. For example,

```
.DELETE FOR EMPLOYEE = 'FRENCH MARK'↵

1 RECORDS

PERSONNEL'OLD' CONTAINS A COPY OF YOUR OLD DATA BASE.
SHALL WE RETAIN IT? YES↵
```

deletes record 3 from our sample data base PERSONNEL and creates a backup file PERSONNEL'OLD'. See *Data Base Recovery*, page 95, for information about restoring the data base to its state before the DELETE command was given.

The command

```
.DELETE↵
```

deletes all records in the data base. In this case, a question DELETE ALL RECORDS? is printed, and the user must answer YES or NO, abbreviating his response if desired. If the answer is NO, the user is returned to RETRIEVE command level.

The REPLACE Command

The REPLACE command[1] allows the user to replace selected fields with any desired expressions. It takes the form

[*range list*] REPLACE $field_1$ WITH $expression_1$ [,$field_2$ WITH $expression_2$,...] [FOR *condition*]

where everything in brackets is optional. This command causes $field_1$ to be replaced with the value of $expression_1$, $field_2$ with $expression_2$, etc., for all selected records.

REPLACE is especially useful when the user wishes to change a number of records in the same way, since it is not necessary for the user to enter the changes for each record separately as is the case with CHANGE. For example, the command

```
.REPLACE PAY WITH SALARY FOR HRS=-1↵
```

replaces the value of PAY with the value of SALARY in *all* records in which the HRS field has the value of -1. To accomplish this with CHANGE, the user would have to type

```
.CHANGE PAY FOR HRS=-1↵
```

and then enter the new values of PAY for all specified records separately.

REPLACE also allows the user to avoid performing complicated arithmetic operations manually. Our example illustrates this with the command

```
•REPLACE PAY WITH SALARY*HRS FOR HRS#-1↵

5 RECORDS
```

RETRIEVE thus allows the user to enter a value of zero for PAY in all records and then use the power of the computer to calculate the true value of PAY with the REPLACE command.

When more than one field replacement is specified in a REPLACE command, the replacements are executed from left to right. If an expression involves a previously replaced field, the replaced value is used. For example,

```
•REPLACE SALARY WITH 1.05*SALARY,PAY WITH SALARY FOR HRS=-1↵
```

would give each salaried employee a 5% raise and then replace the value of PAY with the new value of SALARY. But, if the command were given instead as

```
•REPLACE PAY WITH SALARY,SALARY WITH 1.05*SALARY FOR HRS=-1↵
```

the old value of SALARY would replace PAY and the employees would not receive their raise in the current paycheck.

1 – REPLACE may also be used with the RESULTS TO and ALL TO modifiers. See *Destination Modifiers*, page 61, for details.

COMPUTING TOTALS: THE SUM COMMAND

The SUM command computes the totals for any specified numeric fields or expressions. It takes the form

[range list] SUM *[expression list]* [FOR *condition*]

where everything in brackets is optional. The expression list may contain any numeric expressions separated by commas. The command prints the sums of all specified expressions for the selected records.

Examples (for our data base PERSONNEL)

```
•SUM PAY↵
        SUM   OF
    1541.10   PAY
7 RECORDS
```

Prints total weekly pay for all records.

```
•SUM PAY FOR HRS=-1↵
        SUM   OF
     608.70   PAY
2 RECORDS
```

Prints total pay for salaried employees.

```
•SUM HRS,PAY FOR HRS#-1↵
        SUM   OF
        220   HRS
     932.40   PAY
5 RECORDS
```

Prints total hours worked and total pay for hourly employees.

```
•SUM PAY*52↵
        SUM   OF
   80137.2    PAY*52
7 RECORDS
```

Prints yearly payroll, assuming constant weekly pay.

```
•
```

THE QUIT COMMAND

To return to the EXECUTIVE from RETRIEVE, the command

.QUIT↵

is used.

SAVING THE DATA BASE

Note that the user does not need to save any of the changes made to his data base during a RETRIEVE session. RETRIEVE automatically updates the data base file during any command which changes the data base. However, the SAVE command is available for saving the data base; it is useful for saving part of the data base or for creating a binary or scrambled data base file. These features are discussed in Section 3.

SECTION 3
THE DATA BASE

The large volume of data, or information, which can be accessed and updated by an information retrieval system such as RETRIEVE is known as the data base. RETRIEVE is designed to access a uniformly formatted data base, that is, a data base in which each record contains the same number of fields, or items, arranged in the same order. The user may design his data base to suit his individual needs, specifying the desired structure in the manner described in Section 2.[1] Once this structure is specified during either the CREATE or BASE command, RETRIEVE automatically creates a file containing the structure description and allows the user subsequently to access his data base without redescribing the structure.

RETRIEVE allows creation, access, and manipulation of data bases in symbolic, binary, and scrambled form. Data bases may be created by entering data

- directly into RETRIEVE, entering the data at the terminal, or
- from a file created in Tymshare's text editing language, EDITOR, in a program written in another language such as SUPER FORTRAN, or in the EXECUTIVE with the TAPE program or the COPY command.

Once a data base has been created, RETRIEVE automatically stores it on a file (different from the structure file mentioned above) which can subsequently be accessed using the BASE command.

In Section 2 of this manual we discussed the rules for defining and describing the data base structure. We also defined the basic structure specifications which must be supplied by the user: field names, field width, and file type. The CREATE command was discussed in some detail.

In this section, the following RETRIEVE commands which are related to data base creation and access are discussed:

CREATE APPEND BASE *or* LOAD MERGE SAVE	Used for data base creation and access.
STRUCTURE SIZE	Used for obtaining information about the data base.

Binary and scrambled data bases are discussed below. The methods of creating and accessing binary and scrambled data bases are given in the descriptions of the individual commands.

DATA FILES

The commands for data base creation and access require the user to specify the data file to be created or accessed by the command, and also, in some cases, to specify the file type.

1 – See *The RETRIEVE Data Base*, page 6, and *The CREATE Command: Describing and Creating a New Data Base*, page 8, for details about data base structures.

The *file name* specified in a RETRIEVE command may be any valid file name allowed by the Tymshare EXECUTIVE, with the exception of file names with comments. For example, in the command

.CREATE TRANSACTIONS↵

the user specifies that his data base be stored on a file named TRANSACTIONS in his directory.[1]

The *file type* specified may be either SYMBOLIC, BINARY, or SCRAMBLED.

Symbolic Data Bases

Symbolic files contain information in the standard alphanumeric character representation, the same representation in which data appears on the terminal.

When a data base is being created during CREATE, RETRIEVE assumes it is a symbolic file until otherwise specified in the command; therefore, SYMBOLIC need not be specified.

Thus, the command

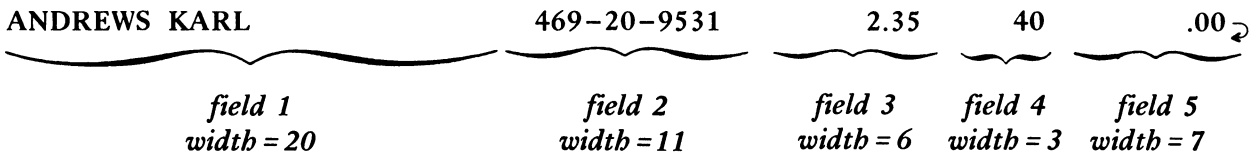
.CREATE↵
DATA BASE: PERSONNEL↵

used in the example in Section 2 of this manual, created a symbolic data file named PERSONNEL in the user's directory.

RETRIEVE stores symbolic data bases in fixed record length form. The length of each record is equal to one more than the sum of the individual field widths; the extra character is a Carriage Return. Each data item within the record is stored in a field of width equal to the maximum specified in the structure description. Blanks are used to fill the field when needed; numbers are right justified and character items are left justified. For example, for the data base PERSONNEL in Section 2, the structure description was entered as:

EMPLOYEE,20,C
SOC.SEC,11,C
SALARY,6,N,2
HRS,3,I
PAY,7,N,2

so the first record in the file PERSONNEL (after termination of CREATE) is:



The record length is 48 (20+11+6+3+7+1).

Binary Data Bases

The information stored on a binary file is written in internal machine code. Binary files have the advantage of being faster to process than symbolic files. However, symbolic files may be more convenient to use than binary files since they can be listed at the terminal from the EXECUTIVE and can be edited with EDITOR, while binary files cannot.

1 – For more information about files, file names, and file directories, see the *Tymshare EXECUTIVE Reference Manual.*

Binary data bases created with RETRIEVE are also in fixed record length form, the record length being the number of internal machine words allocated to each record. RETRIEVE allocates storage space to each field according to the following table:

Field Type	Storage Allocated (1 word = 24 bits)
N (numeric)	2 words
I (integer)	1 word
C (character)	(field width + 2)/3 words rounded down to the next integer.

Note that the field width has no relation to the storage allocated in the case of I and N fields. An I or N field is always assigned one or two words of storage, respectively. The field width specifications for these fields are used for symbolic output only, such as in the PRINT command.

In a C field the value of the storage allocation expression is rounded down to the nearest integer. For example, a field of 1, 2, or 3 characters takes 1 word; a field of length 4, 5, or 6 uses 2 words.

Scrambled Data Bases

RETRIEVE data bases may be scrambled for optimum information security. Scrambled data bases allow the user to encode his data file into a form which cannot be deciphered without a special password which is created by the user. RETRIEVE performs the actual scrambling of the data base. For example, a scrambled data base called SALES may be created as follows:

```
•CREATE SALES SCRAMBLED ↵
NEW BASE, OK? YES ↵

PASSWORD:        ↵          The user types his password, which does not print.
REPEAT FOR CHECK:    ↵      RETRIEVE requests the password again for verification.

PLEASE TYPE IN THE STRUCTURE OF YOUR DATA BASE

FIELD  NAME,WIDTH,TYPE,DECIMAL PLACES

    1                       The user enters the data base structure followed by the data records, as usual.
```

Scrambled data bases created in RETRIEVE are always binary.

Whenever a new scrambled data base is created, RETRIEVE requests a password twice to verify that the user typed the password he wanted. The password may be a combination of any characters except Carriage Return and Line Feed.

When a previously created scrambled data base is accessed, RETRIEVE requests the password, which must be the same as the one specified when the file was created. RETRIEVE uses the password to unscramble the data base. Thus, information security is ensured since no one who does not know the password can unscramble the data base.

NOTE: Sometimes the user may be working with a data base that is already scrambled and may wish to create another scrambled data base using the RESULTS TO modifier or the SAVE command, discussed later in this manual. In this case, if he types only a Carriage Return after the request for a password, RETRIEVE assigns the password for the current data base to the new data base. (A different password may be entered for the second data base, if desired.) In any other circumstances, however, the Carriage Return is not an acceptable response, and the password is requested again.

More information about creating and accessing scrambled data bases may be found in the descriptions of individual commands in this section.

THE CREATE COMMAND

The CREATE command is used to create a new data base directly from the terminal. It takes the form

CREATE *file name* [*file type*] **[SEQUENCED]**

Alternatively, the user may simply type

.CREATE ↵

and RETRIEVE prompts with

DATA BASE:

The user then types the file name and the file type, if desired. The file type is optional; if none is specified, SYMBOLIC is assumed.

Examples

.CREATE TRANSACTIONS ↵ — *Creates a symbolic data base named TRANSACTIONS.*

.CREATE UPDATES BINARY ↵

.CREATE ↵
DATA BASE: UPDATES BINARY ↵ — *Create a binary data base UPDATES.*

.CREATE ↵
DATA BASE: PROFITS SCRAMBLED ↵ — *Creates a scrambled data base PROFITS.*

After the file name and type are specified, RETRIEVE responds with one of the following questions:

NEW BASE,OK?	*When neither a data file with the specified name nor a structure file for that name exists in the user's directory.*
CLEAR OLD BASE,OK?	*When a structure file for the specified file name exists in the user's directory.*
CLEAR OLD FILE,OK?	*When a data file, but no structure file, with the specified name exists in the user's directory.*

These questions may all be answered by typing YES or NO (or Y or N) followed by a Carriage Return. In each case, NO aborts the command and returns the user to command level, indicated by the period. If YES is typed, RETRIEVE requests the new data base structure and records are shown in Section 2.[1] The data records entered are stored on the file with the specified name; the structure information is stored on a file named

file name'STR.*x*'

where the file name is the one specified in the CREATE command, and *x* is the letter representing the current version of RETRIEVE.

CAUTION: When an old data file and/or structure file exist in the user's directory, the new information entered replaces any old information on the files. Thus, the user should abort the CREATE command by typing NO or N in response to the above questions if he wants to save the old files.

Example

```
.CREATE TRANSACTIONS↵
NEW BASE, OK? YES↵


PLEASE TYPE IN THE STRUCTURE OF YOUR DATA BASE

FIELD  NAME,WIDTH,TYPE,DECIMAL PLACES

     1  CUSTOMER,20,C↵
     2  PART,5,C↵
     3  QTY,3,I↵
     4  PRICE,6,N,2↵
     5  DATE,6,I↵
     6  TOTAL,8,N,2↵
     7  ↵

CUSTOMER                PART  QTY PRICE  DATE   TOTAL

AMER. TRANSISTOR,42-85,3,56.79,710201,0↵
TRANS-MAGNETO LTD,42-85,12,56.79,710314,0↵
TRANS-MAGNETO LTD,42-85,5,56.79,710428,0↵
AMER. TRANSISTOR,20-04,2,179,710213,0↵
SIMON IND,20-04,3,179.00,710213,0↵
WATSON INT'L,20-04,3,179.00,710514,0↵
NUCLEAR SYSTEMS,16-34,1,49.80,710201,0↵
ELECTRO LABS,16-34,1,49.80,700115,0↵
GENERAL RADIATION,16-34,1,49.80,701215,0↵
POTOMAC ENT,58-78,1,14.88,710601,0↵
NUCLEAR SYSTEMS,58-78,2,14.88,710727,0↵
SEMICONDUCTOR,58-78,7,14.88,710610,0↵
↵

12 RECORDS
```

1 – See *The CREATE Command: Describing and Creating a New Data Base*, page 8.

```
•PRINT↵

CUSTOMER                PART  QTY   PRICE    DATE     TOTAL

AMER. TRANSISTOR        42-85   3   56.79  710201       .00
TRANS-MAGNETO LTD       42-85  12   56.79  710314       .00
TRANS-MAGNETO LTD       42-85   5   56.79  710428       .00
AMER. TRANSISTOR        20-04   2  179.00  710213       .00
SIMON IND               20-04   3  179.00  710213       .00
WATSON INT'L            20-04   3  179.00  710514       .00
NUCLEAR SYSTEMS         16-34   1   49.80  710201       .00
ELECTRO LABS            16-34   1   49.80  700115       .00
GENERAL RADIATION       16-34   1   49.80  701215       .00
POTOMAC ENT             58-78   1   14.88  710601       .00
NUCLEAR SYSTEMS         58-78   2   14.88  710727       .00
SEMICONDUCTOR           58-78   7   14.88  710610       .00

12 RECORDS

•
```

The optional SEQUENCED modifier in the CREATE command prints the record number and a comma as a prompt for each new record. This modifier may be used only when data input comes directly from the terminal. For example,

```
•CREATE TRANSACTIONS SEQUENCED↵
NEW BASE, OK? YES↵


PLEASE TYPE IN THE STRUCTURE OF YOUR DATA BASE

FIELD  NAME, WIDTH, TYPE, DECIMAL PLACES

    1  CUSTOMER,20,C↵
    2  PART,5,C↵
    3  QTY,3,I↵
    4  PRICE,6,N,2↵
    5  DATE,6,I↵
    6  TOTAL,8,N,2↵
    7  ↵

RECNO  CUSTOMER                PART  QTY PRICE  DATE    TOTAL

1, AMER. TRANSISTOR,42-85,3,56.79,710201,0↵
2, TRANS-MAGNETO LTD,42-85,12,56.79,710314,0↵
3, ↵
```

THE BASE OR LOAD COMMAND

The BASE or LOAD command is used to specify a file as the current data base. It takes the form

BASE
or *file name* **[FIXED] [NEW]** [*file type*]
LOAD

Alternatively, the command may be typed as

.BASE↵
or
.LOAD↵

and RETRIEVE prompts with

DATA BASE:

The user then types the file name, followed by the FIXED and/or NEW modifiers and the file type, all of which are optional. FIXED specifies that the file being declared as data base is in fixed format; LOAD and BASE cannot be used with a free format file. NEW specifies that a new base is being created. If no file type is specified, SYMBOLIC is assumed if the data base is new. If it is old, no type may be specified. RETRIEVE checks the type of the old base and loads it properly.

Examples

.BASE TRANSACTIONS↵
.LOAD TRANSACTIONS↵
.LOAD↵
DATA BASE: TRANSACTIONS↵

These commands are all equivalent. They specify the data base TRANSACTIONS, assuming TRANSACTIONS is an old data base.

.BASE SALES NEW SCRAMBLED↵ *Specifies a new, scrambled data base, SALES.*

.BASE ACCT FIXED NEW↵ *Declares the fixed format file ACCT to be a new data base.*

The BASE or LOAD command releases any data base already in RETRIEVE before the new data base is loaded. This data base file is not destroyed; it is simply removed from RETRIEVE.

The BASE or LOAD command may be used for three purposes:

1. Loading a previously created data base into RETRIEVE.
2. Creating a data base from a free format data file, in conjunction with the APPEND command.
3. Creating a data base from a fixed format data file, either alone or in conjunction with APPEND.

BASE or LOAD may also be used in conjunction with APPEND to create a new data base directly from the terminal. However, CREATE is the best command to use for this purpose since it is equivalent to BASE followed by APPEND.

Loading a Previously Created Data Base

When loading a previously created data base, the BASE or LOAD command takes the form

BASE
or *file name*
LOAD

The NEW and FIXED modifiers are not allowed, since they are meaningless in this case. The file type is not specified. RETRIEVE checks the type of the old data base and loads it as such.

RETRIEVE checks to see if the file name specified is an old data base by checking for the existence of a corresponding structure file in the user's directory. It then loads the data base into RETRIEVE. If the data file is not structured according to the specifications in the structure file, an error message is given.

Example

The previously created data base TRANSACTIONS could be loaded as follows:

```
•BASE TRANSACTIONS↵

12 RECORDS(49)          Twelve records of length 49 have been loaded.

•
```

TRANSACTIONS can only be loaded as above if the corresponding structure file exists in the user's directory. The data in TRANSACTIONS must be in the form specified in the structure file.

If there is a structure file but no data file in the user's directory, as might happen if the data file is accidentally deleted, BASE still reads the structure specifications; it then creates an empty data base file and returns control to the user. The APPEND command, discussed below, could then be used to enter the data records.

Creating a Data Base From a Free Format Data File

One of the features of RETRIEVE is that data files created in EDITOR, in another programming language, or in the EXECUTIVE may be used to create RETRIEVE data bases.[1] RETRIEVE allows creation of data bases from either a free format data file (data values separated by commas) or a fixed format data file (fixed length records, uniformly formatted fields).

To create a new data base from a free format data file, the BASE command is used together with the APPEND command.[2]

First, the BASE command is used to specify the data base name, which must be different from the name of the data file; it should be a completely new file name. The user types

BASE *new base name* [*file type*]

1 – See *Creating Data Base Files*, page 97, for methods of creating such files.

2 – APPEND is discussed fully under *The APPEND Command*, page 40.

If no type is specified, SYMBOLIC is assumed. RETRIEVE responds with

NEW BASE OK?

This question indicates that there is no structure file corresponding to the base name in the user's directory. It may be answered YES (or Y) to confirm the command or NO (or N) to abort it. The question may be suppressed by using the form

BASE ***new base name*** **NEW**

There is then no need to type YES or NO.

When the BASE command is used in either of the above forms, RETRIEVE requests the data base structure as in the CREATE command. After structure input is terminated by a Carriage Return alone, RETRIEVE creates an empty data base file. It then prints a period to indicate it is ready to accept another command.

The APPEND command is used to enter the data on the free format file. The form to use is

APPEND FROM ***free format file name***

This command stores the records from the free format file on the data base file; the structure indicated during the BASE command is used.

The format of the data on the file being appended should be exactly the same as that of data being entered from the terminal during CREATE. Data items should be separated by commas and each record should be terminated by a Carriage Return. If a character item contains a comma, the item must be enclosed in single or double quote marks.

If there is an error in the data file, RETRIEVE detects it and prints a message on the terminal indicating the nature of the error. The user may correct the error in EDITOR and then return to RETRIEVE, use BASE to specify the base name as described above, and then use APPEND to enter the corrected records. There is no need to respecify the data base structure since the structure was saved the first time BASE was used. RETRIEVE assumes an old data base and does not request the structure.

Example

In this example, the user creates a new data base, DIRECTORY, from the data on the free format file ADDRESSES.

```
-COPY ADDRESSES TO T↵

HARKER,RALPH,977 ALTA WAY,BALTIMORE MD,24055
BAKER,ROBERT,546 MARINA,LOS ANGELES CA,90140
JOHNSON,DAVID,948 WESTOVER,OAKLAND CA,95015
PALMER,ARTHUR,147 AVENUE A,BOSTON MASS,02011
OLSON,FRANK,300 BROADWAY,NEW YORK NY,10018
```

```
-RETRIEVE↵

•BASE DIRECTORY↵                    Equivalent to
NEW BASE, OK? Y↵                    .BASE DIRECTORY NEW↵


PLEASE TYPE IN THE STRUCTURE OF YOUR DATA BASE      RETRIEVE requests the
                                                    structure and stores it.
FIELD  NAME, WIDTH, TYPE, DECIMAL PLACES

    1  LNAME, 12, C↵
    2  FNAME, 12, C↵
    3  ADDRESS, 16, C↵
    4  CITY, 2←16, C↵               Control A deletes a character.
    5  ZIP, 5, I↵
    6  ↵

0 RECORDS(62)                       The data base DIRECTORY has been created; it is empty.

•APPEND FROM ADDRESSES↵             The data on ADDRESSES is entered into the base DIRECTORY.


5 RECORDS

•PRINT↵                             The records now in the new data base are displayed.

LNAME      FNAME      ADDRESS          CITY              ZIP

HARKER     RALPH      977 ALTA WAY     BALTIMORE MD      24055
BAKER      ROBERT     546 MARINA       LOS ANGELES CA    90140
JOHNSON    DAVID      948 WESTOVER     OAKLAND CA        95015
PALMER     ARTHUR     147 AVENUE A     BOSTON MASS       02011
OLSON      FRANK      300 BROADWAY     NEW YORK NY       10018

5 RECORDS

•
```

NOTE: If the BASE command is aborted with an Alt Mode/Escape during structure input, the incomplete structure file is automatically deleted.

Creating a Data Base From a Fixed Format Data File

RETRIEVE allows the user to create a data base from a fixed format data file as well as from a free format file. The fixed format file must be in the format of a RETRIEVE data base. Thus, it must

- have fixed length records, each terminated by a Carriage Return, if symbolic, and
- be uniformly formatted; that is, each record must contain the same number of fixed length fields arranged in the same order.

Such files can be created in any of the Tymshare languages by using fixed record length random files; they can also be created in EDITOR or from paper tape.[1]

NOTE: The Line Feed option of the WRITE command must be used if a fixed format data base is created in EDITOR.

A fixed format data file may be declared to be a RETRIEVE data base in two ways: using BASE and APPEND or using BASE alone.

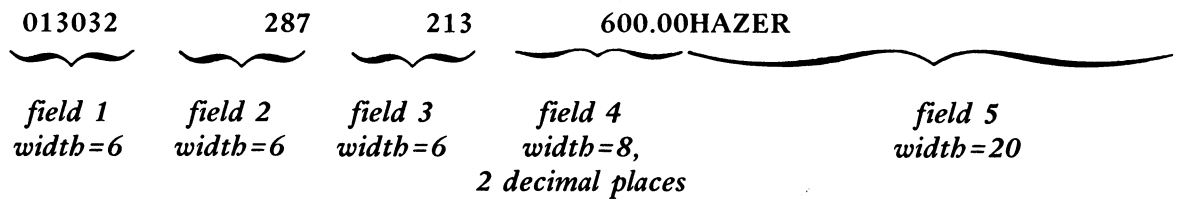
The user may take the same approach as in creating a data base from a free format data file (discussed above), by using BASE and APPEND in the forms

BASE *new base name* [*file type*]
:
:
APPEND FROM *fixed format file name* FIXED

If a file type is not specified in BASE, SYMBOLIC is assumed. During the BASE command, RETRIEVE requests the structure of the new data base. If the fixed format data file being appended does not have the structure specified during base, an error message is printed at the terminal and no data is entered into RETRIEVE during APPEND.

Example

In this example, the user creates a new data base INVENTORY from the data on the fixed format file PARTSLIST. The format of the data file, indicated for the first record, is



The total record length is 47, including the Carriage Return at the end of the record. Note that the descriptions in field 5 must have trailing spaces to fill out the required field width of 20.

```
-COPY PARTSLIST TO T↵

013032   287   213  600.00HAZER
214975  2804  1000    9.75WIDGET
421675   804    52    8.49BOLT
504088   804   388    9.09UPPER OVERHANGER
514975    56    12   65.00FRUZ
604088   406   930   39.72LOWER OVERHANGER
848076  9514  2203   39.16GADGET
```

1 – See *Creating Data Base Files*, page 97.

```
-RETRIEVE↵

•BASE INVENTORY NEW↵


PLEASE TYPE IN THE STRUCTURE OF YOUR DATA BASE

FIELD  NAME,WIDTH,TYPE,DECIMAL PLACES

     1  PARTNO,6,I↵
     2  ONHAND,6,I↵
     3  ORDERED,6,I↵
     4  PRICE,8,N,2↵
     5  DESCR,20,C↵
     6  ↵

0 RECORDS(47)

•APPEND FROM PARTSLIST FIXED↵


7 RECORDS

•LIST↵

 RECNO PARTNO ONHAND          PRICE DESCR
                   ORDERED

     1 013032     287     213  600.00 HAZER
     2 214975    2804    1000    9.75 WIDGET
     3 421675     804      52    8.49 BOLT
     4 504088     804     388    9.09 UPPER OVERHANGER
     5 514975      56      12   65.00 FRUZ
     6 604088     406     930   39.72 LOWER OVERHANGER
     7 848076    9514    2203   39.16 GADGET

7 RECORDS

•
```

Alternatively, the fixed file may be declared to be the data base directly using

BASE *fixed format file name*

followed by a Carriage Return. RETRIEVE responds to this command by typing NEW BASE OK?

to indicate that no structure file corresponding to the specified file name exists in the user's directory. Typing YES (or Y) confirms the command; typing NO (or N) aborts it. This question and the need to answer may be suppressed by using

BASE *fixed format file name* NEW

In either case, RETRIEVE then types the question

FIXED FORMAT?

confirming that the specified file exists in the user's directory. Again, YES (or Y) confirms the BASE command and NO (or N) aborts it. The user should answer NO if the specified file is not in fixed format; if he does, RETRIEVE types a message indicating that APPEND should be used to create a data base from a free format file. This question may be suppressed using the FIXED modifier as follows:

BASE *fixed format file name* FIXED

After FIXED FORMAT? is confirmed (or the FIXED modifier is used), RETRIEVE requests the data base structure as usual; the fixed format data file then becomes the data base.

Example

The fixed format file PARTSLIST appended to the data base INVENTORY in the preceding example could also be declared to be the data base directly as follows:

```
•BASE PARTSLIST ↵
NEW BASE, OK? YES ↵
FIXED FORMAT? YES ↵
```

Equivalent to

```
.BASE PARTSLIST NEW FIXED↵
```

```
PLEASE TYPE IN THE STRUCTURE OF YOUR DATA BASE

FIELD  NAME,WIDTH,TYPE,DECIMAL PLACES

    1  PARTNO,6,I ↵
    2  ONHAND,6,I ↵
    3  ORDERED,6,I ↵
    4  PRICE,8,N,2 ↵
    5  DESCR,20,C ↵
    6  ↵

7 RECORDS(47)
```

```
•LIST↵

RECNO PARTNO ONHAND          PRICE DESCR
                ORDERED

    1 013032     287     213  600.00 HAZER
    2 214975    2804    1000    9.75 WIDGET
    3 421675     804      52    8.49 BOLT
    4 504088     804     388    9.09 UPPER OVERHANGER
    5 514975      56      12   65.00 FRUZ
    6 604088     406     930   39.72 LOWER OVERHANGER
    7 848076    9514    2203   39.16 GADGET

7 RECORDS

•
```

When the BASE command is used to declare an already existing data file as data base, the file type may not be specified. The base assumes the type of the fixed format file. However, if BASE and APPEND are used, binary or scrambled data bases may be created since the data base file being created is a new file. For example, the data base INVENTORY created on page 37 could have been declared binary using the commands

```
.BASE INVENTORY NEW BINARY↵
  :
  :
.APPEND FROM PARTSLIST FIXED↵
```

THE APPEND COMMAND

The APPEND command is used to add records to the current data base. It has the general form

APPEND [FIELDS] [FROM *file name*] [FIXED *or* SEQUENCED][1]

APPEND may be used with the BASE command to create a new data base, as was discussed under the preceding description of BASE, or it may be used to add records to the current data base. The records may be appended from a file, which may be a previously created data base or another data file, or directly from the terminal.

To append records to the current data base directly from the terminal,[2] the form used is simply

```
.APPEND↵
```

After the command is typed, RETRIEVE prompts with the field names just as during CREATE. The new records must be typed in free format just as they are during CREATE: field values are separated by commas, and any field containing a comma must be surrounded by single or double quote marks. The command is terminated by typing a Carriage Return at the beginning of a record.

1 – The RESULTS TO modifier may be used with the APPEND command. See page 61 for a discussion of this feature.

2 – If a command file is being executed, this form of APPEND takes data from the command file instead of from the terminal. See Section 7, *COMMAND FILES.*

Example

```
•BASE TRANSACTIONS↵

12 RECORDS(49)

•PRINT↵

CUSTOMER                PART   QTY   PRICE    DATE      TOTAL

AMER. TRANSISTOR        42-85    3   56.79  710201       .00
TRANS-MAGNETO LTD       42-85   12   56.79  710314       .00
TRANS-MAGNETO LTD       42-85    5   56.79  710428       .00
AMER. TRANSISTOR        20-04    2  179.00  710213       .00
SIMON IND               20-04    3  179.00  710213       .00
WATSON INT'L            20-04    3  179.00  710514       .00
NUCLEAR SYSTEMS         16-34    1   49.80  710201       .00
ELECTRO LABS            16-34    1   49.80  700115       .00
GENERAL RADIATION       16-34    1   49.80  701215       .00
POTOMAC ENT             58-78    1   14.88  710601       .00
NUCLEAR SYSTEMS         58-78    2   14.88  710727       .00
SEMICONDUCTOR           58-78    7   14.88  710610       .00

12 RECORDS

•APPEND↵

CUSTOMER                PART   QTY PRICE   DATE    TOTAL

AMER. TRANSISTOR,42-71,3,12.95,710731,0↵
ELECTRO LABS,16.--34,3,49.80,710801,0↵   Note use of Control A.
NUCLEAR SYSTEMS,20-04,1,179.00,710801,0↵
↵                                        Carriage Return terminates APPEND.

3 RECORDS
```

```
•PRINT↵

CUSTOMER                PART   QTY   PRICE    DATE      TOTAL

AMER. TRANSISTOR        42-85    3   56.79  710201      .00
TRANS-MAGNETO LTD       42-85   12   56.79  710314      .00
TRANS-MAGNETO LTD       42-85    5   56.79  710428      .00
AMER. TRANSISTOR        20-04    2  179.00  710213      .00
SIMON IND               20-04    3  179.00  710213      .00
WATSON INT'L            20-04    3  179.00  710514      .00
NUCLEAR SYSTEMS         16-34    1   49.80  710201      .00
ELECTRO LABS            16-34    1   49.80  700115      .00
GENERAL RADIATION       16-34    1   49.80  701215      .00
POTOMAC ENT             58-78    1   14.88  710601      .00
NUCLEAR SYSTEMS         58-78    2   14.88  710727      .00
SEMICONDUCTOR           58-78    7   14.88  710610      .00
AMER. TRANSISTOR        42-71    3   12.95  710731      .00
ELECTRO LABS            16-34    3   49.80  710801      .00
NUCLEAR SYSTEMS         20-04    1  179.00  710801      .00

15 RECORDS

•
```

RETRIEVE indicates errors typed during this form of APPEND just as it does during CREATE.[1]

The user may add records to the current base from a file, as well as from the terminal, using

APPEND FROM *file name*

The specified file may be a data base, another data file, or the terminal. RETRIEVE recognizes that a file is a data base by the existence of a corresponding structure file in the user's directory. If a data base is being appended, it must have the same structure as the current data base.

If the specified file is not a data base, RETRIEVE assumes it is a free format data file unless FIXED is specified as discussed below. The data on the fixed format file must be in the same form as data entered from the terminal: fields must be separated by commas, and each record must be terminated by a Carriage Return. Thus, if the data appended to TRANSACTIONS in the above example had been on the file NSALES, it could have been appended using

```
.APPEND FROM NSALES↵
```

To add records from a fixed format data file that is not a data base (that is, does not have a structure file), the form

APPEND FROM *file name* FIXED

must be used.[2] The file must have fixed length records of the same size as the current data base.

1 – See *Error Messages*, page 12.

2 – For an example using APPEND with the FIXED modifier, see *Creating a Data Base From a Fixed Format Data File*, page 37.

The FIELDS Modifier

The optional FIELDS modifier in the APPEND command allows records to be accepted field-by-field instead of record-by-record.[1] Thus, the user can include commas and quote marks in character fields without enclosing the field in quote marks.

The form

.APPEND FIELDS↵

allows records to be accepted field-by-field from the terminal. For each record, RETRIEVE displays the record number and then requests each field in the record by printing the field name followed by a colon. Each field must be terminated by a Carriage Return. To terminate input, a Carriage Return alone is typed for the first field in the record.

Example

```
•BASE SUBJECTS↵

2 RECORDS(44)

•PRINT↵

NAME                    CITY                    AGE

CHATSWORTH, RALPH       CHELMSFORD,MASS          53
LATHAM, WILLIAM         LEOMINSTER, MASS         14

2 RECORDS

•APPEND FIELDS↵

RECNO: 3
NAME: MIFFLAN,THOMAS J.↵
CITY: MIDDLETON, NEW HAMP↵
AGE: 38↵

RECNO: 4
NAME:↵

1 RECORDS
```

1 – The FIELDS modifier may also be used with the CREATE command.

```
•PRINT↵

NAME                    CITY                    AGE

CHATSWORTH, RALPH       CHELMSFORD,MASS          53
LATHAM, WILLIAM         LEOMINSTER, MASS         14
MIFFLAN,THOMAS J.       MIDDLETON, NEW HAMP      38

3 RECORDS

•
```

The form

APPEND FIELDS FROM *file name*

allows records to be accepted field-by-field from a file. The fields in the file must each be terminated by a Carriage Return, and the file must contain enough fields to fill each record. Thus, if record number 3 in the above example were on the file NSUB in the form

```
MIFFLAN,THOMAS J.
MIDDLETON,NEW HAMP
38
```

it could have been added to the data base SUBJECTS using

```
.APPEND FIELDS FROM NSUB↵
```

The FIXED modifier cannot be used with this form of APPEND.

The SEQUENCED Modifier

The SEQUENCED modifier may also be used with the APPEND command. A record number and comma are printed as a prompt for each new record. This modifier may be used only when data input comes directly from the terminal. The SEQUENCED and FIELDS modifiers cannot both be used in the same command.

Example

```
•BASE INVOICE↵

3 RECORDS(35)

•FAST↵

AMERICAN PAINT CO       409201   1534.79
MARBLE FINISHING CO.    409202    457.92
CERAMICS INC.           409203    589.05

3 RECORDS
```

```
•APPEND SEQUENCED↵

RECNO   CUSTOMER                    INVOICE
                                              BALANCE

4, TEXTURES UNLIMITED, 409204, 5973.40↵
5, RAINBOW ART CO., 409205, 67.31↵
6, ↵

2 RECORDS

•
```

THE MERGE COMMAND

The MERGE command allows the user to merge data from a fixed format data file into the current data base. It has the form

MERGE [ON *field list*] FROM *fixed format file name*[1]

The field list may consist of one or more current field names separated by commas. The fields specified in this list determine the order of the merge. The fixed format file to be merged need not be a data base; that is, it need not have a structure. It must, however, have records of the same length as the current data base; RETRIEVE assumes that the current data base structure applies.

The field list is optional in the MERGE command. If not present, the first 20 fields of the base are used. If the base contains fewer than 20 fields, all the fields are used.

RETRIEVE requires that the current data base and the file to be merged both be sorted on the fields specified in the field list.[2]

The data base, as it exists just before the MERGE command is given, is written on a backup file for security. MERGE asks the user if he wishes to retain this backup file. For example, if the current data base is STOCK, the following occurs:

```
•MERGE ON STOCK, CERT.NO FROM STOCK70↵

51 RECORDS

STOCK'OLD' CONTAINS A COPY OF YOUR OLD DATA BASE.
SHALL WE RETAIN IT?
```

The user may choose whether to retain the backup file.[3]

1 – The RESULTS TO modifier, described on page 61, may be used with the MERGE command.

2 – See *SORT*, page 59, for a rapid, efficient method of sorting a data base on a specific field list.

3 – The backup data base may be recovered. See *Data Base Recovery*, page 95, for details.

Example

```
-COPY NEWADDR TO T↵

BAKER       JOSEPH     4221 GREEN ST     LOS ANGELES CA   90140
BROWN       EDITH      624 4TH ST        NEW YORK NY      10018
JOHNSON     JOHN       4234 LILAC LANE   BALTIMORE MD     24055
THOMPSON    MARY       39-A ELM ST       PALO ALTO CA     94301
THOMPSON    STANLEY    8915 BAY RD       MENLO PARK CA    94025

-RETRIEVE↵

•BASE DIRECTORY↵

5 RECORDS(62)

•PRINT↵

LNAME       FNAME      ADDRESS           CITY             ZIP

BAKER       ROBERT     546 MARINA        LOS ANGELES CA   90140
HARKER      RALPH      977 ALTA WAY      BALTIMORE MD     24055
JOHNSON     DAVID      948 WESTOVER      OAKLAND CA       95015
OLSON       FRANK      300 BROADWAY      NEW YORK NY      10018
PALMER      ARTHUR     147 AVENUE A      BOSTON MASS      02011

5 RECORDS

•MERGE ON LNAME,FNAME FROM NEWADDR↵

10 RECORDS

DIRECTORY'OLD' CONTAINS A COPY OF YOUR OLD DATA BASE.
SHALL WE RETAIN IT? NO↵
```

•FAST↵ *FAST lists the data base without headings.*

```
BAKER       JOSEPH      4221 GREEN ST      LOS ANGELES CA   90140
BAKER       ROBERT      546 MARINA         LOS ANGELES CA   90140
BROWN       EDITH       624 4TH ST         NEW YORK NY      10018
HARKER      RALPH       977 ALTA WAY       BALTIMORE MD     24055
JOHNSON     DAVID       948 WESTOVER       OAKLAND CA       95015
JOHNSON     JOHN        4234 LILAC LANE    BALTIMORE MD     24055
OLSON       FRANK       300 BROADWAY       NEW YORK NY      10018
PALMER      ARTHUR      147 AVENUE A       BOSTON MASS      02011
THOMPSON    MARY        39-A ELM ST        PALO ALTO CA     94301
THOMPSON    STANLEY     8915 BAY RD        MENLO PARK CA    94025

10 RECORDS
```

THE SAVE COMMAND

As we have seen, RETRIEVE automatically saves the data base as it is entered or modified. However, the user may wish to save part of a data base for processing as another data base or to save his data on a new base of a different type. The SAVE command allows him to do this. It has the form

[*range list*] SAVE TO *file name* [*file type*] [FOR *condition*]

It creates a new data base on the specified file containing the records selected by the range list and/or condition.[1] If the file type is not specified, the new data base has the same type as the old data base. For example, if the current data base is scrambled and a file type is not specified, the new base is also scrambled.

Examples

.SAVE TO LOANDATA BINARY↵ *Creates a binary data base LOANDATA which contains all the records in the current data base.*

.SAVE TO 1968ACCT FOR YR=68↵ *Saves all records for which the value of the field YR is 68 on the base 1968ACCT. The new base 1968ACCT has the same type as the current data base.*

.1:100,200:300 SAVE TO INVESTMENTS SCRAMBLED↵ *Creates a new, scrambled data base INVESTMENTS which contains records 1 through 100 and 200 through 300 of the current data base.*

If the new data base is scrambled, RETRIEVE requests the password. If the current data base is scrambled, a Carriage Return indicates that the same password should be used. In all other cases, a new password is entered.

1 – Range lists are discussed in *Record Number Addressing*, page 15; conditions in *Expressions, Conditions, and the FOR Modifier*, page 17.

RETRIEVE responds to the SAVE command by typing one of the following messages:

NEW BASE,OK?

CLEAR OLD BASE,OK?

CLEAR OLD FILE,OK?

These messages always have the same meaning as they do during CREATE.[1]

NOTE: The SAVE command creates a structure file as well as a data file. The structure of the new data base is the same as that of the current data base.

THE STRUCTURE COMMAND

This command prints a description of the structure of the current data base.

Example

```
•STRUCTURE↵

FIELD   TYPE   WIDTH   NAME

    1   C        20    CUSTOMER
    2   C         5    PART
    3   I         3    QTY
    4   N       6, 2   PRICE
    5   I         6    DATE
    6   N       8, 2   TOTAL
```

Indicates a field width of 8 with 2 decimal places.

```
•
```

THE SIZE COMMAND

This command prints the number of records in the current data base.

Example

```
•SIZE↵

15 RECORDS

•
```

1 – See *The CREATE Command*, page 30, for details.

SECTION 4
RETRIEVAL

In this section, we discuss the following RETRIEVE commands available for displaying selected information from the data base.

Command	Function
LIST PRINT FAST	Display all of the data base, or selected records or fields, in various formats.
COUNT	Prints the number of records satisfying any given condition.
SUM	Prints totals for any numeric fields or expressions.
AVERAGE	Prints average values for any numeric fields or expressions.

LIST, PRINT, AND FAST

The LIST, PRINT, and FAST commands allow the user to display the entire data base, or to display only selected records or fields. These commands all have the same general form:

LIST
or
[*range list*] **PRINT** [*field list*] [**FOR** *condition*]
or
FAST

where the field list may consist of any number of field names separated by commas, and is used to specify the field or fields to be displayed. The range list and condition are used to select records to be displayed.[1]

The only difference among these commands is the format in which the data is displayed:

LIST	Displays the selected data with record numbers and headings.
PRINT	Displays the selected data with headings but no record numbers.
FAST	Displays the selected data without headings or record numbers.

The headings displayed with LIST and PRINT are the field names specified in the data base structure.

To display the entire data base, the commands are given as:

. **LIST** ↵ *or* . **PRINT** ↵ *or* . **FAST** ↵

1 – Range lists and conditions are discussed on pages 15 and 17, respectively.

Example

```
•LIST↵

 RECNO CUSTOMER              PART   QTY   PRICE    DATE     TOTAL

     1 AMER. TRANSISTOR      42-85    3   56.79  710201    170.37
     2 TRANS-MAGNETO LTD     42-85   12   56.79  710314    681.48
     3 TRANS-MAGNETO LTD     42-85    5   56.79  710428    283.95
     4 AMER. TRANSISTOR      20-04    2  179.00  710213    358.00
     5 SIMON IND             20-04    3  179.00  710213    537.00
     6 WATSON INT'L          20-04    3  179.00  710514    537.00
     7 NUCLEAR SYSTEMS       16-34    1   49.80  710201     49.80
     8 ELECTRO LABS          16-34    1   49.80  700115     49.80
     9 GENERAL RADIATION     16-34    1   49.80  701215     49.80
    10 POTOMAC ENT           58-78    1   14.88  710601     14.88
    11 NUCLEAR SYSTEMS       58-78    2   14.88  710727     29.76
    12 SEMICONDUCTOR         58-78    7   14.88  710610    104.16
    13 AMER. TRANSISTOR      42-71    3   12.95  710731     38.85
    14 ELECTRO LABS          16-34    3   49.80  710801    149.40
    15 NUCLEAR SYSTEMS       20-04    1  179.00  710801    179.00

15 RECORDS
```

Example

In this example, the range list and FOR condition clauses are used to select records to be displayed.

```
•1:3,7:9,14 PRINT↵        Displays records 1 through 3, 7 through 9, and 14, with headings.

CUSTOMER              PART   QTY   PRICE    DATE     TOTAL

AMER. TRANSISTOR      42-85    3   56.79  710201    170.37
TRANS-MAGNETO LTD     42-85   12   56.79  710314    681.48
TRANS-MAGNETO LTD     42-85    5   56.79  710428    283.95
NUCLEAR SYSTEMS       16-34    1   49.80  710201     49.80
ELECTRO LABS          16-34    1   49.80  700115     49.80
GENERAL RADIATION     16-34    1   49.80  701215     49.80
ELECTRO LABS          16-34    3   49.80  710801    149.40

7 RECORDS
```

```
•FAST FOR CUSTOMER = 'AMER. TRANSISTOR'↵
```

Displays all records in which CUSTOMER is equal to AMER. TRANSISTOR.

```
AMER. TRANSISTOR      42-85    3   56.79  710201    170.37
AMER. TRANSISTOR      20-04    2  179.00  710213    358.00
AMER. TRANSISTOR      42-71    3   12.95  710731     38.85

3 RECORDS
```

```
•PRINT FOR (PART = '42-85' OR PART = '20-04') AND DATE > 710430↵
```

Displays records for parts 42-85 and 20-04 which were sold after 710430. There are no such parts for PART = '42-85'.

```
CUSTOMER              PART   QTY   PRICE    DATE      TOTAL

WATSON INT'L          20-04    3  179.00  710514    537.00
NUCLEAR SYSTEMS       20-04    1  179.00  710801    179.00

2 RECORDS
```

Example

In this example, the field list is used to display selected fields.

```
•PRINT CUSTOMER, TOTAL↵
```

```
CUSTOMER                TOTAL

AMER. TRANSISTOR        170.37
TRANS-MAGNETO LTD       681.48
TRANS-MAGNETO LTD       283.95
AMER. TRANSISTOR        358.00
SIMON IND               537.00
WATSON INT'L            537.00
NUCLEAR SYSTEMS          49.80
ELECTRO LABS             49.80
GENERAL RADIATION        49.80
POTOMAC ENT              14.88
NUCLEAR SYSTEMS          29.76
SEMICONDUCTOR           104.16
AMER. TRANSISTOR         38.85
ELECTRO LABS            149.40
NUCLEAR SYSTEMS         179.00

15 RECORDS
```

Examples

These examples combine the use of all available options.

```
•LIST PART FOR CUSTOMER = 'NUCLEAR SYSTEMS'↵

 RECNO PART

     7 16-34
    11 58-78
    15 20-04

3 RECORDS

•10:11 FAST CUSTOMER,QTY,TOTAL↵

POTOMAC ENT              1     14.88
NUCLEAR SYSTEMS          2     29.76

2 RECORDS
```

The items in the field list need not be selected in the order in which they occur in the data base records. For example, in the above data base, CUSTOMER occurs before TOTAL, but:

```
•PRINT TOTAL,CUSTOMER↵

   TOTAL CUSTOMER

  170.37 AMER. TRANSISTOR
  681.48 TRANS-MAGNETO LTD
  283.95 TRANS-MAGNETO LTD
  358.00 AMER. TRANSISTOR
  537.00 SIMON IND
  537.00 WATSON INT'L
   49.80 NUCLEAR SYSTEMS
   49.80 ELECTRO LABS
   49.80 GENERAL RADIATION
   14.88 POTOMAC ENT
   29.76 NUCLEAR SYSTEMS
  104.16 SEMICONDUCTOR
   38.85 AMER. TRANSISTOR
  149.40 ELECTRO LABS
  179.00 NUCLEAR SYSTEMS

15 RECORDS
```

Note that LIST, PRINT, and FAST print an extra space between each field column for a more attractive listing. These spaces are not part of the data base. For example, if a data base had the structure

PART,5,C
REST,5,C

and the first record were

12345ABCDE

it would be displayed as

12345 ABCDE

THE COUNT COMMAND

This command prints the number of records satisfying a given condition. It has the form

[*range list*] COUNT [FOR *condition*]

Example

```
•FAST↵

AMER. TRANSISTOR      42-85    3   56.79 710201    170.37
TRANS-MAGNETO LTD     42-85   12   56.79 710314    681.48
TRANS-MAGNETO LTD     42-85    5   56.79 710428    283.95
AMER. TRANSISTOR      20-04    2  179.00 710213    358.00
SIMON IND             20-04    3  179.00 710213    537.00
WATSON INT'L          20-04    3  179.00 710514    537.00
NUCLEAR SYSTEMS       16-34    1   49.80 710201     49.80
ELECTRO LABS          16-34    1   49.80 700115     49.80
GENERAL RADIATION     16-34    1   49.80 701215     49.80
POTOMAC ENT           58-78    1   14.88 710601     14.88
NUCLEAR SYSTEMS       58-78    2   14.88 710727     29.76
SEMICONDUCTOR         58-78    7   14.88 710610    104.16
AMER. TRANSISTOR      42-71    3   12.95 710731     38.85
ELECTRO LABS          16-34    3   49.80 710801    149.40
NUCLEAR SYSTEMS       20-04    1  179.00 710801    179.00

15 RECORDS
```

```
.COUNT FOR PART = '16-34'↵

4 RECORDS

.9:$ COUNT FOR PART = '16-34'↵

2 RECORDS
```

The command

.COUNT↵

is equivalent to the SIZE command; that is, it prints the number of records in the data base.

SUM AND AVERAGE

The SUM command[1] computes totals for any specified numeric fields or expressions, and takes the form

[*range list*] SUM [*expression list*] [FOR *condition*]

The AVERAGE command has the same form as SUM:

[*range list*] AVERAGE [*expression list*] [FOR *condition*]

but it displays the average values of all expressions in the expression list, computed for the records selected by the range list and condition. All expressions in the expression list must be numeric, for both AVERAGE and SUM.[2] .

SUM prints the total to the accuracy defined for the item. If none is defined, results are printed to six decimal places. All expressions are printed in free form.

The command

.SUM↵

prints the total values for all numeric fields in the data base.

The command

.AVERAGE↵

prints the average value for all numeric fields in the data base.

1 – See *Computing Totals: The SUM Command*, page 24.

2 – See page 17 for a description of expressions.

Example

```
•PRINT↵

CUSTOMER              PART   QTY   PRICE    DATE     TOTAL

AMER. TRANSISTOR      42-85    3   56.79  710201    170.37
TRANS-MAGNETO LTD     42-85   12   56.79  710314    681.48
TRANS-MAGNETO LTD     42-85    5   56.79  710428    283.95
AMER. TRANSISTOR      20-04    2  179.00  710213    358.00
SIMON IND             20-04    3  179.00  710213    537.00
WATSON INT'L          20-04    3  179.00  710514    537.00
NUCLEAR SYSTEMS       16-34    1   49.80  710201     49.80
ELECTRO LABS          16-34    1   49.80  700115     49.80
GENERAL RADIATION     16-34    1   49.80  701215     49.80
POTOMAC ENT           58-78    1   14.88  710601     14.88
NUCLEAR SYSTEMS       58-78    2   14.88  710727     29.76
SEMICONDUCTOR         58-78    7   14.88  710610    104.16
AMER. TRANSISTOR      42-71    3   12.95  710731     38.85
ELECTRO LABS          16-34    3   49.80  710801    149.40
NUCLEAR SYSTEMS       20-04    1  179.00  710801    179.00

15 RECORDS

•SUM TOTAL↵          Prints the sum of all values of TOTAL.

        SUM  OF

    3233.25  TOTAL

15 RECORDS

•AVERAGE TOTAL↵      Prints the average of all values of TOTAL.

    AVERAGE  OF

     215.55  TOTAL

15 RECORDS
```

```
•SUM TOTAL FOR CUSTOMER = 'AMER. TRANSISTOR'↵
```

Prints total sales made to AMER. TRANSISTOR.

```
        SUM   OF

     567.22   TOTAL

3 RECORDS
```

```
•AVERAGE QTY FOR PART = '20-04'↵
```

Prints average quantity of part 20-04 sold.

```
    AVERAGE   OF

       2.25   QTY

4 RECORDS
```

```
•1:3 SUM QTY,PRICE,QTY*PRICE↵
```

*Sums QTY, PRICE, and QTY*PRICE for records 1 through 3.*

```
        SUM   OF

         20   QTY
     170.37   PRICE
     1135.8   QTY*PRICE

3 RECORDS
```

```
•
```

SECTION 5
UPDATING

The following RETRIEVE commands are available for updating the data base.

Command	Function
CHANGE MODIFY	Allow the user to change selected records or selected fields within a record. CHANGE displays the old field values; MODIFY does not.
DELETE	Deletes selected records. Allows saving of old data base before deletion occurs.
REPLACE	Replaces selected fields with any desired fields or expressions.
SORT	Sorts a data base on up to 20 fields. Allows saving of unsorted data base.

CHANGE AND MODIFY

These commands allow the user to change selected records or selected fields within a record. They have the general form

[*range list*] CHANGE *or* MODIFY [*field list*] [FOR *condition*][1]

All features of CHANGE are discussed in Section 2.[2] MODIFY is the same as CHANGE except that it does not display a heading showing the field names changed or the old field values. For example,

```
•1:8 PRINT↵

CUSTOMER                PART   QTY   PRICE    DATE     TOTAL

AMER. TRANSISTOR        42-85    3   56.79  710201    170.37
TRANS-MAGNETO LTD       42-85   12   56.79  710314    681.48
TRANS-MAGNETO LTD       42-85    5   56.79  710428    283.95
AMER. TRANSISTOR        20-04    2  179.00  710213    358.00
SIMON IND               20-04    3  179.00  710213    537.00
WATSON INT'L            20-04    3  179.00  710514    537.00
NUCLEAR SYSTEMS         16-34    1   49.80  710201     49.80
ELECTRO LABS            16-34    1   49.80  700115     49.80

8 RECORDS
```

1 – The RESULTS TO and ALL TO modifiers may be used with the CHANGE and MODIFY commands. See page 61 for a discussion of these destination modifiers.

2 – See *The CHANGE Command*, page 20.

```
•CHANGE QTY,TOTAL FOR 'AMER' IN CUSTOMER AND PART = '20-04'↵

QTY     TOTAL : QTY     TOTAL
  2    358.00 : 3,537↵
1 RECORDS

•FAST FOR 'AMER' IN CUSTOMER AND PART = '20-04'↵

AMER. TRANSISTOR      20-04    3 179.00 710213    537.00
1 RECORDS
```

In the above example, the CHANGE command allows the user to change the value of the fields QTY and TOTAL for one specified record. The MODIFY command could also be used to do this, for example,

```
•MODIFY QTY,TOTAL FOR 'AMER' IN CUSTOMER AND PART = '20-04'↵
3,537↵
1 RECORDS

•
```

Headings and old values are not displayed when MODIFY is used.

DELETE

The DELETE command[1] has the general form

[*range list*] DELETE [FOR *condition*]

It deletes the records specified by the range list and/or condition. If both these options are omitted, the entire data base is deleted. Recovery of the data base to its state before a DELETE occurred is allowed under certain conditions.[2]

1 – See *The DELETE Command*, page 22, for details.

2 – See *Data Base Recovery*, page 95, for details.

REPLACE

This command allows the user to replace selected fields with any desired expressions. It has the general form

[*range list*] **REPLACE** *field*$_1$ **WITH** *expression*$_1$ [,*field*$_2$ **WITH** *expression*$_2$,...] [**FOR** *condition*][1]

All features of REPLACE are discussed in Section 2 of this manual.[2]

SORT

The SORT command allows the user to perform an ascending sort on as many as 20 fields in a data base. It has the general form

SORT [ON *field list*]**[3]**

where the field list may contain up to 20 field names, separated by commas. The command may also be expressed as

SORT [BY *field list*]

The field list is used to establish a sorting hierarchy so that if the values of the first field in the list are identical in each of two records, the values of the second field in the list are compared; if the values of the second field are identical, the values of the third are compared, etc.

The field list is optional in the SORT command. If not present, the first 20 fields of the base are used. If the base contains fewer than 20 fields, all the fields are used.

SORT automatically stores the sorted records on the current data base file. The unsorted data base is written on a backup file for security. The user may choose to retain or delete this file. See *Data Base Recovery*, page 95, for information about restoring the unsorted data base.

1 – See *Destination Modifiers*, page 61, for a discussion of the RESULTS TO and ALL TO modifiers which may be used with the REPLACE command.

2 – See *The REPLACE Command*, page 23.

3 – The RESULTS TO modifier may also be used with SORT. See page 61 for a description of RESULTS TO.

Example

```
•PRINT↵

CUSTOMER              PART   QTY   PRICE    DATE     TOTAL

AMER. TRANSISTOR      42-85    3   56.79  710201    170.37
TRANS-MAGNETO LTD     42-85   12   56.79  710314    681.48
TRANS-MAGNETO LTD     42-85    5   56.79  710428    283.95
AMER. TRANSISTOR      20-04    3  179.00  710213    537.00
SIMON IND             20-04    3  179.00  710213    537.00
WATSON INT'L          20-04    3  179.00  710514    537.00
NUCLEAR SYSTEMS       16-34    1   49.80  710201     49.80
ELECTRO LABS          16-34    1   49.80  700115     49.80
GENERAL RADIATION     16-34    1   49.80  701215     49.80
POTOMAC ENT           58-78    1   14.88  710601     14.88
NUCLEAR SYSTEMS       58-78    2   14.88  710727     29.76
SEMICONDUCTOR         58-78    7   14.88  710610    104.16
AMER. TRANSISTOR      42-71    3   12.95  710731     38.85
ELECTRO LABS          16-34    3   49.80  710801    149.40
NUCLEAR SYSTEMS       20-04    1  179.00  710801    179.00

15 RECORDS

•SORT BY PART↵

TRANSACTIONS'OLD' CONTAINS A COPY OF YOUR OLD DATA BASE.
SHALL WE RETAIN IT? NO↵

•FAST CUSTOMER, PART↵        The data base is now sorted by PART.

ELECTRO LABS          16-34
ELECTRO LABS          16-34
GENERAL RADIATION     16-34
NUCLEAR SYSTEMS       16-34
NUCLEAR SYSTEMS       20-04
WATSON INT'L          20-04
SIMON IND             20-04
AMER. TRANSISTOR      20-04
AMER. TRANSISTOR      42-71
TRANS-MAGNETO LTD     42-85
TRANS-MAGNETO LTD     42-85
AMER. TRANSISTOR      42-85
POTOMAC ENT           58-78
NUCLEAR SYSTEMS       58-78
SEMICONDUCTOR         58-78

15 RECORDS

•
```

Destination Modifiers

Certain RETRIEVE commands allow destination modifiers. Such modifiers indicate that the current data base should not be altered, but that the new information should be placed in a new data base.

The RESULTS TO modifier may be used with the APPEND, SORT, MERGE, CHANGE, MODIFY, and REPLACE commands. The ALL TO modifier may be used only with CHANGE, MODIFY, and REPLACE. Both modifiers are discussed below, indicating their meanings for each command.

The destination modifiers must appear at the end of the command in the form

RESULTS TO
or *file name* [*file type*]
ALL TO

where file name indicates the name of the new data base to be created, and file type indicates whether the new data base should be symbolic, binary, or scrambled. If the file type is omitted, the type of the old base is assumed. The new base and its corresponding structure file is placed in the user's directory. The structure of the new base is the same as that of the current base.

The RESULTS TO modifier specifies that the current data base remain unchanged and that the results of the command be stored on a new data base.

For example, using the data base files

JANSALES (Contains sales data for January)

FEBSALES (Contains sales data for February)

the commands

```
.BASE JANSALES↵
  :
.APPEND FROM FEBSALES RESULTS TO SALES↵
```

cause the records on FEBSALES to be added to the records on JANSALES and the results to be stored on SALES. The data base JANSALES (as well as FEBSALES) remains the same after the APPEND. The data base JANSALES remains the current RETRIEVE data base. If the command

```
.APPEND FROM FEBSALES↵
```

were used, JANSALES would contain the sales records for January and February.

Examples

```
.APPEND RESULTS TO EMPLOYEES BINARY↵
```

Creates a binary data base called EMPLOYEES containing whatever records are in the current data base plus the records typed at the terminal.

```
.APPEND FROM BILLING RESULTS TO DELINQUENT SCRAMBLED↵
```

Creates a scrambled data base, DELINQUENT, containing the records in the current data base plus those in the file BILLING.

.APPEND FIELDS RESULTS TO RESUMES↵

Creates a data base, RESUMES, containing the current data base plus records entered field-by-field from the terminal. The type of the current data base is assumed. Thus, if the current data base is symbolic, RESUMES is symbolic; if binary, RESUMES is binary, etc.

.MERGE ON SALESMAN FROM EASTSALES RESULTS TO BSALES BINARY↵

The new base BSALES is binary.

.MERGE ON DEPT,EMPLOYEE FROM NREVIEWS RESULTS TO RESUMES SCRAMBLED↵

Creates the scrambled data base RESUMES.

.SORT ON DEPT,LNAME,FNAME RESULTS TO ROSTER BINARY↵

Creates a binary data base, ROSTER, containing the sorted data records.

NOTE: When the RESULTS TO modifier is used with SORT or MERGE, the base 'OLD' backup file is not created. The current data base contains the original data; the data base named in the RESULTS TO modifier contains the new data.

Consider the sample below.

```
-COPY EASTSALES TO T↵

BROWN      NE        240.09
MARGOLES   SE       1968.20

-RETRIEVE↵

•BASE WESTSALES↵

3 RECORDS(25)

•PRINT↵

SALESMAN    REGION      TOTAL

FISHER      NW          599.99
GOLDBERG    SW          650.28
ROBERTS     SW         1480.36

3 RECORDS

•MERGE ON SALESMAN FROM EASTSALES RESULTS TO SALES↵
NEW BASE, OK? YES↵

5 RECORDS
```

```
•BASE SALES↵

5 RECORDS(25)

•FAST↵

BROWN       NE        240.09
FISHER      NW        599.99
GOLDBERG    SW        650.28
MARGOLES    SE       1968.20
ROBERTS     SW       1480.36

5 RECORDS

•QUIT↵

-
```

When used with CHANGE, MODIFY, and REPLACE, the RESULTS TO modifier specifies that a new base, containing the changed records only, be created. The current data base is unaltered.

Example

```
•PRINT↵

EMPLOYEE                SOC.SEC       SALARY HRS      PAY

ANDREWS KARL            469-20-9531     2.35  48    112.80
BRADFORD SUSAN          202-46-9277     4.90  40    196.00
NELSON DONALD           311-61-2629     5.10  44    224.40
PALMER DAVID            357-48-3158   410.00  -1    410.00
PARKER MARY             351-04-8260     4.10  40    164.00
RODRIGUES MARIA         373-75-7302   198.70  -1    198.70
WINTON JOHN             421-98-7243     4.90  48    235.20

7 RECORDS

•CHANGE HRS FOR HRS=-1 RESULTS TO SALARIED↵
NEW BASE, OK? YES↵

HRS : HRS

 -1 : 40↵
 -1 : 44↵

2 RECORDS
```

```
•FAST FOR HRS = -1↵

PALMER DAVID          357-48-3158  410.00   -1   410.00
RODRIGUES MARIA       373-75-7302  198.70   -1   198.70

2 RECORDS
```

The current data base remains unchanged.

```
•BASE SALARIED↵

2 RECORDS(48)

•PRINT↵

EMPLOYEE              SOC.SEC      SALARY  HRS      PAY

PALMER DAVID          357-48-3158  410.00   40   410.00
RODRIGUES MARIA       373-75-7302  198.70   44   198.70

2 RECORDS
```

The changed records only are stored on SALARIED.

```
•
```

The ALL TO modifier, used only with CHANGE, MODIFY, and REPLACE, specifies that a new data base containing all records in the current data base, both changed and unchanged, be created. The current data base remains unchanged. Thus, for the above data base, the following could occur:

```
•FAST↵

ANDREWS KARL          469-20-9531    2.35   48   112.80
BRADFORD SUSAN        202-46-9277    4.90   40   196.00
NELSON DONALD         311-61-2629    5.10   44   224.40
PALMER DAVID          357-48-3158  410.00   -1   410.00
PARKER MARY           351-04-8260    4.10   40   164.00
RODRIGUES MARIA       373-75-7302  198.70   -1   198.70
WINTON JOHN           421-98-7243    4.90   48   235.20

7 RECORDS
```

```
•REPLACE HRS WITH HRS-4,PAY WITH SALARY*HRS, ↵
FOR 'PARKER' IN EMPLOYEE ↵
NEW BASE, OK? YES ↵

1 RECORDS

•FAST FOR 'PARKER' IN EMPLOYEE ↵

PARKER MARY          351-04-8260    4.10   40   164.00

1 RECORDS

•BASE NEWPAY ↵

7 RECORDS(48)

•PRINT ↵

EMPLOYEE             SOC.SEC        SALARY HRS      PAY

ANDREWS KARL         469-20-9531      2.35  48   112.80
BRADFORD SUSAN       202-46-9277      4.90  40   196.00
NELSON DONALD        311-61-2629      5.10  44   224.40
PALMER DAVID         357-48-3158    410.00  -1   410.00
PARKER MARY          351-04-8260      4.10  36   147.60
RODRIGUES MARIA      373-75-7302    198.70  -1   198.70
WINTON JOHN          421-98-7243      4.90  48   235.20

7 RECORDS

•
```

Remains unchanged.

NEWPAY contains the changed and unchanged data records.

SECTION 6
REPORT GENERATION

RETRIEVE may be used to prepare reports from a data base. The reports may include column headings, totals for numeric information, subtotals, and any values which can be computed from the items of information in a single record. Reports may show all of the records in the data base, selected records, or only totals. If desired, the terminal output can be formatted into 8½- by 11-inch pages, with top and bottom margins.

THE REPORT COMMAND

The REPORT command is used to initiate report generation. This command has the form

[*range list*] **REPORT** [**FOR** *condition*]

where the range list and condition specify a particular group of records to be reported.[1] If neither the range list nor condition are specified, all of the records in the data base are reported.

Examples

The command

. REPORT ↵

generates a report on all records in the data base.

The command

. 50:150 REPORT FOR NETPAY > 300.00 ↵

generates a report on all records in the range 50 through 150 for which the value of NETPAY is greater than 300.

THE REPORT DESCRIPTION DIALOGUE

After the REPORT command is typed, RETRIEVE carries on a dialogue with the user in order to specify the format of the report. The information requested by RETRIEVE during this dialogue is summarized in the following table. Following the table, the dialogue is explained in detail.

1 - Range lists are discussed in *Record Number Addressing*, page 15; conditions are discussed in *Expressions, Conditions, and the FOR Modifier*, page 17.

SUMMARY OF REPORT DESCRIPTION DIALOGUE	
RETRIEVE Prompt	**User Responses Allowed**[1]
REPORT OUTPUT TO:	**T** (For the terminal.) *or* *file name* (The OLD FILE or NEW FILE message must be confirmed or aborted.)
REPORT FORM NAME:	*name or* ↵
UPDATE REPORT FORM? (Asked only if an old report form name is specified.)	**YES** *or* **NO** (If NO, old description is used; all subsequent dialogue is skipped.)
HEADING?	**YES** *or* **NO**
DOUBLE SPACE?	**YES** *or* **NO**
TOTALS?	**YES** *or* **NO** (If NO, RETRIEVE now requests the column descriptions, skipping the next three questions.)
SUBTOTALS?	**YES** *or* **NO** (If NO, RETRIEVE skips next question.)
BY ITEMS:	A list of items on which to subtotal, e.g., **DIVISION,DISTRICT,SALESMAN**
SUMMARY REPORT ONLY?	**YES** *or* **NO**
COL WIDTH,CONTENTS 1 ⋮ (Requests description of each column in report.)	*column width,expression* (For each column desired.)
COL HEADING 1 ⋮ (Requests column headings.)	*heading* (For each column. If no heading is desired, a space is typed followed by a Carriage Return.)
COL TOTALS?,NO. OF DECIMAL PLACES *column number* ⋮ (Asked only if totals were requested above; a column number is prompted for each numeric column.)	**NO** *or* **YES** [,*number of decimal places*] (For each column prompted.)
COL NO. OF DECIMAL PLACES *column number* ⋮ (Asked only if totals were not requested; a column number is prompted for each numeric column.)	*number of decimal places* *or* ↵ (For each column prompted.)

1 – Each response must be followed by a Carriage Return.

The first question and response in the dialogue is

REPORT OUTPUT TO: T↵ *or* ***file name*↵**

The user chooses the output medium for the report: the terminal (T) or a file.

If output is to a file, RETRIEVE checks to see if a file with the specified name exists in the user's directory. If no such file exists in the user's directory, RETRIEVE responds with

NEW FILE

Otherwise, it responds with

OLD FILE

To confirm that the report be printed on the specified file, the user types a Carriage Return. To abort the command, he types an Alt Mode/Escape or NO and a Carriage Return.

If the report written to a file contains no totals or headings and is single spaced, it may be used as a data base for subsequent RETRIEVE sessions. Note, however, that the report generator puts one space between each column. To use the report file as a data base, the user must enter the structure using the BASE command as described in Section 3, defining the field width to allow for the extra spaces. The report generator does not create a structure file.

After specifying the report output medium, the user must specify the report form name. The dialogue is:

REPORT FORM NAME: ***name*↵** *or* **↵**

If a name is specified, the report specification is saved in a file named ***name*'REP'**; otherwise, the report is not saved. This feature allows a standard report to be prepared without requiring the user to describe it after the first time. If the report description file already exists, RETRIEVE allows the user to modify it by asking

UPDATE REPORT FORM? YES↵ *or* **NO↵**

If the response is YES, the remainder of the report description dialogue continues and the new report description is saved in the old file. If the response is NO, the report is prepared immediately using the old description. All subsequent questions are skipped.

If no report description file is desired, the user responds to the REPORT FORM NAME: question with a Carriage Return.

The remainder of the dialogue describes the report. RETRIEVE next asks

HEADING? YES↵ *or* **NO↵**

If the user types NO, RETRIEVE proceeds to the next question. If he types YES, the report contains column headings and is formatted into 8½- by 11-inch pages. A subsequent series of questions describes the text of the headings. The dialogue continues with:

DOUBLE SPACE? YES↵ *or* **NO↵**

YES specifies that the report is double spaced; NO specifies single spacing.

If double spacing is requested, only the body of the report is double spaced; headings and totals are single spaced.

TOTALS? YES↵ *or* **NO↵**

If the response is NO, RETRIEVE goes directly to the column description question. If it is YES, RETRIEVE accumulates totals of report columns containing numeric data as specified by the user. The user specifies which columns of the report are to be totaled by responding to the COL TOTALS? question described later.

If totals are requested, RETRIEVE next asks if subtotals are desired, as follows:

SUBTOTALS? YES↵ *or* **NO↵**

If the user types NO, RETRIEVE proceeds to the SUMMARY REPORT ONLY question. If he types YES, the report shows subtotals for numeric report columns each time the value of a selected data base field changes. The user specifies the fields to be subtotaled by answering the next question:

BY ITEMS: ***field list*↵**

Each time the value of one of the specified fields changes, as the report is being generated, the report generator displays the subtotal for each numeric report column for which totals were specified. Up to ten items may be specified in the field list. The field list items may be character items; in this case, no subtotal is printed.

Example

```
SUBTOTALS? YES↵

BY ITEMS: CUSTOMER,DATE↵
```

In this example, the user specifies subtotals for the fields named CUSTOMER and DATE in the current data base. Thus, if the data base contains the fields

CUSTOMER DATE QTY PRICE

and the report is to contain the columns

DATE QTY PRICE QTY*PRICE

with totals specified for QTY*PRICE only, a subtotal for QTY*PRICE will be displayed each time DATE changes, and also each time CUSTOMER changes.

The next question and response in the dialogue is:

SUMMARY REPORT ONLY? YES↵ *or* **NO↵**

YES specifies that only total and subtotal headings and values are shown on the report. Column headings appear if they have been requested. Individual records do not appear. NO specifies a full report including individual records, totals, and subtotals.

The next section of the report description dialogue covers the report column contents and headings, and selection of columns to be totaled. First, the column contents are specified:

COL WIDTH,CONTENTS

***column number number of characters,expression*↵** *or* **↵**

RETRIEVE prompts the user with a column number. The user responds with the number of characters in the column and an expression describing the value to be reported on that column.[1] RETRIEVE continues to prompt with column numbers until the user types only a Carriage Return to terminate the column description dialogue.

1 – Expressions are defined in *Expressions, Conditions, and the FOR Modifier*, page 17.

Example

In this example, the user specifies five report columns. The column contents dialogue is terminated with a Carriage Return.

```
COL   WIDTH,CONTENTS

1       8,PART↵
2       8,DATE↵
3       5,QTY↵
4       9,PRICE↵
5       9,PRICE*QTY↵
6       ↵
```

Up to 25 columns may be specified in a single report. Columns are right justified for numeric items and left justified for character items. Caution should be exercised in setting column widths to avoid requesting more columns than may be displayed on the terminal being used.

If the user answered YES to the HEADING? question, RETRIEVE requests a description of the column heading text. Column headings need not be the same as field names in the structure file. The dialogue is:

```
COL   HEADING
column number   text↵
```

The user may specify up to three lines, of up to 120 characters each, of column heading text. The heading for a single column may not exceed the width of that column. The user indicates the beginning of a new line in a column heading by typing a slash (/) between successive sections of the heading. If no heading is desired for a particular column, the user types one or more spaces followed by a Carriage Return.

Example

The user specifies a column heading for each column.

```
COL   HEADING

1       PART/NUMBER↵
2       DATE↵
3       QUANT↵
4       PRICE↵
5       EXTENDED/PRICE↵
```

On the report, the headings print as follows:

```
 PART                                        EXTENDED
NUMBER          DATE  QUANT       PRICE      PRICE
```

If the user has requested totals, RETRIEVE asks him to identify the columns to be totaled and to specify the number of decimal places in each column as follows:

COL TOTALS? ,NO. OF DECIMAL PLACES
column number NO *or* YES [,*decimal places*]↵

RETRIEVE prompts with the column number of each numeric column in the report. (Non-numeric columns may not be totaled, nor may they have decimal places specified.) If the user desires no totals for the column, he types NO; otherwise, he types YES, and then specifies the number of decimal places, if any, to be shown in the column. If decimal places are not specified, the data is right justified in the column to the least significant digit. If specified, the data is right justified with decimal points aligned. Any data (excluding totals) which cannot fit in the column causes termination of the report.

Example

In this example, columns 2 and 3 are intended to be integer values and contain no decimal places. Column 4 contains two decimal places and is not to be totaled. Column 5 is to be totaled and contains two decimal places.

```
COL  TOTALS? ,NO. OF DECIMAL PLACES

2      NO↵
3      NO↵
4      NO,2↵
5      YES,2↵
```

CAUTION: The user should ensure that report column widths are large enough to allow the totals to be printed below the columns.

If the user has not requested totals, RETRIEVE asks for the number of decimal places in each column:

COL NO. OF DECIMAL PLACES
column number [*decimal places*]↵

The user types the number of decimal places in each numeric field as described above under the COL TOTALS? question.

Example

This is the same as the previous example except that no totals are specified in the report.

```
COL  NO. OF DECIMAL PLACES

2
3
4     2
5     2
```

RETRIEVE terminates the report description dialogue with the following message indicating the width of the report in characters:

REPORT IS *w* CHARACTERS WIDE

where *w* is the report width in characters.

RETRIEVE then prepares the report, printing it on the terminal or writing it on a file, as specified in the dialogue.

A SAMPLE REPORT

In this section we present a sample session illustrating the use of the REPORT command. More examples using REPORT may be seen in Section 9, *SAMPLE RETRIEVE SESSIONS*.

A data base containing customer orders, called RAWTRANS, is in sequence by date, that is, in the order in which the transactions were received. The user needs a report showing the individual orders placed and the total amount owed by each customer.

The data base is as follows:

```
.BASE
DATA BASE: RAWTRANS

12 RECORDS(41)

.STRUCTURE

FIELD   TYPE  WIDTH  NAME

    1   C      20    CUSTOMER
    2   C       5    PART
    3   I       3    QTY
    4   N     6, 2   PRICE
    5   I       6    DATE
```

```
•PRINT ↵

CUSTOMER                PART   QTY   PRICE    DATE

SIMON IND               20-04    3  179.00  710113
ELECTRO LABS            16-34    1   49.80  710115
AMER. TRANSISTOR        42-85    3   56.79  710201
AMER. TRANSISTOR        20-04    2  179.00  710213
TRANS-MAGNETO LTD       42-85   12   56.79  710314
NUCLEAR SYSTEMS         58-78    2   14.88  710327
GENERAL RADIATION       16-34    3   49.80  710415
TRANS-MAGNETO LTD       42-85    5   56.79  710428
WATSON INT'L            20-04    1  179.00  710514
POTOMAC ENT             58-78    1   14.88  710601
SEMICONDUCTOR           58-78    7   14.88  710610
NUCLEAR SYSTEMS         16-34    1   49.80  710801

12 RECORDS
```

In order to prepare the report, the user first sorts the data base by CUSTOMER, as follows:

```
•SORT BY CUSTOMER, PART ↵

RAWTRANS'OLD' CONTAINS A COPY OF YOUR OLD DATA BASE.
SHALL WE RETAIN IT? NO ↵
```

The user then specifies the report. The report description is saved in a file called RECEIVABLES'REP' for later use. The user specifies headings, column width and format, and total price accumulated for each customer, as shown below:

```
•REPORT ↵

REPORT OUTPUT TO: T ↵

REPORT FORM NAME: RECEIVABLES ↵

HEADING? YES ↵

DOUBLE SPACE? NO ↵

TOTALS? YES ↵

SUBTOTALS? YES ↵

BY ITEMS: CUSTOMER ↵

SUMMARY REPORT ONLY? NO ↵

COL   WIDTH,CONTENTS

1       8,PART ↵
2       8,DATE ↵
3       5,QTY ↵
4       9,PRICE ↵
5       9,PRICE*QTY ↵
6       ↵

COL   HEADING

1       PART/NUMBER ↵
2       DATE ↵
3       QUANT ↵
4       PRICE ↵
5       EXTENDED/PRICE ↵

COL   TOTALS? ,NO. OF DECIMAL PLACES

2       NO ↵
3       NO ↵
4       NO,2 ↵      Specifies no total; two decimal places in column.
5       YES,2 ↵     Specifies total; two decimal places in column and total.

REPORT IS 43 CHARACTERS WIDE
```

RETRIEVE then produces the report:

PAGE 1
DATE: 7/18 11:57
DATA BASE: RAWTRANS
REPORT FORM: RECEIVABLES

PART NUMBER	DATE	QUANT	PRICE	EXTENDED PRICE
* CUSTOMER: AMER. TRANSISTOR				
20-04	710213	2	179.00	358.00
42-85	710201	3	56.79	170.37
* TOTAL FOR CUSTOMER: AMER. TRANSISTOR				528.37
* CUSTOMER: ELECTRO LABS				
16-34	710115	1	49.80	49.80
* TOTAL FOR CUSTOMER: ELECTRO LABS				49.80
* CUSTOMER: GENERAL RADIATION				
16-34	710415	3	49.80	149.40
* TOTAL FOR CUSTOMER: GENERAL RADIATION				149.40
* CUSTOMER: NUCLEAR SYSTEMS				
16-34	710801	1	49.80	49.80
58-78	710327	2	14.88	29.76
* TOTAL FOR CUSTOMER: NUCLEAR SYSTEMS				79.56
* CUSTOMER: POTOMAC ENT				
58-78	710601	1	14.88	14.88
* TOTAL FOR CUSTOMER: POTOMAC ENT				14.88
* CUSTOMER: SEMICONDUCTOR				
58-78	710610	7	14.88	104.16
* TOTAL FOR CUSTOMER: SEMICONDUCTOR				104.16
* CUSTOMER: SIMON IND				
20-04	710113	3	179.00	537.00
* TOTAL FOR CUSTOMER: SIMON IND				537.00

PAGE 2 7/18

PART NUMBER	DATE	QUANT	PRICE	EXTENDED PRICE
* CUSTOMER: TRANS-MAGNETO LTD				
42-85	710314	12	56.79	681.48
42-85	710428	5	56.79	283.95
* TOTAL FOR CUSTOMER: TRANS-MAGNETO LTD				965.43
* CUSTOMER: WATSON INT'L				
20-04	710514	1	179.00	179.00
* TOTAL FOR CUSTOMER: WATSON INT'L				179.00
** GRAND TOTAL **				2607.60

SECTION 7
COMMAND FILES

It is possible to instruct RETRIEVE to take commands and/or data from a file instead of from the terminal. Such a file is called a command file; it may be created in EDITOR or in the EXECUTIVE.[1] Command files can be especially useful when a particular sequence of commands is to be executed repeatedly. For instance, in the example below, the total sales for each salesman is to be computed periodically by executing the command file SALESMEN.

When creating a command file, the commands should be typed into the file exactly as they would normally be given from the terminal. For example, the command file SALESMEN could be created in EDITOR as follows:

```
-EDITOR↵                    Commands are typed exactly as they would be given from the terminal.
*APPEND↵
PRINT SALESMAN,CUSTOMER,TOTAL FOR SALESMAN = 'BROWN'↵
SUM TOTAL FOR SALESMAN = 'BROWN'↵
PRINT SALESMAN,CUSTOMER,TOTAL FOR SALESMAN = 'FISCHER'↵
SUM TOTAL FOR SALESMAN = 'FISCHER'↵
PRINT SALESMAN,CUSTOMER,TOTAL FOR SALESMAN = 'GOLDBERG'↵
SUM TOTAL FOR SALESMAN = 'GOLDBERG'↵
PRINT SALESMAN,CUSTOMER,TOTAL FOR SALESMAN = 'GORDON'↵
SUM TOTAL FOR SALESMAN = 'GORDON'↵
PRINT SALESMAN,CUSTOMER,TOTAL FOR SALESMAN = 'ROBERTS'↵
SUM TOTAL FOR SALESMAN = 'ROBERTS'↵
COMMAND T↵
*WRITE↵                     Control D terminates the APPEND command.
TO:SALESMEN↵
 NEW FILE↵
456 CHARACTERS
*QUIT↵

-
```

If the CREATE, APPEND, CHANGE, or MODIFY command is used in a command file, RETRIEVE expects input from the command file, and *not* from the terminal. However, data may be entered from the terminal during command file execution by storing the command

APPEND FROM T

in the command file. T specifies that data be taken from the terminal instead of the command file. An example using APPEND FROM T is given on page 89.

Each command file should include as its final line the command

COMMAND T

if commands are to be taken from the terminal and not from another command file.

1 – In EDITOR, use the APPEND and WRITE commands to create a command file. In the EXECUTIVE, use the command **COPY T TO** *file name*. See the *Tymshare EDITOR* and *EXECUTIVE Reference Manuals* for details.

EXECUTING COMMAND FILES

Command files may be executed either in RETRIEVE or from the EXECUTIVE.

The DO Command

The DO command is used to execute a command file in RETRIEVE. It has the form

DO *file name*

where the file name is the name of the command file.

Example

For the data base SALES listed in part as

```
ITEM   CUSTOMER            AREA       DATE QTY  PRICE SALESMAN   TOTAL
                             ORDER

16-34 ELECTRO LABS        MB 7846 710115   1  49.80 FISCHER     49.80
16-34 GENERAL RADIATION   MB 7937 710425   3  49.80 BROWN      149.40
```

the command file SALESMEN created above could be executed as follows:

```
-RETRIEVE ↵

•BASE SALES ↵

12 RECORDS(59)

•DO SALESMEN ↵        Command file is executed.

•

SALESMAN CUSTOMER                TOTAL

BROWN    GENERAL RADIATION      149.40
BROWN    NUCLEAR SYSTEMS         29.76

2 RECORDS
```

```
•
        SUM  OF

     179.16  TOTAL      Total sales for BROWN are displayed.

2 RECORDS


•

SALESMAN CUSTOMER              TOTAL

FISCHER  ELECTRO LABS           49.80
FISCHER  SIMON IND.            537.00
FISCHER  SIMON IND             537.00
FISCHER  TRANS-MAGNETO LTD     283.95
FISCHER  TRANS-MAGNETO LTD     681.48

5 RECORDS


•
        SUM  OF

    2089.23  TOTAL

5 RECORDS


•

SALESMAN CUSTOMER              TOTAL

GOLDBERG AMER. TRANSISTOR      358.00

1 RECORDS


•
        SUM  OF

     358.00  TOTAL

1 RECORDS
```

```
•
SALESMAN CUSTOMER                TOTAL

GORDON   POTOMAC ENT             14.88

1 RECORDS


•
        SUM  OF

      14.88  TOTAL

1 RECORDS


•
SALESMAN CUSTOMER                TOTAL

ROBERTS  NUCLEAR SYSTEMS        358.00
ROBERTS  WATSON INT'L           179.00
ROBERTS  SEMICONDUCTOR INC      104.16

3 RECORDS


•
        SUM  OF

     641.16  TOTAL

3 RECORDS


•
•QUIT ↵    Execution of command file terminates; control is returned to user, who then types QUIT.
-
```

A DO command may appear at the end of a command file to transfer control of execution to a new command file or to the terminal. For example, suppose the user wished to execute the command file SALESMEN followed by a command file DISTRICTS which printed the total sales for each district. He could do this by putting the statement

DO DISTRICTS

at the end of SALESMEN and then using the command

.DO SALESMEN↵

The COMMAND Command

The command

COMMAND *file name*

is used to execute a command file from the EXECUTIVE.

Example

```
-COPY SAMPLE TO T↵

RETRIEVE
BASE SALES
APPEND
42-85,NUCLEAR SYSTEMS,MB,7984,710420,4,56.79,BROWN,0
16-34,NUCLEAR SYSTEMS,MB,7895,710420,3,49.80,BROWN,0
↵          Carriage Return necessary to terminate APPEND.
REPLACE TOTAL WITH QTY*PRICE
SORT ON ITEM,CUSTOMER
NO
FAST
QUIT
COMMAND T

-COMMAND SAMPLE↵      The command file SAMPLE is executed.

•

12 RECORDS(59)        BASE SALES is executed.

•

ITEM  CUSTOMER              AREA    DATE    QTY PRICE  SALESMAN TOTAL
                              ORDER

                      APPEND is executed; data is taken from SAMPLE.
2 RECORDS

•                     REPLACE is executed.

14 RECORDS
```

Note that the records entered must be in the command file.

• *SORT is executed.*

```
SALES'OLD' CONTAINS A COPY OF YOUR OLD DATA BASE.
SHALL WE RETAIN IT?
```

• *FAST prints the data base.*

```
16-34 ELECTRO LABS        MB 7846 720115   1   49.80 FISCHER     49.80
16-34 GENERAL RADIATION   MB 7937 720425   3   49.80 BROWN      149.40
16-34 NUCLEAR SYSTEMS     MB 7895 710420   3   49.80 BROWN      149.40
16-34 NUCLEAR SYSTEMS     MB 7805 711201   2  179.00 ROBERTS    358.00
20-04 AMER. TRANSISTOR    SB 7883 720213   2  179.00 GOLDBERG   358.00
20-04 SIMON IND           EA 7905 721201   7  179.00 FISCHER   1253.00
20-04 SIMON IND.          EA 7843 720623   3  179.00 FISCHER    537.00
42-85 NUCLEAR SYSTEMS     MB 7984 710420   4   56.79 BROWN      227.16
42-85 TRANS-MAGNETO LTD   MB 7922 720428   5   56.79 FISCHER    283.95
42-85 TRANS-MAGNETO LTD   MB 7899 720315  12   56.79 FISCHER    681.48
42-85 WATSON INT'L        MB 7940 720514   1  179.00 ROBERTS    179.00
58-78 NUCLEAR SYSTEMS     MB 7914 720327   2   14.88 BROWN       29.76
58-78 POTOMAC ENT         EA 7878 720615   1   14.88 GORDON      14.88
58-78 SEMICONDUCTOR INC   SB 7942 720801   7   14.88 ROBERTS    104.16

14 RECORDS
```

• *QUIT and COMMAND T return the user to the EXECUTIVE.*

-

ANNOTATING COMMAND FILES

The following RETRIEVE commands have been implemented to facilitate the use of command files:

Command Form	Function
HUSH	Suppresses prompt character (.) and such messages as: 5 RECORDS REPORT OUTPUT TO:
TALK	Terminates effect of HUSH.
ECHO ON	Causes the contents of a command file to appear as terminal printout just as if it were typed from the terminal.
ECHO OFF	Terminates effect of ECHO ON.
TYPE *'string of characters'*	Prints specified characters on the terminal.

The HUSH command is useful for improving the appearance of an executing command file, since it suppresses the prompt character and the usual messages printed by RETRIEVE. For example, if HUSH were included at the beginning of the command file SALESMEN created at the beginning of this section, execution of this file would appear as follows:

```
•BASE SALES ↵

14 RECORDS(59)

•DO SALESMEN ↵

•
SALESMAN CUSTOMER                TOTAL

BROWN    GENERAL RADIATION     149.40
BROWN    NUCLEAR SYSTEMS       149.40
BROWN    NUCLEAR SYSTEMS       227.16
BROWN    NUCLEAR SYSTEMS        29.76
       SUM  OF

     555.72  TOTAL

SALESMAN CUSTOMER                TOTAL

FISCHER  ELECTRO LABS            49.80
FISCHER  SIMON IND             1253.00
FISCHER  SIMON IND.             537.00
FISCHER  TRANS-MAGNETO LTD      283.95
FISCHER  TRANS-MAGNETO LTD      681.48
       SUM  OF

    2805.23  TOTAL

    .
    .        The rest of SALESMEN is executed.
    .

TALK ↵       The command file ends, but no period is printed since HUSH is still in effect. The user types
             TALK to terminate HUSH mode.
```

The ECHO ON command causes the executing command file to appear as terminal printout. Thus, for the command file SALESMEN (without a HUSH command), we have:

```
•BASE SALES ↵

14 RECORDS(59)

•ECHO ON ↵

•DO SALESMEN ↵

•PRINT SALESMAN,CUSTOMER,TOTAL FOR SALESMAN = 'BROWN'

SALESMAN CUSTOMER                TOTAL

BROWN    GENERAL RADIATION     149.40
BROWN    NUCLEAR SYSTEMS       149.40
BROWN    NUCLEAR SYSTEMS       227.16
BROWN    NUCLEAR SYSTEMS        29.76

4 RECORDS

•SUM TOTAL FOR SALESMAN = 'BROWN'

        SUM  OF

     555.72  TOTAL

4 RECORDS

•PRINT SALESMAN,CUSTOMER,TOTAL FOR SALESMAN = 'FISCHER'

SALESMAN CUSTOMER •        The DO command is aborted with an Alt Mode/Escape.
ABORTED

•
```

The TYPE command is useful for documenting the progress of command files. For example, the command

TYPE'OLD RECORDS HAVE BEEN DELETED'

in a command file causes the message

OLD RECORDS HAVE BEEN DELETED

to be printed on the terminal when the command is encountered.

EXAMPLE: USING COMMAND FILES TO COMBINE THE POWER OF RETRIEVE AND ANOTHER TYMSHARE LANGUAGE

The real power of command files lies in their ability to combine the features of more than one Tymshare language. By storing all the commands that he wishes to use in a command file, the user can effectively write a single program using more than one Tymshare language and/or applications package. The example presented here illustrates a command file which combines commands of RETRIEVE and SUPER FORTRAN.

This example presents a solution to the problem of updating an inventory data file. Suppose the user has a data base containing the items

PARTNO	The part number.
ONHAND	The number of parts on hand.
ORDERED	The number of parts ordered.
PRICE	The price of the part.
DESCR	A brief description of the part.

He might expect daily transactions of the following types:

- A number of parts sold for a given PARTNO.
- A number of parts arrived from the supplier of a given PARTNO.
- A number of parts ordered of a given PARTNO.
- A supply count revealing the actual number of parts on hand for a given PARTNO.

Hundreds of such transactions might occur each day. With such a large number of transactions, using RETRIEVE commands like CHANGE and REPLACE to update the inventory file would be costly and time consuming. However, the alternative of writing a special program to update the inventory file is usually difficult, error prone, and can be very inefficient if done carelessly.

Our example illustrates how RETRIEVE can be used to create an auxiliary data base from which the daily updates can easily and efficiently be performed with a simple program written in Tymshare SUPER FORTRAN. In the example, the inventory data base is called INVENTORY, and the transaction file is called DAILYTRANS.

The structure and initial contents of INVENTORY are illustrated below:

```
•BASE INVENTORY ↵

7 RECORDS(47)

•STRUCTURE ↵


FIELD   TYPE   WIDTH   NAME

    1   I        6     PARTNO
    2   I        6     ONHAND
    3   I        6     ORDERED
    4   N      8, 2    PRICE
    5   C       20     DESCR
```

```
•PRINT ↵

PARTNO ONHAND          PRICE DESCR
          ORDERED

013032     287     213   600.00 HAZER
214975    2804    1000     9.75 WIDGET
421675     804      52     8.49 BOLT
504088     804     388     9.09 UPPER OVERHANGER
514975      56      12    65.00 FRUZ
604088     406     930    39.72 LOWER OVERHANGER
848076    9514    2203    39.16 GADGET

7 RECORDS

•
```

DAILYTRANS has the following structure:

```
•STRUCTURE ↵

FIELD   TYPE   WIDTH   NAME

    1   I        6     PARTNO
    2   I        1     TCODE
    3   I        6     COUNT
```

The meaning of a given record in DAILYTRANS depends on the value of TCODE:

Value of TCODE	Meaning	Required Updating
1	Sale	Subtract COUNT from ONHAND.
2	Arrival from supplier	Add COUNT to ONHAND. Subtract COUNT from ORDERED.
3	Order	Add COUNT to ORDERED.
4	Inspection	Replace ONHAND by COUNT.

The daily transaction records can be entered at the end of the day. If the APPEND FROM T command is used, they can be entered during execution of the command file. The records can be entered in any order, since the RETRIEVE SORT command is used following data entry to sort them by PARTNO and TCODE. Assuming INVENTORY is also kept sorted by PARTNO, a very simple SUPER FORTRAN program can make a sequential pass through both files making the updates. The program, stored on the file UPDATE, is listed at the end of this example.

All the commands needed to perform the daily updates can be stored on a command file as follows. Note the use of HUSH, TALK, and TYPE to improve the appearance of terminal printout.

```
-COPY INVUP TO T ↵        The previous day's transactions are automatically deleted
                          with this EXECUTIVE command.
DELETE DAILYTRANS
RETRIEVE                  RETRIEVE is called.
HUSH
BASE DAILYTRANS
TYPE 'ENTER DAILY TRANSACTIONS:'
TYPE 'TYPE PARTNO,TCODE,COUNT:'
TYPE ""                   These RETRIEVE commands allow for data
APPEND FROM T             entry and sorting.
SORT BY PARTNO,TCODE
NO
TALK
QUIT
SFORTRAN          SUPER FORTRAN is called.
LOAD UPDATE       UPDATE is loaded with LOAD and executed with RUN.
RUN               Then RETRIEVE is called again.
RETRIEVE
HUSH
BASE INVENTORY
TYPE 'HERE IS YOUR UPDATED INVENTORY:'   These RETRIEVE commands load the updated
PRINT                                    data base INVENTORY and display it.
TALK
QUIT
COMMAND T

-
```

This command file includes EXECUTIVE commands, RETRIEVE commands, and SUPER FORTRAN commands. In creating such command files, the user must remember that the file must include everything that would normally be typed at the terminal.

The command file is executed as follows:

```
-COMMAND INVUP↵

•
ENTER DAILY TRANSACTIONS:
TYPE PARTNO,TCODE,COUNT:
604088,1,5↵                    The daily transactions are entered from the terminal.
421675,1,17↵
013032,1,150↵
514975,1,3↵
421675,1,8↵
848076,1,69↵
013032,1,11↵
504088,2,144↵
514975,2,144↵
214975,2,1000↵
604088,3,500↵
013032,3,200↵
514975,4,45↵
↵

•

UPDATE DONE                    Printed by the QUIT command in SFORTRAN program (line 32).

•
HERE IS YOUR UPDATED INVENTORY:
PARTNO ONHAND          PRICE DESCR
            ORDERED

 13032     126     413   600.00 HAZER
214975    3804       0     9.75 WIDGET
421675     779      52     8.49 BOLT
504088     948     244     9.09 UPPER OVERHANGER
514975      45       0    65.00 FRUZ
604088     396    1430    39.72 LOWER OVERHANGER
848076    9445    2203    39.16 GADGET

•
-
```

The SUPER FORTRAN program used to do the updating is as follows:

```
-SFORTRAN↵

>LOAD UPDATE↵
OK.
>LIST↵
  1             OPEN(2,'INVENTORY',RANDIO(47))
  2             OPEN(3,'DAILYTRANS',INPUT)
  3             INTEGER TPART,TCODE,TCOUNT
  4             INTEGER IPART,IONHAND,IORDERED
  5             REAL IPRICE
  6             STRING IDESCR(20),CR(1)
  7             *CARRIAGE RETURN CHARACTER
  8             CR=CHAR(109)
  9
 10       1     FORMAT(I6,I6,I6,F8.2,S20,S1)
 11       2     FORMAT(I6,I1,I6,I1)
 12
 13       100   READ(3,2,END=900)TPART,TCODE,TCOUNT
 14       200   READ(2,1,END=900)IPART,IONHAND,IORDERED,IPRICE,IDESCR
 15             *CHECK IF TRANSACTION AFFECTS LATER RECORD
 16       250   IF (IPART .LT. TPART) GO TO 200
 17             *CHECK IF AFFECTED RECORD ALREADY PASSED
 18             IF (IPART .GT. TPART) GO TO 800
 19             *MATCH FOUND, UPDATE CURRENT RECORD ACCORDING TO TCODE
 20             GO TO (1000,2000,3000,4000),TCODE
 21       1000  IONHAND=IONHAND-TCOUNT; GO TO 300
 22       2000  IONHAND=IONHAND+TCOUNT; IORDERED=IORDERED-TCOUNT;GO TO 300
 23       3000  IORDERED=IORDERED+TCOUNT; GO TO 300
 24       4000  IONHAND=TCOUNT; GO TO 300
 25             *OVERWRITE MOST RECENTLY READ RECORD WITH UPDATED VALUES
 26       300   WRITE(2,1)(POSITION(2)-1)IPART,
                      IONHAND,IORDERED,IPRICE,IDESCR,CR
 27             *ALLOW FURTHER UPDATES TO CURRENT RECORD
 28             READ(3,2,END=900)TPART,TCODE,TCOUNT
 29             GO TO 250
 30
 31       800   QUIT 'FILES OUT OF SORT OR BAD PARTNO'
 32       900   QUIT 'UPDATE DONE'
 33             END
>
```

SECTION 8
USER ASSISTANCE

In this section, we discuss several features which facilitate use of the RETRIEVE system. These features include the ability to correct errors while typing commands or data, utility commands which provide on-line documentation of the system, command synonyms, command abbreviations, and recovery of the unchanged data base after a MERGE, DELETE, or SORT.

Also discussed in this section are methods of creating a file that can be used as a RETRIEVE data base. Such files can be created in EDITOR, in another Tymshare language, or from paper tape.

CORRECTING ERRORS

RETRIEVE recognizes three special characters, Control A (A^C), Control W (W^C), and Control Q (Q^C), as editing characters. These may be used to correct mistakes made while typing any line, whether that line contains a command or data.

Control A deletes the preceding character typed; it may be used to delete one or more characters up to the beginning of the line being typed. On some terminals, a back arrow (←) prints each time a Control A is typed. For example,

.PRIMTA^C←A^C←NT↵

is read by RETRIEVE as PRINT.

Control Q deletes the entire line being typed and returns the carriage. On some terminals, Control Q prints an up arrow (↑). For example,

.REPLAXE PAY WITH SALARY*40+2*SALARY*Q^C↑
REPLACE PAY WITH SALARY*40+2*SALARY*(HRS-40)↵

Control W deletes the preceding word in the line being typed. The preceding word is defined as including the immediately preceding blanks, if any, plus the immediately preceding non-blank characters, up to but *not* including the first blank preceding them.

Control W may be used repeatedly to delete more than one word in the line, up to and including the first word in the line. It may not be used to delete words in preceding lines.

On some terminals, a back slash (\) is printed when Control W is used. For example,

.SORT TO CUST W^C\W^C\BY CUSTOMER,DATE↵

is interpreted by RETRIEVE as

SORT BY CUSTOMER,DATE

Control A, Control Q, and Control W can be used only on the line being typed; that is, they do not delete any characters typed before the last Carriage Return.

UTILITY COMMANDS

The following commands have been implemented to help the user at the terminal:

CAPABILITIES	Describes RETRIEVE capabilities.
CHARGES	Lists any premium charges.
CREDITS	Lists the author of the program; RETRIEVE was written by Tymshare.
HELP *or* ?	Lists all RETRIEVE commands and a brief description of each.
INSTRUCTIONS	Prints instructions on the use of RETRIEVE.
SAMPLE	Prints a sample run.
VERSION	Prints the current version number.

SYNONYMS

RETRIEVE recognizes the following synonyms for various words used in commands:

Word	Synonym
ALL TO	ALL ON
BASE	LOAD
FIELDS	ITEMS
FOR	IF
MERGE ON	MERGE BY
RESULTS TO	RESULTS ON
SAVE TO	SAVE ON
SEQUENCED	SEQUENCING
SORT ON	SORT BY

ABBREVIATING COMMANDS

All RETRIEVE commands may be shortened to as few characters as necessary to make them unique. For example,

REPLACE PAY WITH SALARY*40

can also be typed

REPL PAY WITH SALARY*40

REPORT FOR QTY>5000

may be shortened to

REPO FOR QTY>5000

FAST EMP,DEPT

may be shortened to

F EMP,DEPT

Certain individual words in a command can be shortened separately. The words FIELDS, ITEMS, SEQUENCED, and SEQUENCING may be shortened to three characters but no fewer. All others must be typed in full. For example,

APPEND FIELDS

can be typed as

APP FIE

CAUTION: Unlike command names, item names can never be abbreviated.

DATA BASE RECOVERY

To protect the user, RETRIEVE allows recovery of the old data base after it has been changed by any of the following commands:

DELETE	
MERGE	(without a RESULTS TO modifier)
SORT	(without a RESULTS TO modifier)

As a safety measure, SORT, DELETE, and MERGE create a backup file containing the unsorted or unchanged data base. The name of the backup file is obtained by appending the extension 'OLD' to the name of the data base. For example, if the data base is TAX, the backup file is TAX'OLD'.

The original data base is never disturbed by the above commands until after the message announcing base 'OLD' is printed. Hence, if the process is terminated before that point by either telephone line disconnection, Alt Mode/Escape, or Control Shift O, no recovery is necessary.

Only after the original base is successfully processed is the user given the option to erase the 'OLD' file. At this point, disconnection recovery is no longer necessary. However, the backup file may be retained to restore the data base to its former state.

To recover the last base 'OLD' file stored, the RECOVER command is used. Recovery may be required in the case of a line disconnection or it may be desirable if the user wishes to return to a previous state. For example,

```
•PRINT↵

LNAME       FNAME      ADDRESS           CITY              ZIP

BAKER       ROBERT     546 MARINA        LOS ANGELES CA    90140
HARKER      RALPH      977 ALTA WAY      BALTIMORE MD      24055
JOHNSON     JOHN       4234 LILAC LANE   BALTIMORE MD      24055
OLSON       FRANK      300 BROADWAY      NEW YORK NY       10018
PALMER      ARTHUR     147 AVENUE A      BOSTON MASS       02011

5 RECORDS
```

```
•DELETE FOR LNAME = 'JOHNSON' ↵

1 RECORDS

DIRECTORY'OLD' CONTAINS A COPY OF YOUR OLD DATA BASE.
SHALL WE RETAIN IT? YES ↵

•FAST ↵
```

The current data base does not contain the record with LNAME = 'JOHNSON'.

```
BAKER       ROBERT      546 MARINA        LOS ANGELES CA   90140
HARKER      RALPH       977 ALTA WAY      BALTIMORE MD     24055
OLSON       FRANK       300 BROADWAY      NEW YORK NY      10018
PALMER      ARTHUR      147 AVENUE A      BOSTON MASS      02011

4 RECORDS

•RECOVER ↵
```

The contents of DIRECTORY'OLD' become the current data base DIRECTORY; DIRECTORY'OLD' is deleted.

```
•FAST ↵

BAKER       ROBERT      546 MARINA        LOS ANGELES CA   90140
HARKER      RALPH       977 ALTA WAY      BALTIMORE MD     24055
JOHNSON     JOHN        4234 LILAC LANE   BALTIMORE MD     24055
OLSON       FRANK       300 BROADWAY      NEW YORK NY      10018
PALMER      ARTHUR      147 AVENUE A      BOSTON MASS      02011

5 RECORDS

•SORT BY ZIP ↵
```

The data base is rearranged by ZIP codes.

```
DIRECTORY'OLD' CONTAINS A COPY OF YOUR OLD DATA BASE.
SHALL WE RETAIN IT? NO ↵

•FAST ↵

PALMER      ARTHUR      147 AVENUE A      BOSTON MASS      02011
OLSON       FRANK       300 BROADWAY      NEW YORK NY      10018
JOHNSON     JOHN        4234 LILAC LANE   BALTIMORE MD     24055
HARKER      RALPH       977 ALTA WAY      BALTIMORE MD     24055
BAKER       ROBERT      546 MARINA        LOS ANGELES CA   90140

5 RECORDS

•RECOVER ↵
```

The DIRECTORY'OLD' file was not retained; recovery is impossible.

```
CANNOT RECOVER

•
```

CREATING DATA BASE FILES

RETRIEVE data bases can be created either from the terminal using CREATE, or from a file using BASE and/or APPEND.[1] A file used to create a RETRIEVE data base may be created from paper tape, in EDITOR, or in another Tymshare language such as SUPER BASIC or SUPER FORTRAN. It may be either a free format data file (fields separated by commas, and each record terminated by a Carriage Return) or a fixed format data file (structured according to the rules for a RETRIEVE data base given on page 37).

Creating a File From Paper Tape

A file can be created from data punched on paper tape, either off line or on line, using the EXECUTIVE TAPE program. This feature is described in the *Tymshare EXECUTIVE Reference Manual*. The following example illustrates its use:

```
-TAPE ↵          The program is called by typing TAPE in the EXECUTIVE.

:RUN ↵           The user executes the program.

INPUT FROM: T ↵  T indicates the paper tape reader.

OUTPUT TO: AFILE ↵   The data read from the paper tape will be stored in the file named AFILE.
 NEW FILE ↵

EDITING? NO ↵

TURN ON READER       Control characters are not interpreted as editing characters. The contents
                     of the tape are not printed on the terminal. The tape is read. Reading
                     terminates when the Control D at the end of the tape is encountered.
READ COMPLETE
1735 CHARACTERS WRITTEN

:
```

1 – See *Creating a Data Base From a Free Format Data File*, page 34, and *Creating a Data Base From a Fixed Format Data File*, page 37.

Creating a File in EDITOR

A file used to create a RETRIEVE data base can be created in EDITOR, Tymshare's text editing language. If this method is used, all of EDITOR's extensive text editing features discussed in the *Tymshare EDITOR Reference Manual* are available for editing the data to be stored on the file.

EDITOR may be used to create either a fixed format or free format data file. The data to be stored on the file is typed in the desired format during the EDITOR APPEND command. The data is then written on the file using the WRITE command. If a fixed format file is being created, the Line Feed option must be used.

Example

This example illustrates creating a free format data file STATUS in EDITOR, and then using STATUS to create a RETRIEVE data base named ACCOUNTING.

```
-EDITOR
*APPEND
ABC HDWARE,284A,35575,340,.18,3.05,710219
ZIMMER CO,114B,45986,95,.12,0,710427
SIM IND.,201A,87521,100,75,0,710930
COX & SONS,310B,45903,200,.24,4.19,710815
WATSON MFG,109,23546,200,.47,0,710711
*WRITE                    A Control D terminates APPEND.
TO:STATUS
 NEW FILE
195 CHARACTERS
*QUIT

-RETRIEVE

.BASE ACCOUNTING
NEW BASE, OK? YES


PLEASE TYPE IN THE STRUCTURE OF YOUR DATA BASE

FIELD  NAME,WIDTH,TYPE,DECIMAL PLACES

    1  CUSTOMER,10,C
    2  ACCT,5,C
    3  PART,5,C
    4  QTY,5,I
    5  PRICE,5,N,2
    6  SHIPPING,5,N
    7  DATE,6,I
    8

0 RECORDS(42)
```

```
•APPEND FROM STATUS ↵

5 RECORDS

•PRINT ↵

CUSTOMER    ACCT   PART     QTY PRICE          DATE
                                SHIPPING

ABC HDWARE  284A   35575    340   •18   3•05 710219
ZIMMER CO   114B   45986     95   •12      0 710427
SIM IND•    201A   87521    100 75•00      0 710930
COX & SONS  310B   45903    200   •24   4•19 710815
WATSON MFG  109    23546    200   •47      0 710711

5 RECORDS

•
```

The EDITOR command TABS, which sets tab stops at any desired position, and Control I (I^C), which spaces up to the next tab stop, are useful for creating fixed format data files in EDITOR.[1]

Creating a File in a Programming Language: Random Files

A fixed format data file to be used as a RETRIEVE data base may be created in any of the Tymshare programming languages by using fixed record length random files.[2] The data file should be structured according to the rules for a RETRIEVE data base. In particular, the file should have a record length equal to one more than the sum of the individual item lengths. The extra character is a Carriage Return which must be written at the end of each record.

Example

If the data base is to be structured as

```
NAME,10,C
JOB,5,C
SCODE,3,C
EXT,4,N
```

the data should be written on a fixed record length random file with a record length of 23 characters. The twenty-third character in each record must be a Carriage Return. Field 1 (NAME) should begin at character position 1 in each record, field 2 (JOB) at position 11, field 3 (SCODE) at position 16, and field 4 (EXT) at position 19.

1 – See the *Tymshare EDITOR Manual* for details.

2 – Consult the Tymshare manual in the language desired for further information about creating fixed record length random files.

SECTION 9
SAMPLE RETRIEVE SESSIONS

This section contains sample sessions illustrating various features of RETRIEVE.

RETRIEVING INVESTMENT INFORMATION

In this example, the user creates a data base, STOCKS, from a free format data file, STOCKDATA. By using RETRIEVE conditions, he can obtain information about stocks having any qualities which he might consider desirable for his investments.

The fields in the data base are:

Name	Contents
STOCK	Company name
PRICE	Price per share
EARN	Earnings per share
DIV	Per share dividends
NET.INC	Net income or loss
P.E.R	Price-to-earnings ratio

Notice the use of REPLACE in the example to compute the price-to-earnings ratio so that the user does not have to perform this computation himself. Notice also the power of RETRIEVE conditions to state precisely what information is desired.

The free format file used to create the data base contains the following data:

```
-COPY STOCKDATA TO T↵

BOSWELL RUBBER CO.,35,2.32,1.60,4012000,0
DYNANETICS,3.63,.00,.00,9184,0
FORBES-WILSON CORP.,3.13,.01,.00,21122,0
COSMODATA INC.,24.25,.01,.00,16256,0
WATER WORKS,14.88,.96,.80,30132932,0
WIDGETRONICS INC,4.38,.00,.00,-1992473,0
SECURITY MORTGAGE TRUST,20.63,2.12,.81,4359172,0
ALPHA MEMORY SYSTEMS,21.75,.00,.00,-932000,0
ALGERNON LASER CORP.,12.5,.00,.00,79260,0
BLIVET MFG CO.,12,.00,.00,-92948,0
GLOBAL ALUMINUM CORP.,5.88,1.08,.00,1468992,0
GALACTRONICS INC.,3.38,.00,.00,-897678,0
HEPHAESTUS INC.,6,.44,.30,872000,0
SYNTHETIC FOOD INC.,12.88,.40,.00,1212000,0
KTYM BRDCSTG,17,.28,.00,447240,0
SOLAR POWER CO.,30.63,2.96,1.6,071806780,0
```

Notice that the data records in the file can be in any order. They can be sorted after being entered into RETRIEVE, as is shown in the remainder of the example.

```
-RETRIEVE↵

•BASE STOCKS↵
NEW BASE, OK? YES↵


PLEASE TYPE IN THE STRUCTURE OF YOUR DATA BASE

FIELD  NAME,WIDTH,TYPE,DECIMAL PLACES

    1  STOCK,30,C↵
    2  PRICE,6,N,2↵
    3  EARN,5,N,2↵
    4  DIV,4,N,2↵
    5  NET.INC,8,N↵
    6  P.E.R,7,N,2↵
    7  ↵

0 RECORDS(61)

•APPEND FROM STOCKDATA↵


16 RECORDS


•SORT ON STOCK↵


STOCKS'OLD' CONTAINS A COPY OF YOUR OLD DATA BASE.
SHALL WE RETAIN IT? NO↵
```

```
•PRINT ↵
```

STOCK	PRICE	EARN	DIV	NET•INC	P•E•R
ALGERNON LASER CORP•	12•50	•00	•00	79260	•00
ALPHA MEMORY SYSTEMS	21•75	•00	•00	-932000	•00
BLIVET MFG CO•	12•00	•00	•00	-92948	•00
BOSWELL RUBBER CO•	35•00	2•32	1•60	4012000	•00
COSMODATA INC•	24•25	•01	•00	16256	•00
DYNANETICS	3•63	•00	•00	9184	•00
FORBES-WILSON CORP•	3•13	•01	•00	21122	•00
GALACTRONICS INC•	3•38	•00	•00	-897678	•00
GLOBAL ALUMINUM CORP•	5•88	1•08	•00	1468992	•00
HEPHAESTUS INC•	6•00	•44	•30	872000	•00
KTYM BRDCSTG	17•00	•28	•00	447240	•00
SECURITY MORTGAGE TRUST	20•63	2•12	•81	4359172	•00
SOLAR POWER CO•	30•63	2•96	1•60	71806780	•00
SYNTHETIC FOOD INC•	12•88	•40	•00	1212000	•00
WATER WORKS	14•88	•96	•80	30132932	•00
WIDGETRONICS INC	4•38	•00	•00	-1992473	•00

16 RECORDS

```
•REPLACE P•E•R WITH PRICE/EARN FOR EARN#0 ↵
```

Price-to-earnings ratio is computed for all companies with non-zero earnings.

10 RECORDS

```
•PRINT STOCK,PRICE,DIV,P•E•R FOR P•E•R>0 AND P•E•R<20 ↵
```

Companies with low (but non-zero) P.E.R are displayed.

STOCK	PRICE	DIV	P•E•R
BOSWELL RUBBER CO•	35•00	1•60	15•09
GLOBAL ALUMINUM CORP•	5•88	•00	5•44
HEPHAESTUS INC•	6•00	•30	13•64
SECURITY MORTGAGE TRUST	20•63	•81	9•73
SOLAR POWER CO•	30•63	1•60	10•35
WATER WORKS	14•88	•80	15•50

6 RECORDS

```
•PRINT STOCK,PRICE,NET.INC,P.E.R FOR P.E.R>80 ↵
```

Companies with high P.E.R are displayed.

```
STOCK                          PRICE  NET.INC   P.E.R

COSMODATA INC.                 24.25    16256 2425.00
FORBES-WILSON CORP.             3.13    21122  313.00

2 RECORDS
```

```
•PRINT STOCK,PRICE,DIV FOR DIV#0 ↵
```

The user wants to know which stocks are paying dividends.

```
STOCK                          PRICE  DIV

BOSWELL RUBBER CO.             35.00 1.60
HEPHAESTUS INC.                 6.00  .30
SECURITY MORTGAGE TRUST        20.63  .81
SOLAR POWER CO.                30.63 1.60
WATER WORKS                    14.88  .80

5 RECORDS
```

```
•FAST STOCK,NET.INC FOR NET.INC < 0 ↵
```

Companies with a net loss are displayed.

```
ALPHA MEMORY SYSTEMS          -932000
BLIVET MFG CO.                 -92948
GALACTRONICS INC.             -897678
WIDGETRONICS INC             -1992473

4 RECORDS
```

```
•DELETE FOR NET.INC < 0 ↵
```

Those companies are deleted from the data base.

```
4 RECORDS

STOCKS'OLD' CONTAINS A COPY OF YOUR OLD DATA BASE.
SHALL WE RETAIN IT? NO ↵

•QUIT ↵

-
```

RETRIEVING AND UPDATING SALES DATA

This example illustrates the power and simplicity of various RETRIEVE commands. In the example, an old data base called SALES is loaded into RETRIEVE, some new records are appended, and the data base is then sorted into its original order by item number. Various commands for information retrieval and updating are then illustrated.

```
-RETRIEVE ↵

•BASE SALES ↵        The old data base, SALES, is entered.

12 RECORDS(59)

•STRUCTURE ↵         Its structure is displayed.
```

FIELD	TYPE	WIDTH	NAME
1	C	5	ITEM
2	C	17	CUSTOMER
3	C	2	AREA
4	I	4	ORDER
5	I	6	DATE
6	I	3	QTY
7	N	6, 2	PRICE
8	C	8	SALESMAN
9	N	7, 2	TOTAL

```
•PRINT ↵             The data base is displayed. Note that it is sorted by ITEM and CUSTOMER.
```

ITEM	CUSTOMER	AREA ORDER		DATE	QTY	PRICE	SALESMAN	TOTAL
16-34	ELECTRO LABS	MB	7846	710125	1	49.80	FISCHER	49.80
16-34	GENERAL RADIATION	MB	7937	710915	3	49.80	BROWN	149.40
16-34	NUCLEAR SYSTEMS	MB	7805	701030	2	49.80	ROBERTS	99.60
20-04	AMER. TRANSISTOR	SB	7883	711013	2	179.00	GOLDBERG	358.00
20-04	SIMON IND	EA	7905	710724	7	179.00	FISCHER	1253.00
42-85	NUCLEAR SYSTEMS	MB	7984	710809	4	56.79	BROWN	227.16
42-85	TRANS-MAGNETO LTD	MB	7922	710615	5	56.79	FISCHER	283.95
42-85	TRANS-MAGNETO LTD	MB	7899	710315	12	56.79	FISCHER	681.48
42-85	WATSON INT'L	MB	7940	710608	1	56.79	ROBERTS	56.79
58-78	NUCLEAR SYSTEMS	MB	7914	710128	2	14.88	BROWN	29.76
58-78	POTOMAC ENT	EA	7878	711214	1	14.88	GORDON	14.88
58-78	SEMICONDUCTOR INC	SB	7942	710806	7	14.88	ROBERTS	104.16

```
12 RECORDS
```

```
•APPEND↵

ITEM  CUSTOMER          AREA    DATE    QTY PRICE  SALESMAN TOTAL
                                ORDER

42-71,AMER. TRANSISTOR,SB,7790,710815,3,12.95,ROBERTS,0↵
16-34,NUCLEAR SYSTEMS,MB,7895,710901,3,49.80,BROWN,0↵
42-85,NUCLEAR SYSTEMS,MB,7984,710915,4,56.79,BROWN,0↵
↵

3 RECORDS

•13:15 REPLACE TOTAL WITH QTY*PRICE↵

3 RECORDS

•SORT BY ITEM,CUSTOMER↵

SALES'OLD' CONTAINS A COPY OF YOUR OLD DATA BASE.
SHALL WE RETAIN IT? NO↵

•PRINT ITEM,CUSTOMER,TOTAL↵

ITEM  CUSTOMER              TOTAL

16-34 ELECTRO LABS           49.80
16-34 GENERAL RADIATION     149.40
16-34 NUCLEAR SYSTEMS        99.60
16-34 NUCLEAR SYSTEMS       149.40
20-04 AMER. TRANSISTOR      358.00
20-04 SIMON IND            1253.00
42-71 AMER. TRANSISTOR       38.85
42-85 NUCLEAR SYSTEMS       227.16
42-85 NUCLEAR SYSTEMS       227.16
42-85 TRANS-MAGNETO LTD     283.95
42-85 TRANS-MAGNETO LTD     681.48
42-85 WATSON INT'L           56.79
58-78 NUCLEAR SYSTEMS        29.76
58-78 POTOMAC ENT            14.88
58-78 SEMICONDUCTOR INC     104.16

15 RECORDS
```

The value of TOTAL is computed for the new records.

The data base is sorted to retain its original order.

The user displays selected fields for the entire data base.

```
•CHANGE PRICE FOR QTY>10 ↵
```

The user wishes to adjust the PRICE if the quantity ordered is greater than 10. He uses CHANGE to do this.

```
 PRICE :  PRICE

 56.79 :  52 ↵

1 RECORDS

•REPLACE TOTAL WITH QTY*PRICE FOR QTY>10 ↵
```

He then computes the new total for the changed price.

```
1 RECORDS

•SAVE OLDACCT FOR DATE<710000 ↵
NEW BASE, OK? YES ↵
```

All records dated before 710000 are saved on a data base named OLDACCT.

```
1 RECORDS

•DELETE FOR DATE<710000 ↵
```

The records saved above are deleted from the data base SALES.

```
1 RECORDS

SALES'OLD' CONTAINS A COPY OF YOUR OLD DATA BASE.
SHALL WE RETAIN IT? NO ↵
```

The user indicates that the unchanged data base should not be retained on SALES'OLD' for backup security.

```
•SUM TOTAL FOR SALESMAN='FISCHER' ↵
```

The SUM command prints the total sales attributable to FISCHER.

```
        SUM   OF

    2210.75   TOTAL

4 RECORDS

•COUNT FOR 'NUCLEAR' IN CUSTOMER ↵
```

The COUNT command indicates how many records have the letters NUCLEAR in the field CUSTOMER.

```
4 RECORDS
```

```
•PRINT CUSTOMER, ITEM, DATE, TOTAL FOR ORDER=7905 ↵
```

The user asks to see some of the information for order number 7905.

```
CUSTOMER              ITEM     DATE    TOTAL

SIMON IND             20-04 710724 1253.00

1 RECORDS
```

```
•SUM TOTAL ↵
```

The total sales for the entire data base are computed. The sum of all values in the item TOTAL is 3566.31.

```
          SUM  OF

      3566.31  TOTAL

14 RECORDS
```

```
•QUIT ↵

-
```

USING THE REPORT COMMAND TO GENERATE A NEW DATA BASE

This example illustrates the use of the REPORT command to generate a new data base. Using REPORT to do this allows the new data base records to contain different fields than the current data base records, as specified by the user. The new fields may contain any desired quantity that can be expressed as a RETRIEVE expression.

In this example, REPORT is used to generate a new data base, MBSALES, from the current data base SALES (used in the example on page 105). The report is written only for the sales records for which the field AREA is equal to MB. Notice that the new data base MBSALES contains only seven of the nine fields in the current data base, and that REPORT allows the user to arrange the fields in a different order. (CUSTOMER is followed by ITEM in the new data base, whereas ITEM is followed by CUSTOMER in the current data base.)

To use the file MBSALES as a data base, the user declares it with the BASE command (see Section 3). Note that when he defines the structure of the new base, he adds an extra space to all but one of the new field widths to allow for the spaces generated between each column by the REPORT command.

```
-RETRIEVE↵

•BASE SALES↵        The base SALES is loaded
                    into RETRIEVE.

14 RECORDS(59)

•STRUCTURE↵
```

FIELD	TYPE	WIDTH	NAME
1	C	5	ITEM
2	C	17	CUSTOMER
3	C	2	AREA
4	I	4	ORDER
5	I	6	DATE
6	I	3	QTY
7	N	6, 2	PRICE
8	C	8	SALESMAN
9	N	7, 2	TOTAL

```
•PRINT FOR AREA='MB'↵   The records to be used in
                        the new base are displayed.
```

ITEM	CUSTOMER	AREA	ORDER	DATE	QTY	PRICE	SALESMAN	TOTAL
16-34	ELECTRO LABS	MB	7846	710125	1	49.80	FISCHER	49.80
16-34	GENERAL RADIATION	MB	7937	710915	3	49.80	BROWN	149.40
16-34	NUCLEAR SYSTEMS	MB	7895	710901	3	49.80	BROWN	149.40
42-85	NUCLEAR SYSTEMS	MB	7984	710915	4	56.79	BROWN	227.16
42-85	NUCLEAR SYSTEMS	MB	7984	710809	4	56.79	BROWN	227.16
42-85	TRANS-MAGNETO LTD	MB	7922	710615	5	56.79	FISCHER	283.95
42-85	TRANS-MAGNETO LTD	MB	7899	710315	12	52.00	FISCHER	624.00
42-85	WATSON INT'L	MB	7940	710608	1	56.79	ROBERTS	56.79
58-78	NUCLEAR SYSTEMS	MB	7914	710128	2	14.88	BROWN	29.76

```
9 RECORDS
```

```
.REPORT FOR AREA='MB'↵
```

This REPORT command generates the new data base file.

```
REPORT OUTPUT TO: MBSALES↵
 NEW FILE↵

REPORT FORM NAME: MB↵

HEADING? NO↵
DOUBLE SPACE? NO↵
TOTALS? NO↵

COL  WIDTH,CONTENTS

1      17,CUSTOMER↵
2      5,ITEM↵
3      4,ORDER↵
4      6,DATE↵
5      3,QTY↵
6      6,PRICE↵
7      7,TOTAL↵
8      ↵

COL  NO. OF DECIMAL PLACES

3      ↵
4      ↵
5      ↵
6      2↵
7      2↵

REPORT IS 54 CHARACTERS WIDE
```

These questions must be answered NO if the report file is to be used as a data base.

No decimal places are specified for columns 3, 4, and 5.

```
•BASE MBSALES↵
NEW BASE, OK? YES↵
FIXED FORMAT? YES↵
```

The new base file is declared to be the current data base.

```
PLEASE TYPE IN THE STRUCTURE OF YOUR DATA BASE

FIELD  NAME,WIDTH,TYPE,DECIMAL PLACES

    1  CUSTOMER,18,C↵
    2  ITEM,6,C↵
    3  ORDER,4,I↵
    4  DATE,7,I↵
    5  QTY,4,I↵
    6  PRICE,7,J,2↵
TYPE SHOULD BE EITHER C,N OR I
TYPE: N,2↵
    7  TOTAL,8,N,2↵
    8  ↵

9 RECORDS(55)
```

Its structure is defined. An extra space is added to each field width except ORDER. Extra spaces are added at the end of character items and at the beginning of the numeric items. The space is added at the end of ITEM instead of the beginning of ORDER.

Note error message.

```
•SORT BY CUSTOMER,ITEM↵
```

The new base is sorted.

```
MBSALES'OLD' CONTAINS A COPY OF YOUR OLD DATA BASE.
SHALL WE RETAIN IT? NO↵

•PRINT↵
```

The new base is displayed.

CUSTOMER	ITEM	ORDER	DATE	QTY	PRICE	TOTAL
ELECTRO LABS	16-34	7846	710125	1	49.80	49.80
GENERAL RADIATION	16-34	7937	710915	3	49.80	149.40
NUCLEAR SYSTEMS	16-34	7895	710901	3	49.80	149.40
NUCLEAR SYSTEMS	42-85	7984	710809	4	56.79	227.16
NUCLEAR SYSTEMS	42-85	7984	710915	4	56.79	227.16
NUCLEAR SYSTEMS	58-78	7914	710128	2	14.88	29.76
TRANS-MAGNETO LTD	42-85	7922	710615	5	56.79	283.95
TRANS-MAGNETO LTD	42-85	7899	710315	12	52.00	624.00
WATSON INT'L	42-85	7940	710608	1	56.79	56.79

```
9 RECORDS

•QUIT↵

-
```

REPORTING PERSONNEL INFORMATION

This example illustrates using the REPORT command to extract different kinds of information from a data base and to present the information in a useful format. The data base used, PERSDATA, contains personnel information organized into the following fields:

Field Name	Contents
NAME, ADDRESS, CITY, ZIP	An employee's name, street address, city, and ZIP code
NUMBER	The employee's number
DEPT	The department number
SHIFT	A number indicating the work shift
TITLE	The employee's title
WSALARY	Weekly salary
LREVIEW	Date of last review
NREVIEW	Date of next review
LRAISE	Amount of last raise
REVIEW. RATE	A code indicating a performance rating: EXC, GD, etc.

Two useful reports are generated from this data in the example:

1. A complete company roster giving each employee's name, address, and department number.
2. A report containing information relevant to each employee's next review. Note that in this report totals are requested even though none are desired. This enables the user to specify subtotals by the item DEPT so that RETRIEVE will print the department number each time it changes. Thus, a more readable report is produced.

```
-RETRIEVE↵
```

```
•BASE PERSDATA↵
```

The data base is entered.

```
67 RECORDS(109)
```

```
•STRUCTURE↵
```

The structure of PERSDATA is displayed.

```
FIELD   TYPE  WIDTH  NAME

    1   C      20    NAME
    2   C      24    ADDRESS
    3   C      15    CITY
    4   I       5    ZIP
    5   I       4    NUMBER
    6   I       2    DEPT
    7   I       1    SHIFT
    8   C      10    TITLE
    9   N      6, 2  WSALARY
   10   I       6    LREVIEW
   11   I       6    NREVIEW
   12   N      6, 2  LRAISE
   13   C       3    REVIEW•RATE
```

```
•SORT BY NAME↵
```

The data base is sorted by NAME to produce a roster in alphabetical order.

```
PERSDATA'OLD' CONTAINS A COPY OF YOUR OLD DATA BASE.
SHALL WE RETAIN IT? NO↵
```

```
•REPORT ↵

REPORT OUTPUT TO: T ↵

REPORT FORM NAME: ROSTER ↵

HEADING? YES ↵

DOUBLE SPACE? NO ↵

TOTALS? NO ↵

COL   WIDTH,CONTENTS

1       20,NAME ↵
2       24,ADDRESS ↵
3       15,CITY ↵
4       5 DEPT ↵

INCORRECT SPECIFICATION
4       5,DEPT ↵
5       ↵

COL   HEADING

1       NAME ↵
2       ADDRESS ↵
3       CITY ↵
4       DEPT ↵

COL   NO. OF DECIMAL PLACES

4       ↵

REPORT IS 67 CHARACTERS WIDE
```

The report form will be stored on the file ROSTER'REP'.

The user types a space instead of a comma to separate the width and contents.

RETRIEVE prints an error message and allows immediate correction.

No decimal places are desired in column 4.

After printing the report width as above, RETRIEVE produces the following report.

PAGE 1
DATE: 7/18 13:27
DATA BASE: PERSDATA
REPORT FORM: ROSTER

NAME	ADDRESS	CITY	DEPT
ALBERTSEN JUNE	9632 WASHINGTON ST.	SUNNYVALE	50
BALES STEVEN	427 CHIQUITA ST.	LOS ALTOS	60
BANKS JOHN	1685 PARTNEY LN.	LOS ALTOS	40
BELLINA VERA	4369 ALMA ST.	PALO ALTO	40
BRENNAN CHARLES	1093 LOUIS CT.	HAYWARD	40
BRIDGER GLORIA	4339 CALIFORNIA ST.	PALO ALTO	60
CASTRO DAVE	9610 EL TORO DR.	SAN JOSE	60
COMPTON GEORGE	1673 LEOTA AVE.	SANJOSE	50
COOKE RALPH	734 ACALANES AVE.	LOS ALTOS HILLS	60
COPPAGE JOHN	888 SYLVAN ST.	SARATOGA	60
CRAFT ALBERT	638 FOREST AVE.	SUNNYVALE	90
DANIELS PETER	47 S. THIRD ST.	SAN JOSE	50
DONALDSON MARK	976 AFTON CT.	LOS GATOS	50
ELDER BILL	291 JEFFERSON AVE.	SAN JOSE	60
FRAZER PAUL	476 OLIVE AVE.	PALO ALTO	60
GRANT JOAN	206 S. MARTIN AVE.	MORGAN HILL	30
GREEN JEANIE	1747 PARK WY.	SAN JOSE	30
GREENE HAL	4081 DANA ST.	FREMONT	50
GUNTER REICHARD	7931 PINE AVE.	BELMONT	60
HAMILTON ELINOR	920 EVELYN AVE.	SANTA CLARA	20
HAMMER JIM	121 WASHINGTON ST.	SAN JOSE	60
HAYNES RUSSELL	8461 CEDAR ST.	MILPITAS	60
JONES CURLY	111 HYDE ST.	SAN FRANCISCO	20
JONES GEORGE	111 FORMOST ST.	SAN FRANCISCO	10
JONES HARRY	1455 LARSEN ST.	SAN JOSE	40
JORGENSEN JOY	122 GLORIA WY.	MT. VIEW	20
KELLEY LORINE	1341 EVELYN AVE.	SAN JOSE	50
KELLY JEROME	65 ROBERT LN.	CUPERTINO	30
KENYON RAY	1605 GEORGE AVE.	SAN LEANDRO	40
LARSON MIKE	609 N. J ST.	LOS GATOS	50
LEE FRED	639 SARA WY.	MENLO PARK	60
LEWIS MARTIN	823 SANOMA TR.	MT. VIEW	90
MARTIN GRANT	10 SANORA CT.	LOS ALTOS	20
MCKENZIE JOHN	1491 EDGEWOOD DR.	REDWOOD CITY	60
MITCHEL FLORENCE	629 SARA WY.	MILPITAS	30
NIXON GENE	1063 ORANGE AVE.	REDWOOD CITY	60
PARK ERMA	750 SYLVAN AVE.	MT. VIEW	20
PARKER HELEN	1212 ST. FRANCIS ST.	SAN FRANCISCO	20
PERNELL BETTY	345 FARLEY LN.	LOS ALTOS	10
ROBERTSON HARRY	567 MARY AVE.	SAN JOSE	50
ROGERS ELDON	6079 ALTA WY.	SAN CARLOS	40
SAMPSON RALPH	612 W. IOWA AVE.	MENLO PARK	50
SAMULES GEORGE	1010 SOUTH MARY AVE.	MENLO PARK	30
SHEPARD HELEN	8473 NAVATO AVE.	MT. VIEW	50
SHUMWAY DAVID	810 COLMAN AVE.	SAN JOSE	60
SIGMAN ROBERT	1779 WOODLAND AVE.	REDWOOD CITY	30
SIMPSON JOHN B.	1576 VILLA AVE.	SUNNYVALE	40
SINCLARE RALPH	111 GREENWOOD DR.	PALO ALTO	60
SLAS MARTIN	1443 BITTERN DR.	MT. VIEW	40
SMITH KEN	4721 SAN JUAN RD.	SAN FRANCISCO	60
SMITH SAM	2020 POLK ST.	SUNNYVALE CALIF	40

PAGE 2 7/18

NAME	ADDRESS	CITY	DEPT
STANGER ALFRED	1014 CARSON WY.	LOS GATOS	50
STOVER TOM	47 HARVARD ST.	NEWARK	40
SWANSON JEFFERY	804 BERNARDO AVE.	SAN JOSE	50
TAYLOR TED	1819 SUTTER AVE.	SAN CARLOS	60
THOMAS DOUGLAS	505 OLIVE AVE.	PALO ALTO	10
THOMAS MARY	361 STEWART DR.	SUNNYVALE	20
WALDMAN CLIFF	333 IDAHO AVE.	LOS GATOS	60
WALKER DORTHY	530 MORINGSIDE DR.	LOS ALTOS	20
WALTERS ANN	655 BERNARDO AVE.	SANTA CLARA	20
WEEKS HAROLD	1862 MOOREPARK WY.	REDWOOD CITY	30
WELLS STEVEN	710 CAROLINA AVE.	SAN JOSE	30
WONG ROBERT	1093 CHERRY ST.	MT. VIEW	60
WOOLWORTH JAMES	294 JUNIPER CT.	LOS ALTOS	10
WRIGHT ORVILLE	1359 LATHAM ST.	MENLO PARK	30
YATES GAIL	781 WOLF RD.	PALO ALTO	10
ZETA HELEN	27631 FIRST ST.	SAN JOSE	30

For the next report, the data base should be sorted by DEPT since subtotaling is requested for this item. The user chooses to sort by DEPT,NREVIEW so that the employees whose reviews come soonest are listed first within each department. For example,

```
•SORT BY DEPT,NREVIEW↵

PERSDATA'OLD' CONTAINS A COPY OF YOUR OLD DATA BASE.
SHALL WE RETAIN IT? NO↵

•REPORT↵


REPORT OUTPUT TO: T↵

REPORT FORM NAME: REVIEWS↵

HEADING? YES↵

DOUBLE SPACE? NO↵

TOTALS? YES↵

SUBTOTALS? YES↵

BY ITEMS: DEPT↵

SUMMARY REPORT ONLY? NO↵

COL  WIDTH,CONTENTS

1      20,NAME↵
2      6,NREVIEW↵
3      6,LREVIEW↵
4      6,REVIEW.RATE↵
5      6,LRAISE↵
6      6,WSALARY↵
7      10,TITLE↵
8      4,NUMBER↵
9      ↵

COL  HEADING

1      NAME↵
2      NEXT↵
3      LAST↵
4      RATING↵
5      LRAISE↵
6      SALARY↵
7      TITLE↵
8      EMP#↵

COL  TOTALS? ,NO. OF DECIMAL PLACES

2      N↵
3      N↵
5      N,2↵
6      N,2↵
8      N↵


REPORT IS 71 CHARACTERS WIDE
```

PAGE 1
DATE: 7/22 9:57
DATA BASE: PERSDATA
REPORT FORM: REVIEWS

NAME	NEXT	LAST	RATING	LRAISE	SALARY	TITLE	EMP#
* DEPT: 10							
YATES GAIL	700802	690802	EXC	19.50	195.00	SEC	2847
PERNELL BETTY	700908	690908	EXC	16.50	165.00	SEC.	9853
THOMAS DOUGLAS	701001	691001	EXC	51.00	510.00	ASST MGR	3763
JONES GEORGE	710304	700304	EXC	40.00	199.00	ENGINEER	3
WOOLWORTH JAMES	710726	700726	EXC	105.00	695.00	GEN MGR	5896
* DEPT: 20							
HAMILTON ELINOR	701016	691016	EXC	11.00	110.00	CLERK	4811
THOMAS MARY	701114	691114	GD	9.60	120.00	SEC	9320
PARKER HELEN	701217	691217	EXC	20.00	185.00	SEC	5529
MARTIN GRANT	710224	700224	EXC	27.00	270.00	DEPT. MGR	8884
JORGENSEN JOY	710327	700327	EXC	16.50	165.00	SEC	3449
JONES CURLY	710426	700426	EXC	25.00	250.00	SR. ENGR.	9004
WALKER DORTHY	710529	700529	EXC	10.50	105.00	CLERK	3314
WALTERS ANN	710615	700615	EXC	12.00	120.00	SEC	7993
PARK ERMA	710626	700626	EXC	16.72	167.23	SEC.	7345
* DEPT: 30							
SAMULES GEORGE	700812	690812	EXC	18.00	180.00	JR. ACCT	735
MITCHEL FLORENCE	700829	690829	EXC	24.00	240.00	ACCT	9687
ZETA HELEN	700907	690907	GD	12.80	160.00	CLERK	1262
GRANT JOAN	701206	691206	PR	13.50	135.00	CLERK	6030
SIGMAN ROBERT	710124	700124	EXC	35.00	285.99	SR. ENGR.	1242
GREEN JEANIE	710221	700221	EXC	19.00	190.00	SEC	9923
WRIGHT ORVILLE	710514	700514	EXC	27.50	275.00	DEPT. MGR	8581
WEEKS HAROLD	710519	700519	EXC	26.00	260.00	SR. ACCT	6372
KELLY JEROME	710621	700621	FR	22.80	285.00	SR. ACCT.	2968
WELLS STEVEN	710707	700707	EXC	26.00	260.00	SR. ACCT	4563
* DEPT: 40							
SMITH SAM	700916	690916	AVE	10.83	135.42	ASSEMBLER	8288
JONES HARRY	701118	691118	EXC	10.43	155.00	TECHNICIAN	20
BRENNAN CHARLES	701126	691126	FR	13.20	165.00	SR. BUYER	8834
SLAS MARTIN	701201	691201	EXC	45.00	225.00	SR. BUYER	2683
KENYON RAY	710122	700122	GD	12.00	150.00	BUYER	4068
BANKS JOHN	710125	700125	EXC	23.50	235.00	DEPT. MGR	7386
STOVER TOM	710408	700408	GD	10.80	135.00	JR. BUYER	3083
SIMPSON JOHN B.	710413	700413	AVE	20.00	245.01	ENGR.	5702
BELLINA VERA	710517	700517	EXC	16.50	165.00	SEC.	5366
ROGERS ELDON	710610	700610	EXC	18.00	180.00	SR. BUYER	3237
* DEPT: 50							
SWANSON JEFFERY	700828	690828	EXC	16.00	160.00	SR. CLERK	2805
COMPTON GEORGE	700917	690917	EXC	20.50	205.00	DEPT. MGR	2511
DONALDSON MARK	701030	691030	FR	8.80	110.00	CLERK	5912

PAGE 2 7/22

NAME	NEXT	LAST	RATING	LRAISE	SALARY	TITLE	EMP#
DANIELS PETER	701219	691219	GD	9.84	123.00	CLERK	8001
SHEPARD HELEN	701301	691301	EXC	14.00	140.00	SEC.	5144
STANGER ALFRED	710116	700116	PR	7.60	95.00	CLERK	1045
SAMPSON RALPH	710223	700223	EXC	10.00	100.00	CLERK	6389
LARSON MIKE	710321	700321	EXC	13.00	130.00	CLERK	2312
ROBERTSON HARRY	710425	700425	EXC	16.50	165.00	SR. CLERK	4566
GREENE HAL	710720	700720	PR	6.80	85.00	CLERK	9543
KELLEY LORINE	710729	700729	GD	9.60	120.00	CLERK	2867
ALBERTSEN JUNE	710730	700730	FR	10.80	135.00	SEC.	2889
* DEPT: 60							
COPPAGE JOHN	700802	690802	EXC	29.00	290.00	SR. ENGR.	4633
TAYLOR TED	700820	690820	FR	9.22	167.33	SHIPPER	5444
BRIDGER GLORIA	701013	691013	EXC	17.50	175.00	SEC	4074
CASTRO DAVE	701025	691025	EXC	19.50	195.00	SR. TECH.	4118
LEE FRED	701108	691108	EXC	41.00	410.00	STAFF SCI	5622
WONG ROBERT	701128	691128	EXC	25.00	250.00	ENGR.	1843
SINCLARE RALPH	701228	691228	GD	15.61	195.09	BUYER	5950
COOKE RALPH	710127	700127	EXC	48.00	480.00	DEPT. MGR	6813
SHUMWAY DAVID	710129	700129	EXC	23.00	295.00	SR. ENGR.	2649
BALES STEVEN	710202	700202	EXC	38.50	385.00	STAFF SCI	1034
ELDER BILL	710215	700215	FR	12.00	150.00	TECH.	6383
MCKENZIE JOHN	710227	700227	EXC	18.50	185.00	TECHNICIAN	9605
WALDMAN CLIFF	710309	700309	GD	12.80	160.00	TECH.	9435
GUNTER REICHARD	710427	700427	EXC	23.00	230.00	SR. TECH.	4164
FRAZER PAUL	710518	700518	EXC	27.00	270.00	SR. ENGR.	5200
NIXON GENE	710520	700520	FR	15.20	190.00	SR. TECH.	1941
HAMMER JIM	710530	700530	EXC	20.50	205.00	SR. TECH.	5766
HAYNES RUSSELL	710620	700620	GD	12.80	160.00	TECH.	7227
SMITH KEN	710628	700628	EXC	20.00	200.00	SR. TECH.	4673
* DEPT: 90							
LEWIS MARTIN	701122	691122	EXC	30.00	360.00	DEPT. MGR	4163
CRAFT ALBERT	710106	700106	EXC	16.70	167.00	ASSEMBLER	1487

MONTHLY SALES REPORT

A report containing three subtotal levels is illustrated in this example. The structure of the data base SALESFILE is given by:

```
•STRUCTURE ↵

FIELD    TYPE    WIDTH    NAME
    1      C       10     REGION
    2      C       15     DISTRICT
    3      C       10     SALESMAN
    4      C       10     CITY
    5      N       10, 2  SALES1
    6      N       10, 2  SALES2
    7      N       10, 2  SALES3
    8      N       10, 2  SALESLM
```

Sales in dollars of products 1, 2, and 3.

Total sales of products 1, 2, and 3 for previous month.

The user wishes to produce a monthly report showing the total sales of each product (1, 2, and 3), totaled for each region, district, and salesman. He also wishes to show the total sales of all products for each region, district, and salesman, and the percent change in sales from the previous month for each data record. To do this, he defines a report form named SALESREPORT as follows:

```
.REPORT ↵

REPORT OUTPUT TO: T ↵

REPORT FORM NAME: SALESREPORT ↵

HEADING? YES ↵

DOUBLE SPACE? NO ↵

TOTALS? YES ↵

SUBTOTALS? YES ↵

BY ITEMS: REGION,DISTRICT,SALESMAN ↵        Three subtotal levels
                                            are specified.
SUMMARY REPORT ONLY? NO ↵

COL  WIDTH,CONTENTS

1      11,CITY ↵
2      10,SALES1 ↵
3      10,SALES2 ↵
4      10,SALES3 ↵
5      12,SALES1+SALES2+SALES3 ↵
6      13,((SALES1+SALES2+SALES3-SALESLM)/SALESLM)*100 ↵
7      ↵

COL  HEADING

1      CITY ↵
2      SALES/PRODUCT 1 ↵
3      SALES/PRODUCT 2 ↵
4      SALES/PRODUCT 3 ↵
5      TOTAL/SALES/THIS MONTH ↵
6      MONTHLY/CHANGE IN/SALES (%) ↵

COL  TOTALS? ,NO. OF DECIMAL PLACES

2      YES,2 ↵
3      YES,2 ↵
4      YES,2 ↵
5      YES,2 ↵
6      NO,2 ↵


REPORT IS 71 CHARACTERS WIDE
```

The form specified above is stored on the file SALESREPORT'REP' and, thus, can be used once a month without redescribing the structure as in the following.

```
-RETRIEVE↵

•BASE SALESFILE↵

40 RECORDS(86)

•SORT BY REGION,DISTRICT,SALESMAN,CITY↵

SALESFILE'OLD' CONTAINS A COPY OF YOUR OLD DATA BASE.
SHALL WE RETAIN IT? NO↵

•REPORT↵

REPORT OUTPUT TO: T↵

REPORT FORM NAME: SALESREPORT↵

UPDATE REPORT FORM? NO↵
```

The data base is sorted by the fields specified as subtotal levels, as well as by city. Sorting the transactions for each salesman by city causes them to be displayed alphabetically by city in the report.

This question is asked only if the report form is old. Since the answer here is NO, the report is produced immediately.

PAGE 1
DATE: 7/19 13:36
DATA BASE: SALESFILE
REPORT FORM: SALESREPORT

CITY	SALES PRODUCT 1	SALES PRODUCT 2	SALES PRODUCT 3	TOTAL SALES THIS MONTH	MONTHLY CHANGE IN SALES (%)
*** REGION: WESTERN					
** DISTRICT: OAKLAND					
* SALESMAN: ANDERSON					
ALAMEDA	101.47	100.54	131.10	333.11	9.87
BERKELEY	201.45	120.95	111.24	433.64	15.66
HAYWARD	322.56	140.70	195.42	658.68	3.99
OAKLAND	121.56	147.39	123.57	392.52	18.56
RICHMOND	300.51	160.94	134.21	595.66	11.73
SAN LEAN.	320.91	175.83	98.36	595.10	7.21
WALNUT CR.	450.73	78.32	199.91	728.96	16.61
* TOTAL FOR SALESMAN: ANDERSON	1819.19	924.67	993.81	3737.67	
* SALESMAN: WHITE					
FREMONT	444.44	233.36	92.91	770.71	1.23
LIVERMORE	475.50	11.13	64.57	551.20	.04
MANTECA	310.60	91.89	45.54	448.03	4.99
NEWARK	497.22	134.56	103.45	735.23	-7.29
STOCKTON	319.19	257.86	179.53	756.58	9.37
TRACY	265.29	234.15	152.71	652.15	8.01
* TOTAL FOR SALESMAN: WHITE	2312.24	962.95	638.71	3913.90	
** TOTAL FOR DISTRICT: OAKLAND	4131.43	1887.62	1632.52	7651.57	
** DISTRICT: SAN FRANCISCO					
* SALESMAN: BROWN					
DALY CITY	151.23	310.50	96.40	558.13	5.32
MILLBRAE	121.18	150.16	47.20	318.54	7.21
PACIFICA	15.20	19.20	10.19	44.59	10.51
S.F.	792.50	676.40	211.19	1680.09	33.78
SAN BRUNO	36.30	50.40	25.23	111.93	3.22
SAN CARLOS	96.92	137.20	81.63	315.75	1.51
SAN MATEO	87.30	110.12	71.20	268.62	.23
SOUTH S.F.	620.10	732.50	463.40	1816.00	15.67
* TOTAL FOR SALESMAN: BROWN	1920.73	2186.48	1006.44	5113.65	
* SALESMAN: CARSON					
LOS ALTOS	21.19	35.00	19.30	75.49	-4.58
MENLO PARK	35.15	61.20	30.10	126.45	-2.67

PAGE 2 7/19

CITY	SALES PRODUCT 1	SALES PRODUCT 2	SALES PRODUCT 3	TOTAL SALES THIS MONTH	MONTHLY CHANGE IN SALES (%)
MT. VIEW	23.16	35.11	20.14	78.41	3.44
PALO ALTO	170.60	160.20	120.14	450.94	-5.32
RED. CITY	97.30	85.19	70.30	252.79	4.90
STANFORD	110.00	75.00	60.20	245.20	6.78
* TOTAL FOR SALESMAN: CARSON					
	457.40	451.70	320.18	1229.28	
** TOTAL FOR DISTRICT: SAN FRANCISCO					
	2378.13	2638.18	1326.62	6342.93	
** DISTRICT: SAN JOSE					
* SALESMAN: RICHARDSON					
CAMPBELL	108.19	59.69	54.11	221.99	2.56
CUPERTINO	150.13	222.15	90.13	462.41	-3.00
LOS GATOS	163.33	166.29	100.20	429.82	7.91
SAN JOSE	440.44	50.30	120.56	611.30	10.43
SANTA CL.	139.49	70.23	91.13	300.85	-2.95
SUNNYVALE	500.30	110.30	100.20	710.80	-8.22
* TOTAL FOR SALESMAN: RICHARDSON					
	1501.88	678.96	556.33	2737.17	
* SALESMAN: ROBERTS					
GILROY	134.45	54.20	57.11	245.76	3.44
HOLLISTER	175.47	79.25	64.12	318.84	4.22
MILPITAS	155.49	60.27	78.44	294.20	-2.99
MONTEREY	210.34	81.30	89.56	381.20	5.23
MORGAN H.	43.12	95.11	91.78	230.01	2.67
SALINAS	444.50	25.69	111.10	581.29	8.31
SANTA CRUZ	463.54	26.79	121.20	611.53	4.11
* TOTAL FOR SALESMAN: ROBERTS					
	1626.91	422.61	613.31	2662.83	
** TOTAL FOR DISTRICT: SAN JOSE					
	3128.79	1101.57	1169.64	5400.00	
*** TOTAL FOR REGION: WESTERN					
	9638.35	5627.37	4128.78	19394.50	
** GRAND TOTAL **					
	9638.35	5627.37	4128.78	19394.50	

SECTION 10
COMMAND SUMMARY

DEFINITIONS AND CONVENTIONS

Brackets ([]) indicate optional parts of a command form.

range list — A list of one or more record number and/or range addresses, separated by commas, for example,

10:20,700,800:900

field list — A list of one or more field names separated by commas, for example,

EMPLOYEE,PAY

expression list — A list of one or more expressions separated by commas, for example,

QTY,PRICE,QTY*PRICE

condition — Used to specify records for which a command is to apply. Always preceded by FOR.

Examples

FOR PRICE>500

FOR 'SEC' IN JOB

FOR NAME='GOLDSMITH' OR REGION='EA'

FOR DATE<700000 AND PARTNO=848076

CREATING A NEW DATA BASE

The following table summarizes the RETRIEVE commands for creating a new data base. The word *type* refers to the type of the data base file being created, which may be SYMBOLIC, BINARY, or SCRAMBLED. If no type is specified, SYMBOLIC is assumed for CREATE and BASE; the type of the current base is used for SAVE, RESULTS TO, and ALL TO.

Method	Command Form	Description
From the terminal	**CREATE** *file name* [*type*]	Allows entry of new data base structure and data records from terminal. Data is stored on *file name*; the structure is stored on *file name* '**STR**.*x*', where *x* is the letter of the current version of RETRIEVE.
	BASE *file name* [*type*] ⋮ (Structure is entered.) **APPEND** ⋮ (Data is entered.)	Equivalent to above form of CREATE. BASE allows structure entry; APPEND, data entry.
From a file	**BASE** *new base name* [*type*] ⋮ (Structure is entered.) **APPEND FROM** *free format file name*	Creates new data base containing the records on the free format file specified in APPEND. The base name must be different from the free format file name.
	BASE *new base name* [*type*] ⋮ (Structure is entered.) **APPEND FROM** *fixed format file name* **FIXED**	Same as above except data on APPEND file must be in fixed format.
	BASE *fixed format file name*	Declares specified fixed format file to be the new data base. Requests structure, then loads data from file automatically. No type may be specified; the existing file type is assumed.
From the current data base using SAVE	**SAVE TO** *file name* [*type*]	Saves current data base on a file with the specified name and type.
	[*range list*] **SAVE TO** *file name* [*type*] [**FOR** *condition*]	Saves records specified by range list and/or condition on specified file.

Method	Command Form	Description
From the current data base using the RESULTS TO and ALL TO modifiers. *NOTE: RESULTS TO and ALL TO leave the current data base unchanged by the commands with which they are used.*	**APPEND...RESULTS TO** *file name* [*type*]	Creates new data base with name and type specified, containing current data base plus records appended.
	MERGE ON *field list* **FROM** *fixed format file name* **RESULTS TO** *base name* [*type*]	Creates new data base containing current data base records merged with the records on the specified fixed format file.
	CHANGE *or* **MODIFY ...RESULTS TO** *file name* [*type*] *or* **REPLACE**	Creates new data base containing only the records changed by the specified command.
	CHANGE *or* **MODIFY ...ALL TO** *file name* [*type*] *or* **REPLACE**	Creates new data base containing both changed and unchanged records.
	SORT...RESULTS TO *file name* [*type*]	Creates new data base containing sorted data records. (Current data base remains unsorted.)

LOADING A PREVIOUSLY CREATED DATA BASE

Command Form	Description
BASE *file name* *or* LOAD *file name*	BASE and LOAD are equivalent. They both load into RETRIEVE the base on the specified file, assuming the appropriate structure file is present.

ADDING RECORDS TO A DATA BASE

The APPEND and MERGE commands, summarized in the table below, are used to add records to the current data base. APPEND simply adds the records to the end of the data base; MERGE merges them by fields specified in the command. Both APPEND and MERGE may be used with RESULTS TO, as was summarized in *Creating a New Data Base*, above.

Command	Form	Description
APPEND	**APPEND**	Adds records typed at terminal to current data base. (If used in command file, records are entered from command file.) Each record must be terminated by a Carriage Return; fields are separated by commas.
	APPEND FROM T	Used in a command file to specify that records be appended from the terminal instead of the command file.
	APPEND FROM *file name*	Adds records to the current data base from the specified file. The data on the file must be in free format, unless the specified file is a data base (that is, has a structure file).
	APPEND FROM *file name* **FIXED**	Adds records from the specified file to the current data base. The data on the file must be in fixed format.
	APPEND FIELDS	Allows appending of records field-by-field from the terminal. Each field typed is terminated by a Carriage Return. Commas and quote marks may be included in a field without surrounding it with quote marks.
	APPEND FIELDS FROM *file name*	Adds records from the specified file to the current data base, where each field on the file is terminated by a Carriage Return.
MERGE	**MERGE [ON** *field list*] **FROM** *fixed format file name*	Merges records on the specified fixed format file into the current data base. Merging is done by the fields specified in the field list. Both the data base and the file being merged must be sorted by these fields.

COMMANDS WHICH DESCRIBE THE DATA BASE

Command	Description
STRUCTURE	Prints a description of the structure of the current data base.
SIZE	Prints the number of records in the current data base.

RETRIEVING INFORMATION IN THE DATA BASE

Command Form	Description
[*range list*] **LIST** [*field list*] [**FOR** *condition*]	Displays specified fields from selected records with record numbers and headings. If field list is omitted, entire records are displayed. If range list and condition are omitted, the selected fields are displayed for the entire data base.
[*range list*] **PRINT** [*field list*] [**FOR** *condition*]	Same as LIST except record numbers are not printed. (Headings are printed.)
[*range list*] **FAST** [*field list*] [**FOR** *condition*]	Same as LIST except neither headings nor record numbers are printed.
[*range list*] **COUNT** [**FOR** *condition*]	Displays the number of records (in the range list, if specified) satisfying the given condition. COUNT alone is equivalent to SIZE.
[*range list*] **SUM** [*expression list*] [**FOR** *condition*]	Displays totals for all expressions in expression list, computed for the records specified by range list and condition. The expressions must be numeric. If no expressions are specified, totals for all numeric fields in the data base are displayed, computed for the records specified, or for the entire data base if none are specified.
[*range list*] **AVERAGE** [*expression list*] [**FOR** *condition*]	Displays average values for all expressions in expression list, computed for the records specified by range list and condition. All expressions in list must be numeric. Options are the same as for SUM.

UPDATING THE DATA BASE

Of the commands summarized in the table below, CHANGE, MODIFY, and REPLACE may be used with RESULTS TO and ALL TO to create new data bases; SORT may be used with RESULTS TO. These forms are summarized in *Creating a New Data Base* on page 125.

Command Form	Description
[*range list*] **CHANGE** [*field list*] [**FOR** *condition*]	Allows changing of selected records (if no field list is present) or selected fields within a record. Prompts user with old field values.
[*range list*] **MODIFY** [*field list*] [**FOR** *condition*]	Same as CHANGE except old field values are not printed.
[*range list*] **DELETE** [**FOR** *condition*]	Deletes records specified by range list and/or condition.
[*range list*] **REPLACE** $field_1$ **WITH** $expression_1$ [, $field_2$ **WITH** $expression_2$,...] [**FOR** *condition*]	Replaces specified fields with values of corresponding expressions for all selected records.
SORT [**ON** *field list*]	Sorts the data base by the fields specified in field list, with the first item of prime importance.

REPORT GENERATION

The REPORT command allows the user to describe a report format and then prints the report described. Its general form is:

[*range list*] REPORT [FOR *condition*]

SUMMARY OF REPORT DESCRIPTION DIALOGUE	
RETRIEVE Prompt	**User Response**[1]
REPORT OUTPUT TO:	T *or file name*
REPORT FORM NAME:	*name or* ↵
UPDATE REPORT FORM?	YES *or* NO
HEADING?	YES *or* NO
DOUBLE SPACE?	YES *or* NO
TOTALS?	YES *or* NO
SUBTOTALS?	YES *or* NO
BY ITEM:	*field list*
SUMMARY REPORT ONLY?	YES *or* NO
COL WIDTH,CONTENTS 1 ⋮	*column width, expression* (for each column desired)
COL HEADING 1 ⋮	*heading* (for each column)
COL TOTALS? ,NO. OF DECIMAL PLACES *column number* ⋮ (prompted for each numeric column)	NO *or* YES [, *no. of decimal places*]
COL NO. OF DECIMAL PLACES *column number* ⋮ (prompted for each numeric column)	*no. of decimal places* *or* ↵

1 – All user responses must be terminated by a Carriage Return.

Notes

REPORT FORM NAME:	If a name is specified, the report form is stored on a file called *name*'REP'.
UPDATE REPORT FORM:	Asked only if the file *name*'REP' was created in an earlier REPORT command. If answered NO, all subsequent questions are skipped.
SUBTOTALS?	Asked only if TOTALS? is answered YES.
BY ITEM:	Prompted only if SUBTOTALS? is answered YES.
COL HEADING	Prompted only if HEADING? is answered YES.
COL TOTALS? ,NO. OF DECIMAL PLACES	Prompted if TOTALS is answered YES.
COL NO. OF DECIMAL PLACES	Prompted instead of above if TOTALS is answered NO.

COMMAND FILES

Command Form	Description
DO *file name*	Executes command file from RETRIEVE.
COMMAND *file name*	Executes command file from EXECUTIVE.
HUSH	Suppresses prompt character (.) and such messages as **5 RECORDS** **REPORT OUTPUT TO:**
TALK	Terminates effect of HUSH.
ECHO ON	Causes executing command file commands to appear as terminal printout.
ECHO OFF	Terminates effect of ECHO ON.
TYPE *'string of characters'*	Prints specified characters at the terminal.

UTILITY COMMANDS

Command	Description
HELP *or* ?	Lists all RETRIEVE commands and a brief description of each.
INSTRUCTIONS	Prints detailed instructions on using RETRIEVE.
QUIT	Returns the user to the EXECUTIVE.

Other utility commands are given on page 94.

INDEX

NOTE: Page numbers which appear in **bold** *face type refer to those pages where the listed item receives the most detailed discussion.*

USER EVALUATION

Tymshare would like to improve the quality and usefulness of its publications. However, to achieve this goal, we need your help and critical evaluations. Will you please provide us with such constructive information by filling out this questionnaire and mailing it back to us?

1. (a) Is this manual a useful document? ☐ Yes ☐ No

   (b) If your answer is Yes, what features make it useful.

   (c) If your answer is No, what features prevent it from being a useful document.

2. (a) Is the text clear and readily understandable? ☐ Yes ☐ No

   (b) If your answer is No, please cite the sections, subsections, or paragraphs that are unclear or difficult to understand.

3. (a) Are you pleased with the organization of this manual? ☐ Yes ☐ No

   (b) Should the organization be changed? ☐ Yes ☐ No

   (c) What changes do you suggest?

4. (a) Are the example problems helpful and easy to understand? ☐ Yes ☐ No

   (b) Should more examples be added when this manual is revised? ☐ Yes ☐ No

   (c) What kind of programs would you like to see added?

5. (a) Should anything be deleted from this manual when it is revised? ☐ Yes ☐ No

   (b) If your answer is Yes, give us your suggestions.

To remove, please cut along this line.

6. List any further suggestions you have for the improvement of this manual.

OPTIONAL INFORMATION:

Name _____ Company _____

Street Address _____

City _____ State _____ Zip _____

Occupation _____

THANK YOU FOR YOUR ASSISTANCE.

To mail this questionnaire, simply cut from manual, fold Part A back, fold Part B back, and staple where indicated. **Part A**

BUSINESS REPLY MAIL
NO POSTAGE STAMP NECESSARY IF MAILED IN U.S.A.

First Class
Permit No.
300
Los Altos,
California

POSTAGE WILL BE PAID BY

TYMSHARE, INC.
10340 BUBB ROAD
CUPERTINO, CALIFORNIA 95014
ATTN: Documentation Group

Part B

STAPLE HERE