Total Information Management

TIM® III

VICTOR®

User's Manual

April 1983
User's Manual Revision V-1

Proprietary Notice

Information contained within this manual as well as any accompanying software, either recorded or in listed format may not be duplicated by any person without the prior written consent of **Innovative Software, Inc.**

Informix and
Innovative Software
9875 Widmar
Lenaxa
Kansas 66215

Innovative Software®

9300 W. 110th St., Suite 380
Overland Park, KS 66212 USA
913/~~383-1089~~ 492-3800
Telex: 209542

Table of Contents
Reference Manual

Preliminary

T.I.M. commands

1.0 Add/inspect/update a record

2.0 Create a file

8.0 file Maintenance

9.0 Report generation

10.0 Select records from a file

11.0 Utility commands

12.0 eXit to operating system

Appendices

Primer

Example 1: Mailing list application

Example 2: Patient records and payment files

Example 3: Sales commissions file

Example 4: Search firm applicant records

Quickies to know

In order to understand this manual and T.I.M. like a real pro, there are several definitions you should know.

Prompt—A prompt is merely a question asked by the computer. This question usually requires a response by the user. For example, "Enter command" is a prompt. **Query** is another name for a prompt.

Default—On many questions that T.I.M. asks, there will be a value that T.I.M. assumes if you do not enter anything. This is called the dafault. It is usually listed after the prompt in the form (cr=default) where the 'cr' means press the ◄┘, (enter) key.

Cursor—The cursor is the little white line which shows where on the screen the next character will be printed.

Menu—A menu is a list of commands for you to pick from. These commands will be listed on the screen and you will pick which command you want just like picking a meal from a restaurant menu.

Filename—The filename is the name of a file you wish to operate on. A filename can be up to 8 characters long consisting of letters and digits (as long as a digit is not the first character of the name).

Truncation—A Truncation length is used during the report and list definition stages. It tells how long a field should be. For example, if a field is 60 characters long but you only want to print 40 characters, you would specify a truncation length of 40 and the field will be cut off at 40 characters.

Enhancements in 3.30

Report

In REPORT you now have the option to have averages printed with your totals. When defining a report, if you have chosen a field to be totaled, you will be asked whether or not you want to print an average with your totals. If you answer yes, any place on the report where you have a total, an average will appear under it.

Also in REPORT, the main report title will be printed on every page. (In prior versions the title only printed on the first page).

Maintenance

There is a new command, L-List and count records per key, on the maintenance sub-menu. For a given key field, this command will list all the unique keys and the number of records with each key. You will be asked the filename, and the key field to report by. You will then be given the option to send the report to the terminal or a printer.

Enhancements in 3.22

1. Dates of 99/99/99 (default values) will be printed as spaces in reports.

2. In linked file reports the lines specified to be printed after a break will now be printed after the detail file records, instead of after the main file records.

3. In report you may now specify a truncation length that is greater than the field length. The extra space will be placed between the printed fields.

4. In lists, numbers are now right justified.

The program disk also contains another program, TIMLIBFX.EXE, which is used to compress T.I.M. libraries (REPORT.LIB, LIST.LIB, SELECT.LIB). when you erase formats from these libraries, the formats are removed, but the amount of disk space allocated to the library remains the same. If you wish to free possible wasted space on your libraries, you should run TIMLIBFX. To execute TIMLIBFX, place the program disk in drive A, and then type "TIMLIBFX" when you have an A prompt. The program will then ask you which of the three libraries you wish to compress, the drive it is on, and the drive the new compressed library is to go on. The new library must go on a different disk than the original. The program will then write a compressed library onto the disk in the chosen drive. You may then copy this new compressed library onto your program disk.

ADDENDUM

Double-sided and hard disk operation:

T.I.M. will run on all three Victor 9000 configurations. T.I.M.'s programs for the Victor 9000 are shipped on one single-sided disk with one single-sided example files disk.

Double-sided configuration:

To set up your version of T.I.M. on to a double-sided system, you have to first format a blank diskette double-sided. Once this is completed, you will run your *Copy* command and transfer all the files from the MASTER single-sided T.I.M. Program disk onto your new double-sided formatted disk. Do not run a **dcopy**. This will only give you another single-sided version of T.I.M. You may repeat this procedure if you wish to put your example files onto a double-sided disk also.

Hard disk configuration:

To run T.I.M. on a hard disk unit you only have to copy all your files from your single-sided disk onto your hard disk.

0.1 T.I.M. Program overview

This section gives general information about T.I.M. and its various requirements.

0.1.1 System requirements

T.I.M. is a data base management program which gives the user virtually unlimited freedom in the manipulation of his data base. T.I.M. requires two disk drives except in a few exceptional cases. System requirements other than the disk drives are 128K of RAM.

0.1.2 General information

Any number of data files may be created, each of which may contain up to 32,767 records, consisting of a maximum of 40 separate data fields. The maximum field length is 60 characters and the combination of all fields may not exceed 2400 characters.

To ease record selection as well as list and report generation, T.I.M. employs several libraries. These libraries are used to save various formats and eliminate redundant entries. For example, the user may have 10 different report formats in his Report Library. He may then view these print formats and choose which one to use without having to re-define it.

The number of T.I.M. data files is limited only by the operating system and the amount of available disk space. Disk space is allocated to T.I.M. files dynamically and may be approximated using the following formula:

$$\text{Records per disk} = [(\text{disk storage})—8000][\text{record length} + (2 \times \text{KF})]$$

where the disk storage and record length are measured in bytes, and KF is the number of key-fields.

To input, massage, or output any part of the data base, the user may implement one of the T.I.M. commands listed on the displayed menu. Each of these commands, in turn, employ a number of sub-commands which allow the user total flexibility of his data base. Each command is menu-driven, so when the user wishes to use the *Report generation* command, for example, he simply enters an **R**.

0.2 T.I.M.'s database management system

This section gives a short overview of the T.I.M. system and database systems in general.

0.2.1 Database definition

A database is a collection of information; whether it be names and addresses, inventory and stock numbers, customers and their purchases, or any combination of the above. T.I.M. (Total Information Management) was created to help the user manage his own unique database effectively. By implementing T.I.M. the user is able to inspect, add or delete, as well as output any piece or all of the information in his database.

0.2.2 Files and fields

The user might have several major groups of information which comprise his database, such as a list of all his customer's names, addresses, product purchases and purchase prices. Each of these major groups of data would comprise a data "file". The only prerequisite to working with a particular file is having that file created in advance. This file could contain up to 40 separate sub-groups of information called "fields". To illustrate this, let's use the example of a file containing the employee records and salaries. The user would create and define a file according to his specific needs. For instance, field #1 of the 40 possible fields used within a T.I.M. data file might be the employee's name; field #2, the employee's street address; field #3, the employee's social security number; field #4, the employee's phone number, field #5, the employee's present salary; and so on. When defining this file and subsequent field titles (i.e. NAME, ADDRESS, SOC. SEC. #, PHONE #, SALARY,...) *the user must also specify the length and type of each field used. A typical file is shown later in Example E-1.*

0.2.2.1 Field lengths

The length of each field can vary depending on the user's needs, but it must be at least 1 character and not more than 60 characters long. For instance, the field containing the employee's street address would probably be defined as much larger than the field containing the employee's 9-digit social security number. T.I.M. can accomodate up to 40 separate fields per file, each up to 60 characters in length. While defining each field, the field type must also be defined.

0.2.2.2 Field types

The eight field types and their abbreviations are listed below:

A	=	Alphanumeric tield
C	=	Calculated field
D	=	Date field
I	=	Inverted name field
N	=	Number field
S	=	Sequential field
T	=	Total field
$	=	Dollar amount field

The definitions of these field types are as follows:

ALPHANUMERIC FIELD: a field whose entries may be any combination of alphanumeric characters, hyphens, slashes, periods or commas, etc., but may not include an asterisk (*) as the first character in the field. Examples of this are street addresses, names, social security numbers that include hypens, etc.

CALCULATED FIELD; a field whose entries are calculated by the program rather than entered by the user. The calculation used is defined during file creation and may consist of a field or a constant as an operand and the operators +, −,*, and /. A calculated field might be used to store the total of two fields, one being the quantity purchased, and the other being the price per item. Up to 20 calculated fields may be defined per file.

DATE FIELD: a field whose entries are numerical dates, entered in the normal month/day/year or day/month/year format. Preceding zeros are not required for date entry. If a file is sorted by a date field, T.I.M. will sort by year, then month, then day of the month.

INVERTED NAME FIELD; a field whose entries are proper names, but invisible to the user, T.I.M. inverts normal first-last name entries into a last-first format so these names may be sorted and printed in order of last name. All output generated containing this field type will be re-inverted so as to be identical to the input entry. It should be noted that T.I.M. determines the last name by looking for the first space from the right. Therefore, the name "Tom Mc Daniels" must be entered as "Tom McDaniels", to allow the program to sort correctly. This method allows middle initials to be entered if desired.

NUMBER FIELD: a field whose entries are numbers (either real or integer), such as quantity, number sold, etc. It should be noted that number fields may not include any characters other than numerals and appropriate decimal points and up to 16 digits may be entered. Decimal precision is determined during file creation and may range from 0 to 4 decimal places.

SEQUENTIAL FIELD: a field whose entries are automatically entered by the computer. As each new record is added to a file, the sequential field is incremented by 1. This type of field is very useful for assigning consecutive I.D. numbers or customer numbers.

TOTAL FIELD: a field whose entry is automatically calculated by T.I.M. The calculation is always addition and may total up to 16 other fields in a record. This feature is handy for summing month entries for a year total, or summing departmental figures for a company total. Up to 3 total fields may be included in each record.

DOLLAR FIELD: a field whose contents may be entered as decimal numbers, but upon printing are formatted in a dollar and cents fashion. Report totals will be printed with a dollar sign ($) preceding the amount, commas inserted for every three decimal places, and a decimal point inserted to denote the cents places (e.g. $1,000,000.00 or $1,784.25). When defining the field length. This will allow a neatly formatted output during report generation with 2 points of decimal precision.

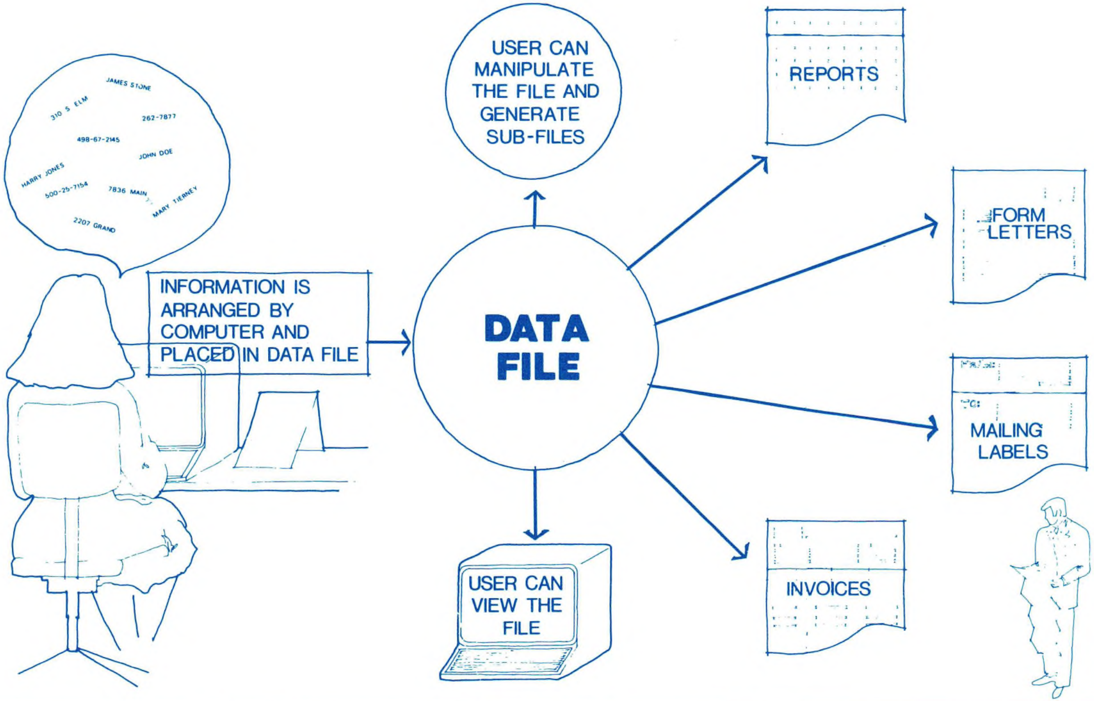
0.2.3 Concept of a record

After the file has been created, the user will want to enter information into the file. At this point, the concept of a data 'record' must be introduced. A record is a set of related fields within a file. Continuing with the previous example, a record might contain the name, address, social security number, phone number, etc. of one given employee. A pictorial view of the example file is shown in example E-1.

0.2.4 Concept of a key-field

When a file is ordered by a certain field, that field is called a key-field. An example of this would be a file of customers ordered by customer name. Reports and mailing labels could then be printed in order of customer name. Similarly, the file might also be ordered by zip code or another field, which would then be a key-field.

INFORMATION FLOWS THROUGH T.I.M.



INFORMATION IS ARRANGED BY COMPUTER AND PLACED IN DATA FILE

DATA FILE

USER CAN MANIPULATE THE FILE AND GENERATE SUB-FILES

USER CAN VIEW THE FILE

REPORTS

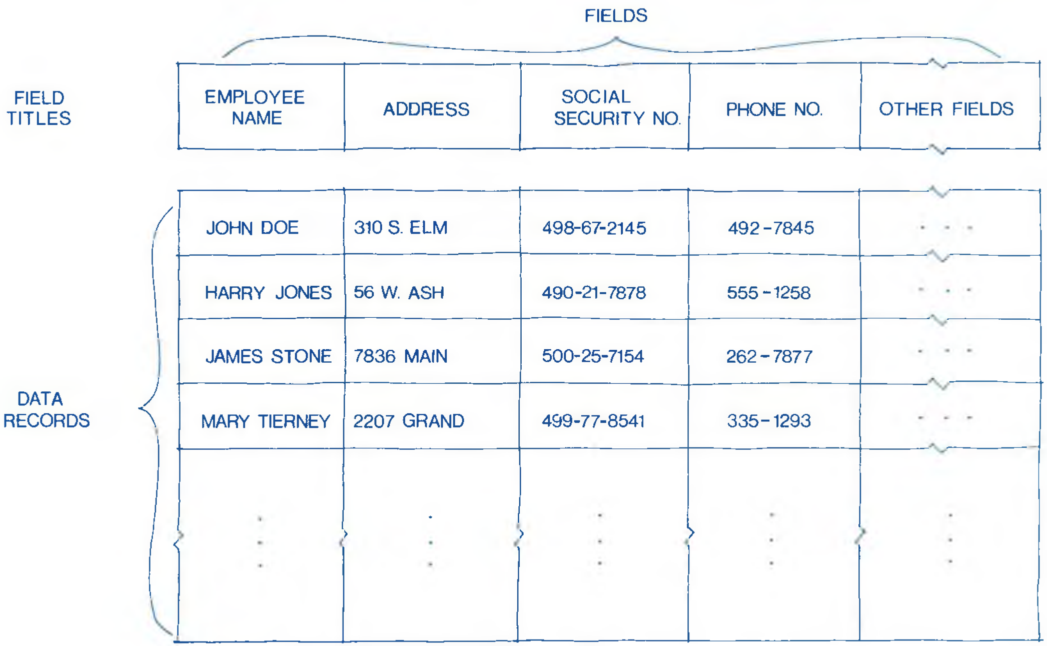
FORM LETTERS

MAILING LABELS

INVOICES

Innovative Software

TYPICAL EMPLOYEE FILE



FIELDS

FIELD TITLES

DATA RECORDS

EMPLOYEE NAME	ADDRESS	SOCIAL SECURITY NO.	PHONE NO.	OTHER FIELDS
JOHN DOE	310 S. ELM	498-67-2145	492-7845	· · ·
HARRY JONES	56 W. ASH	490-21-7878	555-1258	· · ·
JAMES STONE	7836 MAIN	500-25-7154	262-7877	· · ·
MARY TIERNEY	2207 GRAND	499-77-8541	335-1293	· · ·
⋮	⋮	⋮	⋮	⋮

EXAMPLE E-1

0.2.5 T.I.M.'S database management system—Concluded

The true power of T.I.M. can now be seen. Assume this file contained the records of 200 employees; T.I.M. could be used to keep track of employee paychecks or address the envelopes used to mail the paychecks. Form letters could be sent to all customers on file. All of these functions need only to be defined once, and thereafter they would be a quick and painless process. With the same ease, an inventory file could be accessed to give monthly inventory reports in any number of user-definable report formats. A file containing data records of all product purchases could be accessed to print customer mailing labels and invoices. This same inventory file could be used to provide a summary of all sales by product or to give a breakdown of each salesman's monthly totals. The possibilities are endless. T.I.M. can handle any number of data files, each containing up to 32,767 records of information, spanning up to 40 separate fields. The following 12 chapters will explain, in detail, the functions and capabilities of the 12 major T.I.M. commands.

0.3 The concept of a back-up

No computer is perfect. The user of any computer system, therefore, must consider a system crash inevitable if he is to be successful. Even if the hardware and software are assumed to be operating perfectly (a dangerous assumption), it is all too easy for a data base requiring thousands of hours to create, to be obliterated by an accidental system reset, a kicked power plug, or line transient. Such accidents do happen and the intelligent operator will be prepared.

A perfect system would be to back-up the data base after every entry. This is obviously not feasible. Therefore, as a rule of thumb, a back-up should be made often enough so as to be able to re-create the data base in a reasonable amount of time. The phrase "reasonable amount of time" is dependent on how dynamic the data base is. For example, an inventory control system must have back-ups made much more frequently than a mailing list system.

A successful back-up system consists of a working diskette and two back-ups. One back-up is kept at the computer site while the other is removed from the site. This insures that the data base will not be destroyed in case of fire, flood, or other natural disaster. When it is time to back-up the working diskette, a write tab should be placed on it to insure that it will not be re-written. It then becomes the back-up diskette and its contents are copied onto a new working diskette and a second back-up.

Copying a diskette for either back-up or restoration is accomplished using the disk copying utility programs supplied with the operating system.

IMPORTANT! It is important to remember to always write protect the diskette being copied.

0.4 Operating hints

Innovative Software, Inc. has taken every possible precaution to protect T.I.M. data files from system oriented destruction. Once the diskette is removed from the drive, it is the user's responsibility to protect it. The following precautions should be followed to insure maximum diskette life:

1) Diskettes are sensitive to temperature. They should be stored at temperatures between 50 and 125 degrees F. (10 to 52 degrees C). Similarly, the diskettes should be kept in a low humidity environment.

2) The diskettes are flexible recording media but care should be exercised when handling them. They should not be bent or twisted as this may cause loosening of the magnetic storage media.

3) Since the diskettes use magnetic recording techniques for data storage, they are very susceptible to magnetic fields. Care should be taken to avoid any such fields caused by magnets, transformers, motors, or magnetic objects.

4) The magnetic coating on the diskette is very thin and is easily damaged by contaminates. Do not touch or clean the exposed recording surface for any reason. Likewise, do not smoke, eat, or drink while handling the diskette. When ever the diskette is removed from the drive it should be immediately replaced in its protective jacket.

These precautionary measures should be followed religiously in order to preserve data integrity. Should a diskette become severly contaminated, it is not recommended that it be placed back into the system as this could contaminate the read/write head on the drive unit.

A back-up policy should be used to circumvent the loss of data under these circumstances.

0.5 Starting the system

This section is concerned with starting the T.I.M. program for the first time as well as everyday. Before continuing with this section, make sure you have read all sections up to this point.

0.5.1 Daily start up

If you are starting the system for the first time you should skip to section 0.5.2.

Make sure that you have the program disk in drive A. Turn on your computer. The computer will start and the light on the left disk (drive A) will go on. T.I.M. will load and the T.I.M. banner will be displayed on your terminal. T.I.M. will then display the prompt:

```
Enter the default drive (a—p,v = variable)
```

The default drive is where your data file will reside. If your **data will always be on one drive, then that drive** should be entered. T.I.M. will not prompt the user for the drive thereafter. If your data will be on **one of several drives**, then enter a **V** and the program will then prompt you whenever a drive is needed. In most cases the user will enter a **B** meaning the data will be on drive B.

The program will now display the user's name and bring up the main T.I.M. menu.

0.5.2 Starting the system for the first time

For the Victor 9000, the T.I.M. III program is shipped on 1 disk.

Before running your system, you must make a copy of your program diskette. Once this copy is made you should store your original program diskette in a safe place.

0.5.2.1 Copying your T.I.M. master disks

Step 1:

Put your Victor 9000 system master disk in drive A. Put a blank disk in drive B. Type:

```
FORMAT B:
```

This will format your blank disk and put the operating system on it. The terminal will display the message:

```
Diskette format utility
Format drive B complete
```

Then type:

```
SYSCOPY A:  B:
```

This will format and put the system on the T.I.M. WORKING DISK. Remove T.I.M. WORKING DISK and affix an appropriate label.
Step 2:

Put your T.I.M. MASTER DISK in drive A and your T.I.M. WORKING DISK (the disk you just formatted) in drive B. Type:

```
COPY A: *.* B:
```

This will copy all the files from your T.I.M. MASTER DISK to its WORKING DISK copy.

0.5.3 Disk configuration

T.I.M. Master Disk

TIMPARAM	DAT	TIMLST2	EXE
TIMADD	EXE	SELECT	LIB
BASRUN	EXE	TIMSORT	EXE
TIM2	EXE	TIMMERGE	EXE
TIMREP1	EXE	TIMMAINT	EXE
TIMREP2	EXE	TIMFXFER	EXE
REPORT	LIB	TIMBACK	EXE
TIMREMRG	EXE	TIMASC	EXE
TIM	EXE	TIMUTLTY	EXE
TIMHELP	TXT	TIMMAIL2	EXE
TIMSLCT1	EXE	TIMCREAT	EXE
LIST	LIB	TIMSCRN	EXE
TIMLST1	EXE	TIMRCNT	EXE
TIMCHG	EXE	TIMCNFG	EXE
TIMSLCT2	EXE		
TIMLIBFX.	EXE		

0.5.4 Configure general system parameters

Once the T.I.M.program has been properly copied you should start the system. This is accomplished by putting the T.I.M. WORKING DISK in drive A and resetting the system. You will now have to enter the date and the T.I.M. banner will appear. To change the general system parameters such as printer width, available drives, number of disks, the user's name and address, or the user's word processor, the user should enter the letter Z when T.I.M. asks for the default drive.

T.I.M. will ask the user a series of questions about his particular system. These are explained below.

```
Enter the printer width (cr=131)
```

Enter your printer width here. This width is used throughout the program to control different print formats. The maximum you can enter is 255 which indicates infinite width.

NOTE: To users of parallel printers with condensed print capabilities.

Entering the maximum condensed printing width in the configuration stage will allow for greater printing flexibility. When printing lists and reports you will be given the option to print in regular or condensed mode.

```
Enter the available drive letters (eg. ABEF, cr=AB)
```

Here, you should enter which drives are available on your system. T.I.M. will not let you access any drives that are not in this list. Most floppy based systems will enter only **AB** since these are the only drives available.

```
Enter the number of program disks (1-3,cr=1)
```

You should enter how many program disks you will be operating with. The response to this query will be ↵ for one disk.

```
Enter your company's name (40 characters)
Enter your company's address (40 characters)
Enter City, State, Zip (40 characters)
```

Here, you will enter your company's name and address. This will be displayed everytime the system is brought up.

```
Enter the type of word processor you have (cr=none)
```

T.I.M. will now display the menu concerning printer options:

```
1. Parallel Printer
2. Serial Printer
3. Terminal Only
```

EXPLANATION OF PRINTER CHOICES

Choice 1 should be used for any parallel printer. (eg. Epson, Okidata, Victor 6010, Victor 6020, etc.)

Choice 2 should be used for any serial printer. (eg. Diablo, Victor 6040, etc.)

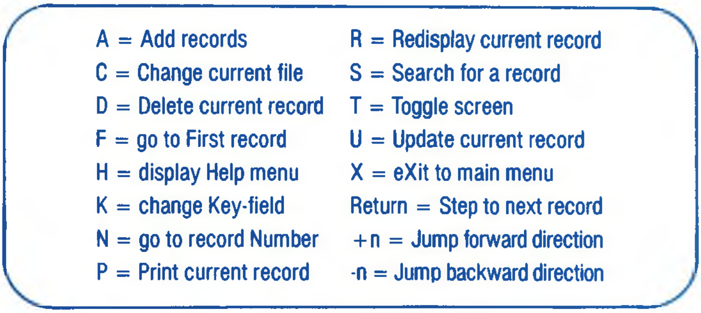
NOTE: You must copy the SET1O.COM command from your MSDOS disk onto your T.I.M. Master disk. This will enable T.I.M. to set your printer parameters.

0.5.4.1 Changing the general system parameters

The parameters defined in the sections above may be reset at anytime by merely entering a **Z** when the system asks for the default drive off the main T.I.M. banner when starting the system.

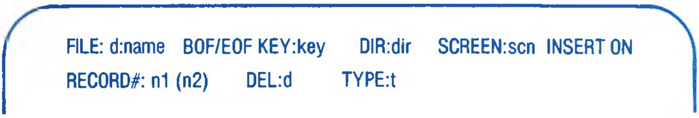
1.0 Add/Inspect/Update A Record

The "*Add*" command allows the user to add new records to his database or update records already present. In addition, there are several methods with which the user may search for and display records. This command may be accessed by entering an "**A**" when the main T.I.M. menu is displayed. After the user enters the filename to work with, T.I.M. will display the 16 "*Add*" sub-commands as follows:



```
A = Add records                R = Redisplay current record
C = Change current file        S = Search for a record
D = Delete current record      T = Toggle screen
F = go to First record         U = Update current record
H = display Help menu          X = eXit to main menu
K = change Key-field           Return = Step to next record
N = go to record Number        +n = Jump forward direction
P = Print current record       -n = Jump backward direction
```

As each record is displayed on the screen a status line will be printed on the top line. This status line serves three distinct purposes. First, it gives information about how the file is being accessed. Second, it gives information about the record being displayed. Lastly, when in the Add or Update sub-commands, it gives specific information about the field where the cursor is located. This status line is organized as follows:



```
FILE: d:name   BOF/EOF KEY:key    DIR:dir    SCREEN:scn  INSERT ON
RECORD#: n1 (n2)      DEL:d       TYPE:t
```

The "FILE" entry gives the name of the file you are currently using, with 'd' as the drive where the file resides.

The "**BOF/EOF**" entry will appear only if the record being displayed is the first record in the file (BOF) or the last record in the file (EOF). Otherwise this entry will be blank.

The "**KEY**" entry shows by what key-field the records are being displayed. (For example, the user may look at the file in order of city.) If the entry after "KEY" is a number, then the records are being displayed in order of that field. If the entry after "KEY" are the letters "SEQ" then the records are being displayed in sequential order (the order in which they were entered).

The "**DIR**" entry shows which direction you are moving through the file. A plus sign (+) indicates you are moving through the file in a forward direction while a minus sign (—) indicates a backward direction. The step function (explained later) will step you in the direction that the status entry indicates.

The "**SCREEN**" entry shows what screen you are looking at. This entry will either be a 1 or a 2.

The "**INSERT ON**" entry is used during the *Add* and *Update* sub-commands. When the "INSERT ON" entry appears, then any characters typed into a field will be inserted and all others will be pushed to the right. If the entry is blank, then any characters typed will write over the current characters.

The "**RECORD #**" shows which record you are looking at. The first number is the actual record number. The second number in parentheses is the relative record number. This number will always be the same as the first number unless you are looking at the file in order of some key-field. For example, the record displayed may be the first record entered into the file, yet it may be the 40th record in alphabetical order. Therefore the record number would be a 1 and the number in parentheses would be a 40.

The "**DEL**" entry will either be an "N" or a "D" corresponding to a non-deleted record or a deleted record. Deleted records remain in the file and are ignored by all commands except the "*Add*" sub-command and the compress command. If the deleted records are updated, they become active again. If the file is compressed, then the deleted records are removed permanently.

The "TYPE" entry is used during the "*Add*" and "*Update*" commands. It shows the field type of the field that the cursor is currently on. It will not appear if you are not in the "*Add*" or "*Update*" modes.

As each record is displayed the status line will be updated and the fields on screen 1 will be displayed. If your file uses a second screen it may be displayed using the Toggle command (T). Additionally, the help menu may be displayed by entering an **H**. The record you were looking at may then be redisplayed by entering an **R**. The various commands on the help menu are explained next, and will be able to be accessed when T.I.M. asks the user to enter the appropriate sub-command.

1.1 Add records

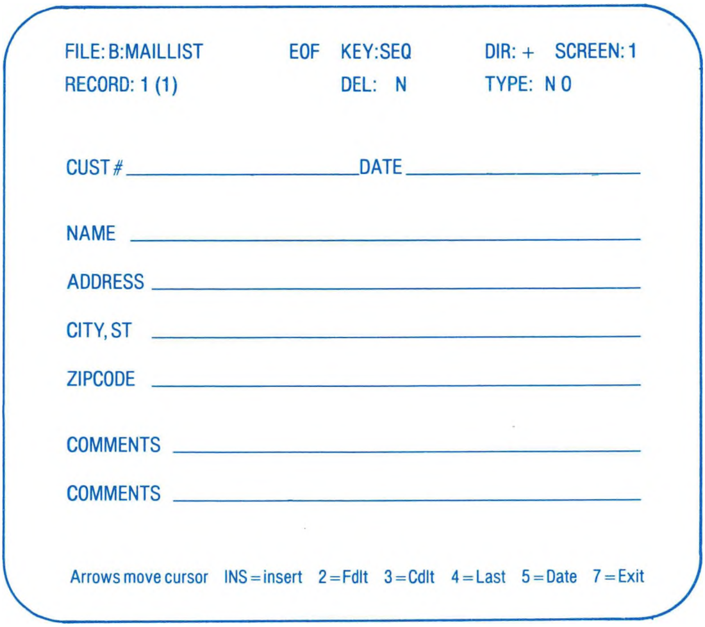
The *Add Records* sub-command is used to add new data records to a file. This mode is entered by typing an **A** in response to the *Add/Inspect/Update* sub-command query. The *Add* and *Update* sub-commands feature true full-screen editing. This means that you may move the cursor to any part of any field and edit only that part of the record. This is the most precise form of data entry and editing since what is on the screen is exactly what the data will look like when accessed or printed later.

Once the number of records has been set you may look at your file using the *Add/Inspect/Update* command. If the first record is correct but the second record appears to be jumbled, then you have not defined the field lengths to be exactly the same as the original file.

1.1.1 Entering data

Once the "*Add*" sub-command is entered, the first screen will be displayed with the cursor located at the first position of the first field. The status line will also be updated to reflect the field type of the field that the cursor is on.

The line at the bottom of the screen will explain the function keys. These 10 function keys are found at the top of the main keyboard keys. They are used to move the cursor and edit data. These edit commands will be explained shortly. For now, take a look at a typical screen format of a blank record, ready for information to be put into it.



```
FILE: B:MAILLIST          EOF   KEY:SEQ        DIR: +    SCREEN: 1
RECORD: 1 (1)                   DEL:   N       TYPE:  N 0


CUST #__________________________DATE___________________________

NAME   ________________________________________________________

ADDRESS _______________________________________________________

CITY, ST _______________________________________________________

ZIPCODE ________________________________________________________


COMMENTS _______________________________________________________

COMMENTS _______________________________________________________


Arrows move cursor   INS=insert   2=Fdlt   3=Cdlt   4=Last   5=Date   7=Exit
```

Data may now be entered into each field. Characters are entered into a field in one of two modes. If the "INSERT ON" flag (right hand corner of the status line) is blank, then characters will overwrite the position where the cursor resides. If the "INSERT ON" flag is displayed, then characters will be inserted at the present cursor location, and all characters to the right of the cursor will be moved over to the right. If the field becomes too long, then the characters to the right will be pushed off the edge.

Data entry in a particular field is terminated by pressing ↵ or by leaving that field using one of the direction keys. As the cursor leaves each field, the data in the field is checked for validity. If invalid data is detected, an error message is given at the bottom of the screen and the field can be edited.

Calculated, Sequential and Total fields will be calculated upon exiting of the field. If characters are inadvertently entered into one of these fields, they will merely be written over by the calculation.

If the record being added to consists of two screens, the second screen will be automatically displayed upon exiting the last field on screen 1. Similarly, the first screen may be re-edited by using the direction keys to return to a field on the first screen.

As each record is added, the next record will be brought up automatically. If you make a mistake in a record after having added it, you must exit the "*Add*" mode and enter the "*Update*" mode as explained later in section 1.10. When you are finished adding records press F10 to exit (as explained in section 1.1.2).

1.1.2 Using the edit commands

There are 11 edit commands to assist the user in entering and editing data. The edit commands are displayed at the bottom of the screen as shown in the previous screen pictorial. The editing & movement commands are explained as follows:

INS = **Insert on/off**—This command turns the insert mode on and off. The insert mode is on when the "INSERT ON" flag appears in the right hand corner of the status line.

F3 = **Delete character**—This command deletes the character under the cursor. All other characters to the right of the cursor are moved one space to the left to fill in for the deleted character.

F4 = **Last field**—This command copies the information from the last record written to disk into the current field. For example, if you are adding many consecutive records with the same city, state, and zip, this command will copy the city, state, and zip from the last record eliminating redundant entries.

F5 = **Today's date**—This command will insert today's date into the current field. This date was entered by the user when the system was first brought up.

F2 = **Delete field**—This command deletes the entire contents of the field where the cursor is positioned.

F7 = **EXIT**—This command exits the "Add" or "Update" mode. In the "Add" mode the current record will not be saved, but all previously added records will. In the "Update" mode the current record being edited will be saved.

TAB = **Tab cursor**—This command will move the cursor 10 spaces to the right in a field.

Cursor movement controls:

← = **Cursor left**—This command moves the cursor to the previous character to the left inside the current field.

→ = **Cursor right**—This command moves the cursor to the next character to the right inside of the current field.

↑ = **Cursor up**—This command moves the cursor up to the previous field in the record.

↓ = **Cursor down**—This command moves the cursor to the next field in the record.

1.2 Change current file

This sub-command allows the user to exit the file being inspected and begin inspecting a new file. **NOTE:** If the user changes files while in the *Add/inspect/update* command, the original file will be exited without merging the newly added or updated records (if any). These records may be merged by returning to the *Add/Inspect/Update* command from the main menu and then choosing to merge the records when prompted.

1.3 Delete current record

This sub-command deletes the current record on the screen. When the record is deleted, the "DEL" entry in the status line will be a "D". Once deleted, the record will stay in the file until the *Compress* sub-command is used. All other commands will ignore deleted records except for "*Add*", "*Compress*" and "*Select*". The record will become active again if it is updated (assuming the file had not been compressed).

1.4 go to First record

This sub-command displays the first record in the file. If you are in the sequential mode, it is the first physical record with a record number of 1. If you are inspecting by a key-field, this will be the first record in that order with an undetermined record number.

1.5 display Help menu

This sub-command will display the entire list of sub-commands to assist the user. To return to the record you were inspecting, press R *Redisplay current record* sub-command.

There are several commands that cannot be accessed when the help menu is displayed. These are: *change Key-field*, *Delete current record*, *Search for a record*, *Toggle screen*, and *Update current record*. These sub-commands either ask for a field number or operate on a specific record so they may be accessed only when a record is displayed on the screen.

1.6 change Key-field

This sub-command is used to change the way that records are retrieved from the file. When the *Add/inspect/update* command is first invoked, records will be displayed in sequential order. This is the order that they were entered. The user may want to look at the records in some other order (eg. in order of zip code). To do this, a key-field may be specified using this command. T.I.M. will prompt the user with:

```
Enter the key-field to inspect by (cr=sequential)
```

You must enter the field number of the key-field to inspect by. Once entered, the "KEY" entry in the status line will display the key-field number you entered. Pressing ↵ will restore the sequential mode of inspecting.

As you are looking through a file in order of a key-field, the numbers to the right of "RECORD #" in the status line are important. The leftmost number corresponds to the sequential order of entry into the file. This is the physical record number. The second number corresponds to the actual ordered position of that record by the key-field specified.

1.7 go to record Number

This sub-command allows the user to access any record by its absolute record number. When this sub-command is used, T.I.M. will query with:

```
Enter the record number to display (x=exit)
```

If the record number entered is greater than the number of records in the file, then the last record will be displayed. Pressing **X** will exit the sub-command. Once this command is executed the "KEY" status entry will return to "SEQ" meaning sequential order.

1.8 Print current record

The user frequently has need to print the record that is displayed on the terminal during the Inspect mode of this command. This sub-command will print a record in a list format.

1.9 Redisplay current record

This sub-command is used to display the record you were looking at before the *Help* sub-command was used.

1.10 Search for a record

This sub-command may be used only when a record is currently being displayed on the terminal. It will look-up a record in the file, by either a part of a field or the whole field. Once this sub-command is invoked T.I.M. will prompt with:

```
Enter field number to search (x=exit)
```

You must enter the number of the field that you want to search through. T.I.M. will then prompt with:

```
Enter the information to be found (x=exit)
```

You must enter what you are looking for. For example, if you are looking for Joe Smith you would enter "Joe Smith": or if you were looking for any Smith just enter "Smith".

It is important to enter this information in the same case (upper or lower) as it appears in the file. For example, "SMITH" is not the same as "Smith". T.I.M. will now prompt with:

```
Is this the complete field or part of a field (c or p,cr=c,x=exit)
```

Here, you must tell T.I.M. whether this is only a part of the information you are looking for or all of it. If you were looking for "Joe Smith" you would press **c** and if you were looking for any "Smith" you would press **P**.

Once you have entered this, T.I.M. will make a decision on which type of search to use. One is faster than the other. If you are looking through a key-field using a complete field search, the fast search will be used. Otherwise T.I.M. will display the prompt:

```
Enter the starting record number (cr=1,x=exit)
```

This prompt serves two purposes. First, if you know that the record you are looking for is toward the end of the file, you can start your search in the middle which will cut some time off of the search. Second, if you are looking for "Smith" and you find "Bill Smith" at record 1000 but are looking for "Joe Smith", you may search a second time starting at record 1001. This will not stop at "Bill Smith" again.

Once the search is started, T.I.M. will display the record numbers as it looks for the specific record. If the record is not found, an error message will be displayed. This could be caused by not entering the information in the proper case (upper and lower).

While the search is in progress you may stop it by entering an **X**. This is useful if you have entered the wrong information and do not want to wait for the search to finish.

If a sequential search is being used and a record has been found, T.I.M. will prompt with:

```
Continue search or exit (c or x, cr=c)
```

If this is not the correct record, you may look for the next one by entering a **c** or ↵.

1.11 Toggle screen

This sub-command is used when a record has two screens of information. When the record is displayed, the user may toggle back and forth between the two screens using this sub-command.

1.12 Update current record

This sub-command is used to change records that are already in the file. Before this sub-command can be used, a record must be displayed on the screen. This can be done using the search or step sub-commands.

The data entry and editing are exactly the same as described in section "1.1 Add records" with one exception: the exit sub-command will exit the current record and save it as opposed to abandoning it as in "*Add*".

The "*Update*" sub-command will allow the user to update up to 100 records in a file at a time. If more than 100 records are needed to be updated, the user must merge the records first before continuing.

1.13 eXit to main menu

This sub-command is used to exit the *Add/inspect/update* command. All changes made are saved on the disk. If no changes were made the program will return to the main menu. Otherwise, T.I.M. will display the prompt:

```
Do you want to merge records now (y or n,cr=y)
```

T.I.M. is asking whether you want the records put in order now. If you say yes, the records will be put in order according to the defined key-fields. Once they are in order they will show up in reports, lists, etc.

If they are not merged they will still be in the file but will not show up until they are merged. The reason you may not want to merge the new records is that it takes time to put them in order. During data base creation, you may want to add several thousand entries before ever printing a report. You probably would not merge the records for several sessions since there is no reason to do so.

If you have changed files using the *Change current file* sub-command, all files prior to the last one looked at will require merging. You may merge any one of these files by exiting the *Add/Inspect/Update* command, return to it from the main T.I.M. menu, and then exit directly after looking at any one of the non-merged files.

Note: If you have not defined any key-fields, then always say yes to this query. This will store all records permanently in the file.

1.14 Step to next record

This sub-command will display the next record in the direction you are moving by pressing the ↵ key. If you are inspecting by a key-field, the next record displayed will be the next record in that order. If you are inspecting sequentially, the next record will be the next physical record (e.g.1,2,3...). Remember that you will be moving in the direction indicated by the "DIR" status entry (either + or —).

1.15 Jump in forward direction

This sub-command lets you jump forward through the file. It also changes the "DIR" status entry to a plus sign (+). For example, if you are on record 10 and you enter **+10** you will go to record 20 (if you are in the sequential mode). If you are inspecting the file by a key-field, you will jump ahead 10 records in that order rather than 10 physical records as in the previous example. Simply entering a plus sign with no number is the same as entering **+1** and may be used to change direction.

This sub-command may be used to find the last record in the file. To do this enter **+n** where n is a larger number than the number of records in the file, but is smaller than 32,767.

1.16 Jump in backward direction

This sub-command is the same as the *Jump in forward direction* sub-command with the exception that everything happens in the reverse direction. When this command is invoked, a minus sign (—) will be put in the "DIR" status entry denoting a backward movement through the file.

1.17 Points to remember

1.) This command combines three functions, adding new records to a file, inspecting current records and updating (changing the data in), current records.

2.) This command can be used to search for a record which contains specific information. Entire fields may be searched for exact matches or just for part of a field matching the information searched for.

3.) A deleted record may be made non-deleted by simply updating the information in it.

4.) The function keys control the edit functions.

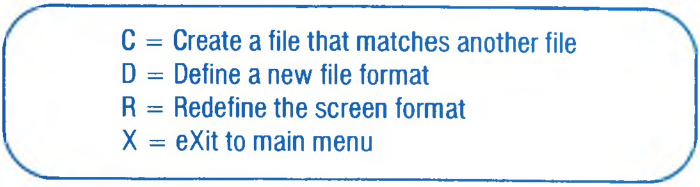
5.) Help is always available at the bottom of your screen.

6.) Certain sub-commands cannot be accessed without having a record displayed on the screen.

7.) You may not update more than 100 records at a time without merging records.

2.0 Create a file

The *Create a file* command can be used to create a new file or redefine a screen format. This command has 4 sub-commands as displayed below:



```
C = Create a file that matches another file
D = Define a new file format
R = Redefine the screen format
X = eXit to main menu
```

A file consists of up to 40 fields of up to 60 characters each. Within each record of up to 20 calculated fields and up to 3 total fields may be defined. **Every field may be a key-field if desired although each additional key will slow execution during the merging of newly added records.**

2.1 Create a file that matches another file

This sub-command allows the user to create a new T.I.M. file with file specifications exactly the same as an existing file. This saves the user much time if two files are needed, but both have the same field requirements. After accessing this sub-command, T.I.M. will prompt:

```
Enter the origin filename (x=exit)
```

Here the user should enter the filename of the file to copy. After the origin filename is entered, T.I.M. will prompt:

```
Enter the destination filename (x=exit)
```

The user will then enter the name of the new file, or destination filename. Lastly, the prompt:

```
Enter comment (cr=none)
```

will be displayed. This comment is used only as a reference for yourself. It is not used by the program. Once entered, T.I.M. will create the new file and write it to disk. This new file will have the exact file specifications as the origin file, and might differ only in the comment area. This new file, however, will have no records in it.

2.2 Define a new format

By entering a **D** when the create sub-command menu is displayed, the user is able to define a new file format. First, field information is entered, followed by overall file information.

2.2.1 Entering field information

As soon as the *Define a new format* sub-command is invoked, the prompt:

```
1 <SL=2400> Title:
```

is displayed on the screen. Here the user enters the field title of the field he is defining. The title is used when adding or updating records and in reports. This title must be 16 characters or less in length and consist of any combination of alphanumeric characters or symbols except the asterisk. The asterisk is used for error correction purposes which will be covered shortly. If the user should enter a question mark (?) in response to the field title query, T.I.M. will display all previously defined fields on the user's terminal.

After the field title is entered, the cursor will move to the right and display the prompt:

```
Length:
```

Here the user enters the field length of the field he is defining. This length may be from 1 to 60 characters.

Lastly the field type must be defined. The prompt:

```
Type:
```

will be displayed after the field length is entered. Here the user defines the field type. T.I.M. will accept any of the characters A,C,D,I,N,S,T or $ as legitimate field type entries. These 8 field types are listed next:

```
A = Alphanumeric field
C = Calculated field
D = Date field
I = Inverted name field
N = Number field
S = Sequential field
T = Total field
$ = Dollar amount field
```

For additional information regarding these field types, consult section *0.2.2.2 FIELD TYPES.*

If any numeric type field is defined, the prompt:

```
Precision:
```

will be displayed. Here the user must specify how many places of decimal precision are needed. The user may define decimal precision from 0 to 4 decimal places, with a carriage return defaulting to 0 decimal places. A dollar field will be automatically given 2 decimal places.

If a date field is defined, the prompt:

```
Date type:
```

will be displayed on the terminal. There are two date types, American and European. American dates are in the form, month/day/year and European dates are in the form day/month/year. These two date types are included for correct sorting of both types of dates. Press **A** for American or an **E** for European.

2.2.1.1 Calculated and total fields

A calculated field can be defined if the user needs arithmetic operations on fields within a record. For example, a field containing quantity can be multiplied by a field containing price and put the result in an extension field. A variation of the calculated field is the total field. A total field allows up to 16 fields to be totaled into one field. Both the calculated field and the total field are automatically calculated by the program.

If a calculated field was specified, the user must now define the calculation to be performed. T.I.M. will list the previously defined fields in the normal field list fashion and then query with:

```
Enter the field number of the first operand (x=exit)
```

The first operand must be one of the previously defined fields and must also be some type of numeric field (numeric, dollar, sequential, total or calculated). Once a valid field number has been entered for the operand, T.I.M. wil! query with:

```
Enter the operator (+,—,*,/,x=exit)
```

Once the operator has been entered T.I.M. will query with:

```
Enter the field number of the second operand (c=constant)
```

If a field number is entered, it must follow the same requirements as the first operand. If the user desires to use a constant as the second operand, he should press C. T.I.M. will then query with:

```
Enter the  constant (x=exit)
```

The user may now enter a constant to complete the calculated field definition. This constant may be up to 5 digits long. The length of a calculated field should be chosen carefully since the result of a calculation can be longer than the length of the two operands. There may be up to 20 calculated fields defined per file.

If a total field is defined, the user starts by defining the decimal precision of the field just like in the number or calculated field. Next, T.I.M. will prompt:

```
Enter fields to total separated by commas (e.g. 1,2,5,x=exit)
```

The user should enter all the fields he wishes to total and separate them by commas. A maximum of 16 fields may be entered to total. You may have up to 3 total fields.

2.2.1.2 Error correction

If the user enters the field title, field length, or field type incorrectly, he may enter an asterisk (*) and T.I.M. will go back to the preceding field number to allow the user to enter the appropriate field information. If the user's mistake was more than one field preceding, the user simply enters the single asterisk repeatedly until the incorrect field is reached.

When the user is through defining field information, he should press ↵. The prompt:

```
Are you finished defining (y or n)
```

will be displayed. If the user is finished, he should enter a Y, and T.I.M. will continue with the file definition queries.

2.2.2 Defining file information

Now that the new file format has been defined, T.I.M. will ask:

```
Enter the new filename you wish to create  (x=exit)
```

If the user does not wish to continue, he should enter an **X** and T.I.M. will return to the original "*Create*" sub-command menu. If the user wishes to create a new file, he should enter a name 1-8 letters or numbers in length. Filenames may not contain spaces or symbols. After the filename is entered, the prompt:

```
Is this a confidential file (y or n, cr=n)
```

will be displayed. If a **y** is entered, the user will be asked to enter a 4 letter password. A file created with this option cannot be accessed through T.I.M. in any way without knowing the password. This password may be eliminated or changed using the password update sub-command located in the *file Maintenance* command.

T.I.M. will then display the prompt:

```
Enter comment (cr=none)
```

Here the user may enter a comment to help identify the new file. The comment may be up to 60 characters long. This comment is not used by the program but is merely a reference for the user. And may be seen by using the *File specifications* command.

2.2.3 Key-field definition

T.I.M. will ask the user if he wishes to define key-field(s) for his file. (For additional information regarding sorting, key-fields and record merging, consult the chapter 8 subcommand, *Sort a file.*

The user is not required to define any key-fields upon file creation since it may be done at any time using the *Sort a file* sub-command.

If the user does not wish to define any key-fields, T.I.M. will move on to the final file information queries; if the user does, T.I.M. will query as shown below:

```
Enter major key-field number (cr=end)
```

If the user defines a key-field, T.I.M. will then query with:

```
Enter minor key-field number n (cr=end)
```

The user may continue entering minor key-fields until 16 are defined or until he wishes to stop. If the user wishes to stop defining minor key-fields, [↵] should be pressed.

This key-field definition process will repeat if the user desires, so as to give the capability of defining up to 40 different key-fields per file.

Finally T.I.M. will display the prompt:

```
Define custom screen or use standard screen (c or s, cr=s)
```

A custom screen is a screen layout defined by the user. Using a custom screen, the fields may be placed anywhere on the screen. A standard screen merely lists the fields down the left side of the screen, one under the other. If custom screens are not desired, T.I.M. will return to the "*Create*" sub-command menu.

2.2.4 Custom defined screens

If the user wishes to define a custom screen layout for his file, this option should be used. The custom screen option allows the user to define the placement of each of the fields of a file on one or two 80x20 screens. The user can create very attractive data entry forms using custom screens. For instance, fields may be logically grouped and more than one field may be put on a line (assuming their field lengths permit). Data entry could then resemble the form of the user's data before being computerized. T.I.M. will display three 'help' lines on the terminal during the screen definition process. The first line is the "status line" and contains field and screen information. It is shown below:

```
FIELD:n title   Length:l   Line:ln   Column:c   Screen:s
```

This line is explained as follows. "n" is the field number of the field currently being positioned, "title" is the field title of the same field, "l" is the field length of the field currently being positioned, "ln" is the line (1-20) that the cursor is currently on, "c" is the column number (1-80) of the cursor's present position, and "s" is the screen (1 or 2) that is currently being displayed.

To define a field position, the user should use the direction keys to move the cursor to the position wanted, and then type:

INS

to insert the field at that position. T.I.M. uses a variety of commands to facilitate the screen definition process. These commands are accessed by use of the function keys and other keys and are explained as follows:

```
B = Blank all fields
E = go to End of screen
F = go to First field
P = insert Previously defined screen
DEL = Delete current line
S = go to Start of Screen
T = Toggle to next screen
SPACE = Insert blank line
TAB = Tab 10 spaces
```

Blank all fields: all fields in one or both screens will be blanked, that is, purged from the screen definition.

Delete field: this will remove the field at the cursor position from the defined screen format. (Note: the cursor must be on the first character of the field to delete)

Go to first field: will set the current field to insert at the first field.

Next Help menu: There are 3 successive Help menus which list all screen definition commands. These menus are displayed on the bottom two lines of the user's terminal. By using this command, the user may view any or all of these Help menus.

Insert field at this position: will place a field at the present cursor position.

Toggle to next screen: will display the alternate screen (i.e. screen 1 toggles to screen 2).

go to next field: will display the next field in the upper left corner of the status line. This will allow a new field to be inserted.

go to previous field: will display the previous field in the upper left corner of the status line.

Delete current line: will delete all defined field positions on the current cursor line and move all fields below the cursor up one line.

Insert blank line: will insert a blank line between two defined lines and move all lines below the cursor down one line. If there was a line at the bottom, it is pushed off and must be redefined.

TAB 10 spaces: will move the cursor 10 spaces to the right.

eXit screen definition: will exit the custom screen definition process so the new screen can be saved.

← = Cursor left

→ = Cursor right

↑ = Cursor up

↓ = Cursor down

INS = Insert field at this position

F2 = Delete field

F3 = Go to first field

F4 = Go to next field

F5 = Go to previous field

F8 = Next HELP menu

F7 = Exit screen definition

After defining the screen T.I.M. will then display the prompt:

```
Do you wish to save this screen (y or n, cr=y)
```

2.3 Redefine screen format

This sub-command allows the user to redefine the screen format of any defined file. The full explanation of this command may be found by consulting the previous section, *2.2.4 Custom defined screens*.

2.4 eXit to main menu

The user may exit the "*Create*" command by entering an **X** when asked to enter the "*Create*" sub-command.

2.5 Points to remember

1.) Before adding information to a file, you must use this command to define and create the file.

2.) To Create a file, you will define the field titles, field lengths, field types and key-fields to be used.

3.) Fields may be made key-fields using this command. A key-field is a field by which the entire file may be ordered. You may have up to 36 key-fields per file.

4.) Remember that each key-field defined takes time to merge records after record addition. Keep the key-fields to a minimum to avoid longer merge times.

5.) Screen formats may be defined to look like your own paper forms. Fields may be placed anywhere on up to two screens.

6.) If you do not wish to define a custom screen, you may choose the "standard screen" option.

3.0 Display T.I.M. directory

If the user wishes to know which T.I.M. files are contained on a particular disk, he should use this command.

If the user wishes to access this command, he should enter a **D** when the main T.I.M. menu is displayed on the user's terminal. T.I.M. will then ask:

```
Enter drive (a—p,cr=b)
```

Here the user enters the drive letter of the disk he wishes to see. The program will then display all the T.I.M. data files on the specified drive and the number of records in each one.

3.1 Points to remember

1.) Only T.I.M. data files will be displayed using this command.

4.0 File specifications

If the user desires to list the specifications of a particular file, he should enter an **F** when the main T.I.M. menu is displayed on the terminal.

T.I.M. will ask the user to enter the filename of the file he wishes to list the specifications of. If the default drive was specified as variable, the appropriate drive will be asked for next.

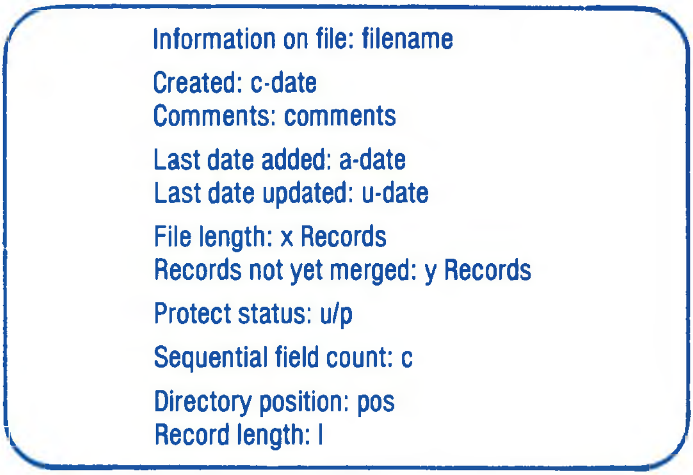
Also at this time, the password must be entered if the file being listed is a protected file.

4.1 File specifications format

After the filename and password (if needed) have been entered, T.I.M. will display three screens of File specifications.

4.1.1 Screen 1

The first screen contains general file information and is shown below:



```
Information on file: filename

Created: c-date
Comments: comments

Last date added: a-date
Last date updated: u-date

File length: x Records
Records not yet merged: y Records

Protect status: u/p

Sequential field count: c

Directory position: pos
Record length: l
```

where **"filename"** is the name of the file being specified, **"c-date"** is the date the file was created and **"comments"** are any comments regarding the file.

"a-date" and **"u-date"** are the last dates the file was added to or updated, respectively.

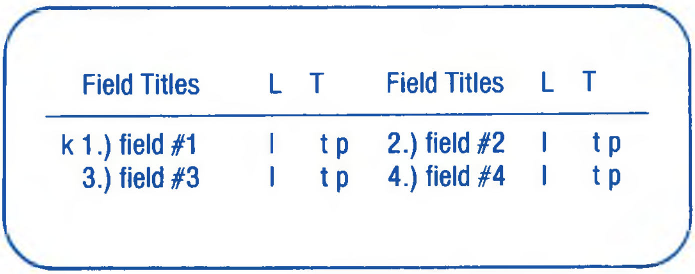
"x" is the file length of all merged records, and **"y"** is the number of records not yet merged into the sorted orders of the file. (For additional information, consult the merge section of the *Add/inspect/update* command).

The variable **"u/p"** designates whether the file is either unprotected or protected. **"c"** is the sequential field count, which is the number that was last entered into a sequential field. Even if a sequential field was not used this number will be incremented with every added record.

The last two variables are for use by the systems people. **"pos"** is the record position within the T.I.M. directory (TIMDIR.DAT) that the header records of the file begin. **"l"** is the total record length in bytes. This record length is the totaled lengths of all the defined fields plus 1 extra byte added for a record delete flag.

4.1.2 Screen 2

The second screen displays all the field information and will look like the one below:

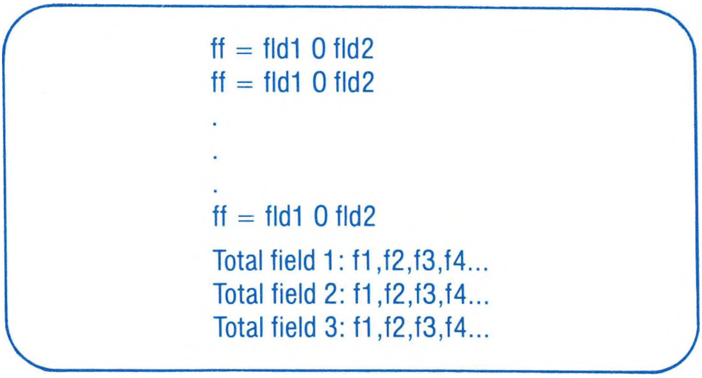


```
   Field Titles     L   T      Field Titles    L   T
 ------------------------------------------------------
 k 1.) field #1     l    t p   2.) field #2    l    t p
   3.) field #3     l    t p   4.) field #4    l    t p
```

Here, "**field #1, field #2,...**" are the field titles of all defined fields. (There may be up to 40). "**l**" and "**t**" are the field lengths and types respectively, and "**p**" is the decimal precision of any numeric, calculated, dollar or total field. Also, if a date field is defined, the "**p**" variable will be either an A or E depending on if the date type defined was American or European. Lastly, the lower case k will be displayed to the left of any field that is a key-field.

4.1.3 Screen 3

Screen #3 contains all the calculated and total field information, and will be displayed in the following manner:



```
ff = fld1 0 fld2
ff = fld1 0 fld2
.
.
.
ff = fld1 0 fld2
Total field 1: f1,f2,f3,f4...
Total field 2: f1,f2,f3,f4...
Total field 3: f1,f2,f3,f4...
```

If the file being specified contains any calculated fields, they will be displayed as above. "**fld1**" is the first field operand, "**0**" is the calculation operator (+, −,*,/) and "**fld2**" is the second field operand or a constant. If more than one calculated field was defined, they will appear one under the other as shown above. If the file being specified contains any total fields, their operands would be listed as "**f1,f2,f3,...**"

After any one screen is displayed, the next screen may be displayed by typing a ↵, and the command may be exited by entering an X. If the user wishes to print the information, he should enter a P.

4.2 Points to remember

1.) File specifications on any T.I.M. file, residing on any drive may be listed on the screen or printed.

2.) Listed information includes field titles, number of records in the file, calculated field equations, protect status, and merge status.

5.0 Help menu

This command will display help messages about each of the T.I.M. commands. To access the "HELP" command, enter an H followed by the command letter of the command in question. For example, to get to the help menu for the "*Report generation*" command, you would enter **HR**. An **HH** will display general program information.

If an invalid command letter is entered the error:

```
Error—no help available
```

will be displayed. This error can also be caused by not having the help text file on the disk.

6.0 word processing Interface

This command reads a T.I.M. data file and creates a new file which is compatible with the user's word processor. The command determines the correct output format depending on which word processor was selected during system configuration. The output file will have the extension "DTA".

6.1 Using word processor Interface

T.I.M. will prompt the user with:

```
Enter the origin filename (x=exit)
Enter the drive (a—p,cr=b)
```

The user must enter the name of the T.I.M. file to transfer.

T.I.M. will then prompt the user with:

```
Enter the destination filename (x=exit)
Enter the drive (a—p,cr=b)
```

This is the name of the word processor compatible file and drive which the program will output. Do not enter the extension.

T.I.M. will display the fields of the file and then prompt the user with:

```
Enter the key-field number to retrieve records by (cr=seq)
```

The user must enter which order the records will be put in. If a key-field is chosen, the output file will be in order by that field. By pressing ↵ , the records will be in the same order as the T.I.M. file (sequential order).

T.I.M. will again display the field list and prompt the user with:

Where "n" is the number of fields the user has chosen to transfer. The user must enter the field numbers of the fields he wishes to transfer to the new file one at a time. This prompt will appear repeatedly until ⏎ is pressed.

As the transfer is in progress the message:

```
Record # n out of t
```

will be displayed. This message indicates how many records have been transferred to the new file where "n" is how many and "t" is the total to transfer.

Once the transfer is complete, the user's word processor may access the new data file. This will permit the printing of form letters, simple invoices, etc.

6.2 Points to Remember

1.) This command will provide a link between T.I.M. data and word/processors to give an integrated computer system.

2.) The origin filename is the name of the T.I.M. file that will provide the information that the word processor needs.

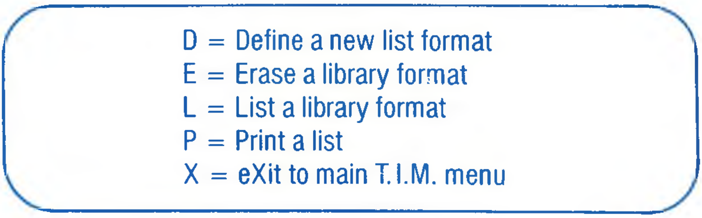
3.) The destination filename is the new file that will be written which will be read by the word processor.

4.) This command does not directly print merged documents on a word processor, it is just the means to communicate, via a disk file, between T.I.M. and a word processor.

7.0 List generation

The *List Generation* command is used to generate lists from a T.I.M. data file. This command can be used to generate mailing labels, fill in forms, print checks or print invoices.

When you access this command, T.I.M. will display the following **List generation** sub-command menu.



```
D = Define a new list format
E = Erase a library format
L = List a library format
P = Print a list
X = eXit to main T.I.M. menu
```

7.1 Defining a new list format

Before a user may print any file information using the list generation routine, he must define the list format to be used when printing. To define a new list format, the user enters a "D".

7.1.1 The concept of a list format

When the user defines the parameters of a list format, he should understand the built-in features that T.I.M. supports to facilitate the defining process. First, the user may define a list format which will include 1, 2, 3, or 4 lists across the printed page, thereby allowing the use of any of the standard mailing label paper.

The overall concept of the list generation routine is that the user needs only to define one list format regardless of how many lists are to be printed across the page. This defined list format may include up to 48 field entries from the host data file, and each may be truncated to any length, if the user desires. In addition to the field entries, the user may include any text he wishes to have printed along with the file information, on every printed list. This would be useful in order to add punctuation as needed or to have internal store information, record numbers, or dates printed on every list.

The user may instruct T.I.M. to print 1 to 4 lists across the page. Consequently, the user needs to define the starting tabs of each list printed across the page. For example, the first list may start at a tab of 1, the second list at a tab of 40, and the third list at a tab of 80. In addition to the tabs, the user must specify the number of lines between lists (the number of blank lines between the last line of one list and the first line of the next list down the page).

As a conclusion, the user should treat the list format as a block of information to be printed. This block of information may be repeated to allow up to four lists of information to be printed across any one page. The overall list format information is asked for first and pertains to the overall listing process, not one particular list. Lastly, the user will enter particular information such as field entries, field widths, and the user defined text of each list format.

7.1.2 Entering general list information

After the *Define a new list format* sub-command is accessed, T.I.M. will display the following prompt:

```
Enter the filename to generate lists from (x=exit)
```

Here, the user should enter the name of the file he wishes to use. Next, T.I.M. will begin the overall information queries. T.I.M. will prompt:

```
Enter number of lists across page (1-4)
```

The number to be entered will tell T.I.M. how many lists will be printed across on one page.

After the number of lists is entered, T.I.M. will ask for the starting tabs of each defined list across the page. If you have three lists, the tabs might be 1, 40 and 80.

After the tabs are entered, T.I.M. will prompt:

```
Enter number of lines between lists
```

(Think of each list as a block of information.) The number entered should be the number of blank lines between the last line of one list and the first line of the list which will be printed directly beneath it.

The last overall query will read:

```
Enter the key-field number to print from (cr=seq)
```

The key-field is the ordered field by which the lists will be printed. For example, you may want to print in alphabetical order by name. Here you would enter the number of the name field for the key-field. If a sequential order of lists is desired, press ↵ instead of a field number.

7.1.3 Entering specific list information

After the user completes entering the overall format information, T.I.M. will begin asking for the particular list information. T.I.M. will query the user in the following manner:

```
Line#:    Column#:    Field#:    Truncation:    Text:
```

T.I.M. will ask for each of these five entries one at a time, starting with "Line #:" and ending with "Text:". The user should enter all the information as asked. For example, assume a user wanted the contents of field #3 printed on the first line of every list, starting in column 10. Also assume the user wanted the contents of this field truncated after 20 characters. He would then enter a "1" for the Line #:, "10" for the column #:, "3" for the field #:, "20" for the truncation and ⏎ for the text entry.

The "Text:" entry allows the user to print a text entry on every list generated. This might be handy for store codes or a date entry.

T.I.M. allows for error correction throughout the entire "List" definition process. If the user needs to correct a previous entry, he should enter an asterisk (*) for the present query, and T.I.M. will back-up to allow him to re-enter the correct information.

It should be noted that each of the sub-formats may be entered in any order, not necessarily in a first-line-first-order. T.I.M. will automatically sort all entered sub-formats.

When the user has entered all the list sub-formats desired, pressing ⏎ will end the sub-format definition process. T.I.M. will then ask:

```
Enter the library format name (20 char.,x=exit)
```

Here, the user enters a name to identify the newly defined list format. Once entered, the format and name will be added to the List Library. Control will then return to the *List generation* sub-command menu. This list may now be printed repeatedly without re-defining the format.

7.2 Erase a library format

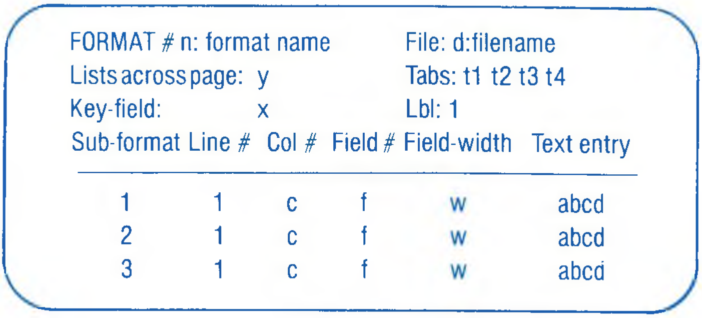
If the user decides that a particular List Library format is no longer needed, it may be erased using this sub-command. After the user enters an E, T.I.M. will ask:

```
Enter library format number to erase (x=exit)
```

The user should then enter the number of the format to be deleted from the List Library. After the number is entered, control will return to the "*List*" sub-command menu.

7.3 List a library format

The *List a library format* sub-command will display the information stored in the List Library which pertains to a specific list format as seen below:



```
FORMAT # n: format name              File: d:filename
Lists across page:  y                Tabs: t1 t2 t3 t4
Key-field:          x                Lbl: 1
Sub-format Line #  Col #  Field #  Field-width  Text entry
----------------------------------------------------------
    1        1       c       f         w           abcd
    2        1       c       f         w           abcd
    3        1       c       f         w           abcd
```

where "n" is the format number used to identify the list format and "format name" is the name of the forementioned format. Additionally, "x" is the key-field number by which the lists will be printed in order. (If a sequential print has been defined, the "x" will be replaced with a "SEQ"). "d:filename" is the drive letter and the filename of the file being printed. "y" is the number of lists to be printed across one page, and "t1, t2, t3, t4," are the starting tabs of each list to be printed. The variable "l" is the number of blank lines between lists.

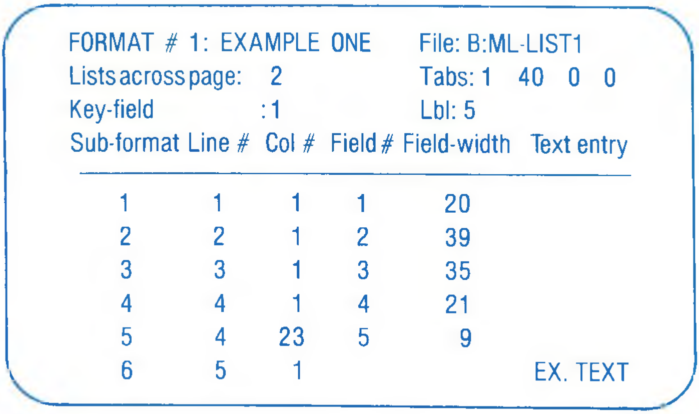
starting tabs of each list to be printed. The variable "l" is the number of blank lines between lists. You may define up to 48 sub-formats in a library format.

The specific list information is listed below the overall list information in the sub-format table. Here "l" is the line number the particular sub-format is to be printed on: "c" is the column number that the defined information will start in: "f" is the field number (if any) of the field to be printed; and "w" is the field width (if a field is defined) to be used as a truncation length; and "abcd" (if used) represents any text the user desired to be printed on every list. For more information regarding any of the information listed in the List Library, consult section *7.1 Defining a new list format.*

After the individual list format is displayed on the user's terminal, it may be sent to the printer by entering a "Y" when queried.

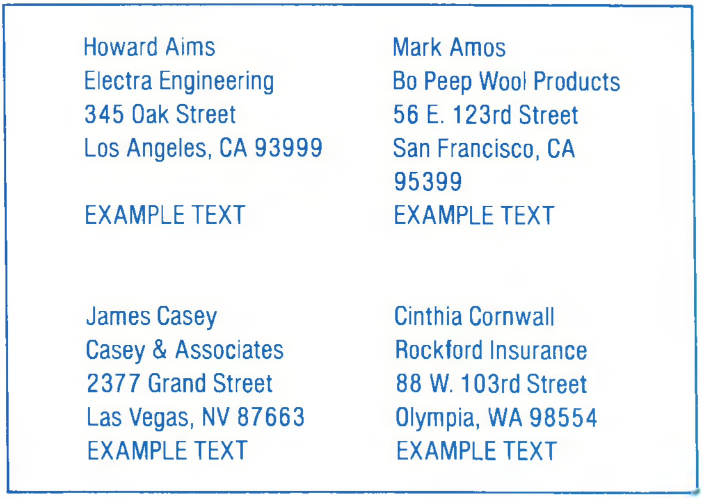
7.3.1 Example List Library format

An example List Library format is shown below. Notice the general list information at the top and the specific list information at the bottom. The List Library format number is "1", and the List Library format name is "EXAMPLE ONE". This particular format is designed for two lists across the page, with the starting tabs for those two lists at column 1 and 40. The number of lines between lists (Lbl) is 4 lines and the order by which the list is printed is by field number 1 (the key-field).



```
FORMAT # 1: EXAMPLE ONE             File: B:ML-LIST1
Lists across page:   2              Tabs: 1   40   0   0
Key-field           :1              Lbl: 5
Sub-format Line #  Col #  Field # Field-width  Text entry
--------------------------------------------------------
   1        1       1       1         20
   2        2       1       2         39
   3        3       1       3         35
   4        4       1       4         21
   5        4      23       5          9
   6        5       1                              EX. TEXT
```

Notice that there are six sub-formats, each printing a particular field, the last one not printing a field, but only some text that is common to all the lists. The text may be printed anywhere on the list. A portion of this list is shown next.



```
Howard Aims                    Mark Amos
Electra Engineering            Bo Peep Wool Products
345 Oak Street                 56 E. 123rd Street
Los Angeles, CA 93999          San Francisco, CA
                               95399
EXAMPLE TEXT                   EXAMPLE TEXT


James Casey                    Cinthia Cornwall
Casey & Associates             Rockford Insurance
2377 Grand Street              88 W. 103rd Street
Las Vegas, NV 87663            Olympia, WA 98554
EXAMPLE TEXT                   EXAMPLE TEXT
```

7.4 Print a list

When the user decides to print a list which has been defined in the List Library, he should enter a **P** for the "List" subcommand. T.I.M. will then display a directory of the formats stored in the List Library and query:

```
Enter the library format number (x=exit)
```

The user should enter the number which corresponds to the List Library format he wishes to print. T.I.M. will then access this library format and prompt:

```
Use defined file named <d:filename> (y or n, cr=y)
```

Here "d:filename" is the drive number and filename of the file to be printed. If a different file is desired, type an **N** and enter the new filename when asked.

After the file is chosen, T.I.M. will access the disk and display the prompt:

```
Send to printer or terminal (p or t,cr=p)
```

The user has two options at this point. He may print the lists on paper or send them to his terminal (usually used for test purposes). Entering a **T** will send the lists to the terminal, a **P** sends them to the printer.

T.I.M. will then query:

```
Pause between lists (y or n, cr=n)
```

This option, if enabled, will cause T.I.M. to stop printing between each vertical group of lists. This will allow single sheets to be inserted for each list.

T.I.M. will then query:

```
Start at what record number (cr=1,x=exit)
```

This option allows printing to begin anywhere in the file. For example, the user may want to print half of his mailing labels one day and half the next day. This can be done by stopping the listing on the first day and restarting them the second day where you left off.

If the terminal was chosen, the list generation will begin, if the printer was chosen, T.I.M. will give a "paper ready" or "Test" prompt. This feature is explained in the next section.

7.4.1 List generation test feature

T.I.M.'s *List generation* command possesses another feature which should be mentioned at this time. Immediately before the lists are printed, T.I.M. will remind the user to check to see if his printer paper is set correctly, by displaying the query:

```
Paper ready (t=test,cr=print)
```

on his terminal. At this time, the user may wish to "test print" a line of characters. This "test print" feature, accessed by entering a T when the "PAPER READY" query is displayed, will cause a line of characters to be printed on the user's printer so he may correctly line up mailing labels to be printed upon. In addition, asterisks (*) will be printed down the left margin corresponding to how many additional lines will be printed on each list. This process may be repeated until the paper is correctly aligned in the printer.

7.4.2 List printing abort feature

After T.I.M. begins the printing of a list format, the message:

```
Press "X" to stop printing
```

will appear on the user's terminal. If the user wishes to abort the list generation process and return to the "List" subcommand menu, he should press and hold down the X key until the printing process is aborted.

T.I.M. will then ask the user if he wishes to continue printing or abort the printing process completely.

7.5 eXit to main menu

If the user wishes to exit the *List generation* routine, he should enter an X. This will return control to the main T.I.M. command menu.

7.6 Points to remember

1.) Lists are generated by definition of fields to print, tabs, etc.

2.) When defining remember that a **list format** is all the fields and text will be printed in one list (i.e. one mailing label).

3.) First you will give T.I.M. the overall list information. General information includes how many lists are printed across the page, what tabs these lists will start at, the lines between each list and so on.

4.) After the overall information, you will define which fields to print on the list and their location.

5.) Text may be printed common on every list if desired.

8.0 file Maintenance

If the user wishes to use T.I.M.'s varioius file maintenance functions, he should access the *file Maintenance* command.

The *file Maintenance* command possesses 11 sub-commands as follows:

```
A = Alter sequential count
C = Compress a file
D = Display disk directory
E = Erase a key-field
K = Kill a T.I.M. file
M = reMove duplicate records
P = Password update
R = Rename a T.I.M. file
S = Sort a file
U = Update field titles
X = eXit to main menu
```

8.1 Alter sequential count

The *Alter sequential count* sub-command will reset the current count of a Sequential field in a file. After it is reset, the next added record will be incremented starting with the new value.

T.I.M. will ask for the filename of the file containing the sequential field and after it is entered, will prompt:

```
Enter the new count (x=exit)
```

At this time the new count can be entered.

8.2 Compress a file

The *Compress a file* sub-command allows the user to remove all deleted records from a T.I.M. data file. All key-fields, however will remain in their sorted order after the file has been compressed.

The user will need to enter the name of the file to be compressed. T.I.M. will then display the message:

```
Compressing file  <filename>
```

where "filename" is the name of the file being compressed. Additionally, the message:

```
Pass 1 of x
```

will be displayed. Here, "x" is the total number of passes, each pass being 1000 records. During each pass, T.I.M. will locate all deleted records found in the current pass. If no deleted records were found during a pass, the pass will be terminated, and the message:

```
No deleted records found
```

will be displayed. If one or more deleted records were found, the message:

```
x deleted records found
```

would be displayed. Here, "x" is the number of deleted records in the current pass. T.I.M. will then perform the overhead necessary to preserve the sorted field orders. Next, T.I.M. will display the message:

```
Actually removing deleted records
```

Here, T.I.M. actually removes the deleted records from the data file. When the file compression process is completed, T.I.M. will display the message:

```
Original file size: x records
Compressed file size: y records
```

where "x" is the original file size before compression, and "y" is the current file size after the deleted records are removed.

It is important to make sure that a back-up copy of the file diskette exists before compression. This will prevent any loss of the user's data base due to a power outage during compression. For more information along these lines, consult section *0.3 Concept of a back-up*.

8.3 Display disk directory

If the user desires to display the **complete disk directory** of any particular disk, he should enter a **D**, and then enter the appropriate disk letter of the drive to be displayed.

8.4 Erase a key-field

If the user desires to eliminate one of the key-fields in a T.I.M. file, he should enter an **E.** T.I.M. will list the current key-fields and allow the user to erase one. This will remove the index file from the disk and update the T.I.M. directory, giving more disk space and substantially speeding up the record merging process.

This is sometimes desirable before compressing a file since it will greatly speed compression. Key-fields can then be added later using the *Sort a file* sub-command.

8.5 Kill a T.I.M. file

If the user wishes to delete a file from a disk, the *Kill a T.I.M. file* sub-command should be used. If a T.I.M. data file is "Killed" the corresponding entry in the T.I.M. directory will be deleted also. The user simply enters the filename. T.I.M. will then display the query:

```
Are you sure you want <filename> deleted from disk?
```

where "filename" is the name of the file to be deleted.

If the answer is "Y", T.I.M. will delete the file and display the message:

```
The file  <filename>  is deleted from disk
```

If the user enters any response other than the "Y", the message:

```
Cancelled
```

will be displayed and no action taken.

This is the only way that a T.I.M. file should be deleted.

8.6 reMove duplicate records

This command allows the user to report on or remove duplicate records from a T.I.M. file. T.I.M. will check for duplicate field entries in a key-field, and delete all except the first one found. All sorted orders (key-fields) will remain intact after this command is used.

After entering the **M**, T.I.M. will ask for the filename of the file to be worked on. T.I.M. will display the field list of this file and prompt:

```
Enter the key-field to search through (x=exit)
```

Before this sub-command will work, the user must enter the key-field number to search through. T.I.M. will then ask the user if he wishes to have a report detailing the duplicate records, or just default to the *reMove duplicate records* process. Then, T.I.M. will search down through the file by order of this key-field and look for duplicate entries in this field. These entries will be marked as deleted and may be removed later using the *Compress a file* sub-command.

Be sure to have a back-up diskette before you use this sub-command.

8.7 Password update

By entering a **P**, the user may change or eliminate the password that secures any T.I.M. file. First the user must enter the filename that is to be changed. Next, the user must enter the old password (if one exists) before it can be changed to a new one. The user will now be asked by T.I.M. to enter the new password. This password must be 4 characters long.

8.8 Rename a T.I.M. file

If the user desires to rename a T.I.M. data file, he should enter an **R**. The user must enter the appropriate filename of the file to be renamed. Next T.I.M. will ask the user to enter the new filename. The *Rename a T.I.M. file* sub-command will rename a T.I.M. data file in the T.I.M. directory, in the disk directory, as well as rename all index files associated with it.

This is the only way a T.I.M. file should be renamed.

8.9 Sort a file (add a key-field)

If the user wishes to sort a data file he should enter an **S** for the file *Maintenance* sub-command. This command is used to create a new key-field or re-construct a current key-field. T.I.M. will respond with the prompt:

```
Enter the name of the data file to sort (x=exit)
```

Here, an x brings you back to the *Maintenance* sub-command menu, and entering a filename continues the sort queries.

If a filename is entered, T.I.M. will display the field information of the data file. This is to assist the user in defining the key-fields.

The user is now ready to enter the key-fields.

```
Enter yyy sort key # x (cr=end)
```

Where "x" is the number of the sort key and "yyy" is either the word "major" or "minor". Up to 16 keys may be defined for each sort. The first key entered is always the major sort key while the second is the primary minor sort key. When the user is finished entering the sort keys he should press ↵ in response to the key query.

All data types are stored uniquely within the data file, in order that they may sort correctly. Note that Inverted name fields will sort in order of last name then first name.

T.I.M.'s sort requires a work area whose size depends on the length and number of defined key-fields of the file being sorted. T.I.M. will decide if there is enough work area in memory to sort the entire file. If there is not enough room in memory, T.I.M. will ask the user to specify a drive which will be used as a work area. T.I.M. will build a work file on this disk so that the file may be sorted. The user must be sure there is enough room on the disk before specifying it as a work disk.

If the user is unsure of the amount of unused disk space on his work disk, he may use an empty formatted disk as the work disk.

The message:

```
*** Sorting—do not interrupt ***
```

will be displayed while T.I.M. is working on the file.

The sorting of a data file is a two-step process, sorting small groups of records, and then merging all the small groups together. The program drive may be specified as the work drive. If this is done, T.I.M. will allow the user to remove the program diskette and insert the work diskette.

When sorting is complete, the merging process will begin. If the program drive contains the work file, T.I.M. will stop and wait for the user to place the program disk into drive A. The "MERGE" program will then load and T.I.M. will again ask the user to place the work disk into drive A, and T.I.M. will display the following message:

```
***Merging—do not interrupt***
```

If more than 250 records are being sorted, T.I.M. will display the record numbers as it merges them into final order.

When the merging process is complete, T.I.M. will return to the *Sort* sub-command prompt. If drive A was specified as the work drive, the user will first be asked to put the program disk back into drive A.

When the sort is finished, the *Sort* sub-command prompt will be displayed.

The *Sort* sub-command can be used to add a key-field to a file at any time.

8.10 Update field titles

This sub-command allows the user to change the title of any field in any one of his T.I.M. data files. As always, the field title cannot exceed 16 characters in length. Note that if a custom screen exists, it may have to be re-defined if the new field title is longer than the original.

8.11 eXit to main menu

When the user desires to end the *Maintenance* command, he should enter an **X**. This will return control to the main T.I.M. menu.

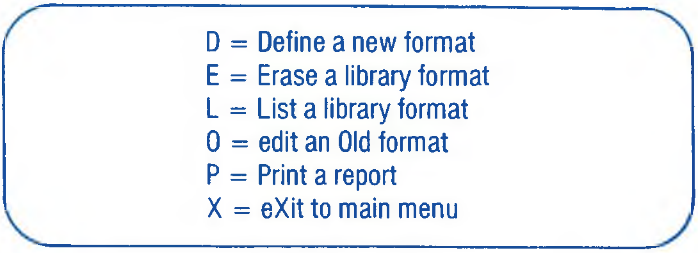
8.12 Points to remember

1.) You may use this command to alter the sequential count, display the complete disk directory, erase key-fields, kill & rename T.I.M. data files etc.

2.) You can compress T.I.M. files thereby eliminating all the deleted records in the file. This will yield more disk space.

9.0 Report generation

If the user wishes to print file information in a report format, he should use the *Report generation* command. The 6 *Report generation* sub-commands are listed below.



```
D = Define a new format
E = Erase a library format
L = List a library format
O = edit an Old format
P = Print a report
X = eXit to main menu
```

9.1 Define a new format

If the user wishes to define a new report format, he should enter a **D** when the *"Report"* sub-command menu is displayed.

9.1.1 Entering general report information

There are two groups involved in the general report information entries: the report titles and the print specifications. First, T.I.M. will prompt:

```
Enter title #n (cr=end)
```

Here, the user may enter up to 6 lines of report titles which will be centered at the top of the first page of the report. 'n' is the number of the title (1-6) that is being entered. These titles may be up to 40 characters in length.

Other than text, a slash (/) may be entered as a report title, and upon report printing, the current date will be inserted automatically. In addition, a double asterisk (**) may be entered as a report title and upon report printing, will allow the user to enter a title line. These two features give the Report Library true versatility, because you may store the report format, but variable information can be entered at the time of printing.

After the titles are entered, T.I.M. will prompt:

```
Enter the page width (cr=131)
```

The page width defaults to 131 characters, but widths of up to 255 characters may be defined.

Next, T.I.M. will display:

```
Enter the page length (cr=66)
```

The user now specifies the page length. This length will allow T.I.M. to paginate the report and rewrite the field titles on every page.

T.I.M. allows for error correction throughout the entire report definition process. If the user wishes to back-up and correct a previous entry, he needs only to enter an asterisk (*) for the current entry, and T.I.M. will back-up to the previous entry.

9.1.2 File and field definition

After the user has entered the overall report information, he must define which file and fields he wishes to have printed in the report. First, T.I.M. will prompt:

```
Enter the main filename (x=exit)
```

The user should enter the name of the T.I.M. file he wishes to print. If a linked file is also to be printed in the report, a second filename will be asked for later.

Now the user will define the fields he wishes to print in the main file. The field definition consists of defining the fields to print, and then defining the truncation lengths.

T.I.M. will ask for the fields to print as follows:

```
Enter field number n to print (#=Rec #,cr=end)
```

The user should enter the field number of each field that is to be printed across the page (left to right). After the field number is entered, T.I.M. will display:

```
Truncation length (cr=ln)
```

on the same line. Pressing [↵] will default to a truncation length of the field width. The "ln" in the above prompt designates the length of the field being defined. A truncation length other than the default value might be used to compress as much information as possible on, say, a 131 character line. The field definition process may be ended by pressing [↵] for the field number to print.

9.1.3 Entering fields to total

After the fields are defined, the user should enter the fields he wishes to total (if any). T.I.M. will prompt for these total fields as follows:

```
Enter field number n to total (cr=end)
```

The user may define up to 20 fields to total in each file. (There may be a main and a detail file used (but each field totaled must be either a number, dollar, calculated, sequential or total field. Pressing [↵] will end the total field definition process and an asterisk (*) will allow the user to back up to redefine a prior entry. (Note: On the displayed field list, T.I.M. will display a "p" to the left of any field which has been defined to be printed).

9.1.4 Entering the break-fields

T.I.M. will ask for break-fields as follows:

```
Enter the major break-field number (cr=none)
```

If a major break-field is entered, T.I.M. will prompt:

```
Skip line(s) or page after each break (l or p,cr=l)
```

Here the user has the option of skipping a line after each major break or skipping to the next page. If a line was specified after each break T.I.M. will prompt:

```
Enter the number of lines after the major break (cr=1)
```

This is the number of blank lines that will be printed after each major break.

Next T.I.M. will prompt:

```
Enter the minor break-field number (cr=end)
```

If a minor break is entered T.I.M. will prompt:

```
Enter the number of lines after the minor break (cr=1)
```

This is the number of blank lines that will be printed after each minor break. If you are not familiar with break-fields, they are explained as follows.

A break-field is the field by which summaries may be determined. For example, if a user had a data file which contained the field "NET SALES", he may wish to total the NET SALES field for each salesman, thereby allowing him to see how well each salesman was doing. An example of this is shown as follows.

```
    SALESMAN                                NET SALES
    ---------------------------------------------------
    JONES                                     $100.00
    JONES                                      200.00
    JONES                                       50.00
    JONES                                      100.00
                                   ----------------
                                    TOTAL     $375.00
                                   ================
                              GRAND TOTAL     $825.00
```

In this example, the major break-field is the SALESMAN field, with the NET SALES field being the only summarized field. The reason the SALESMAN field is called the break-field, is because the field (NET SALES) "breaks" after every different salesman, and is totaled at each break as well as at the end, to give the grand total.

T.I.M. also allows the user to summarize up to 20 different fields, which will allow the user to keep track and calculate information on many aspects of his data base. The previous example might be expanded to include net profit summarizations to give a report like the one that follows.

```
SALESMAN                    NET SALES    NET PROFIT
----------------------------------------------------
JONES                         $100.00        $25.00
JONES                          200.00         75.00
JONES                           50.00         20.00
JONES                          100.00         30.00
            ----------------------------  ----------
                       TOTAL  $450.00       $150.00

SMITH                          175.00         60.00
SMITH                          200.00         70.00
            ----------------------------  ----------
                       TOTAL  $375.00       $130.00
            ============================  ==========
                 GRAND TOTAL  $825.00       $280.00
```

It should be noted that the file in the previous example would need to have SALESMAN as a key-field to give the results in a coherent order.

The two previous examples incorporate one break-field, called the major break-field. Not only will the T.I.M. program give the user 20 possible summary fields, but it (T.I.M.) will allow the user to have up to two break-fields; called the major break-field and the minor break-field. This feature would allow the user to summarize fields by two breaks, thus allowing sub-totals, totals and grand totals. The previous example could be expanded to include districts, as well as salesmen, to give a more informative report. To do this, the file would have to be sorted by DISTRICT as the major sort key-field, and SALESMAN as the primary minor sort key-field. This way each salesman's sub-total could be added up by district to give district totals, with the sum of these district totals giving the grand total for all the entries in the file. (For additional examples of two or more summarized fields and two break fields, consult the example section of the Primer).

In addition to the set of field sub-totals, totals and grand totals, T.I.M. allows the user to print a record count, which keeps track of all records printed up to each break and/or the end of the report. This record count is printed automatically if breaks are used.

9.1.5 Entering key-fields

If break-fields are not entered, T.I.M. will ask the user to enter the key-field to print the report by. This prompt will be displayed as follows:

```
Enter the key-field to print by (cr=seq)
```

If a key-field is entered, a report will be printed by order of that field. For example, a report printed in order of smallest to largest zip code could be defined. Pressing ↵ will print the records in sequential order (the order which they were entered).

9.1.6 Entering link-fields

T.I.M. will now ask:

```
Enter the link-field if a second file will be used (cr=none)
```

The link-field option allows the user to tie two different files together based on a common field; the link-field. An example of this would be the user who has one file of customer information and another file of customer purchases, both linked through a common field. This “link-field” could be a customer I.D. number.

The link-field must be the same length and type in both files. Additionally, the link-field in the detail file must be a key-field. Refer to the primer for examples of linked files.

If a link-field is defined, T.I.M. will prompt:

```
Enter the detail filename (x=exit)
```

Here the user should enter the name of the second (detail) file which will be linked to the first (main) file. If this detail file is entered, T.I.M. will display the query:

```
Enter the link-field to link the detail file to the main file (cr=none)
```

This link-field must be common to both the main and detail files and as mentioned earlier, must be the same length and type. Once the link field is entered, the user must enter the fields to print and their truncation lengths. This section of entries is the same as those explained for the main file in section 9.1.2.

9.1.7 Entering the format into the Report Library

After the link-field query, T.I.M. will display all the information in the report format and display the prompt:

```
Save format, edit or abandon (s,e, or a,cr=s)
```

Here, the user has three options. First, the user may save the report format permanently in the Report Library. This is the usual procedure. Second, the user may edit his report format if he sees a mistake. This can be done repeatedly until the format is correct. Lastly, the user may abandon the format altogether. It will not be saved if abandoned.

If the user wishes to save the format, he should respond to the prompt by typing an **S**. T.I.M. will respond with:

```
Enter the library format name (20 chrs.,x=exit)
```

Here the user should enter the Library format name which will be used to recognize the format. This name must not exceed 20 characters in length. After the library entry, control will return to the "*Report*" sub-command menu.

9.2 Erase a library format

If the user decides that a particular Report Library format is no longer needed, it may be erased using this sub-command. After the user enters an **E**, T.I.M. will display each format in the report library and ask:

```
Enter format number to erase (x=exit)
```

The user should then enter the number of the format to be erased from the Report Library. After the number is entered, T.I.M. will prompt:

```
Are you sure (y or n )
```

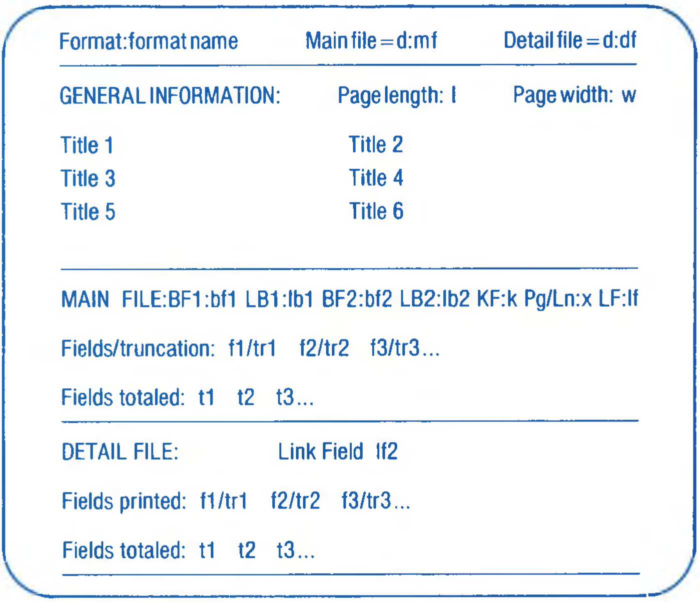
The user may then continue by erasing the file or aborting this sub-command.

9.3 List a library format

If the user wishes to examine and/or print all of the information stored in the Report Library of a given format he should use this sub-command. T.I.M. will prompt:

```
Enter format number to list (x=exit)
```

The user should enter the number which corresponds to the format he desires to list. The *List a library format* sub-command will display all the information contained in the Report Library concerning the format in question in the following manner:



```
Format:format name          Main file=d:mf          Detail file=d:df

GENERAL INFORMATION:          Page length: l          Page width: w

Title 1                       Title 2
Title 3                       Title 4
Title 5                       Title 6

MAIN  FILE:BF1:bf1 LB1:lb1 BF2:bf2 LB2:lb2 KF:k Pg/Ln:x LF:lf

Fields/truncation:  f1/tr1   f2/tr2   f3/tr3...

Fields totaled:  t1   t2   t3...

DETAIL FILE:              Link Field  lf2

Fields printed:  f1/tr1   f2/tr2   f3/tr3...

Fields totaled:  t1   t2   t3...
```

Here, "format name" is the name of the report format. "d" is the drive where "mf", the main file is located and "df" is the detail file (if used). "l" and "w" are the page length and width respectively. "title 1, title 2..." are the six lines of possible report titles.

"bf1" and "bf2" are the major and minor break-fields. (A zero will be displayed if this option is not used). "lb1" and "lb2" are the lines after each break. The key-field number to print by, is "k". "x" will either be "page" or "line" depending on which was desired at a defined break. "lf" is the link-field of the second file.

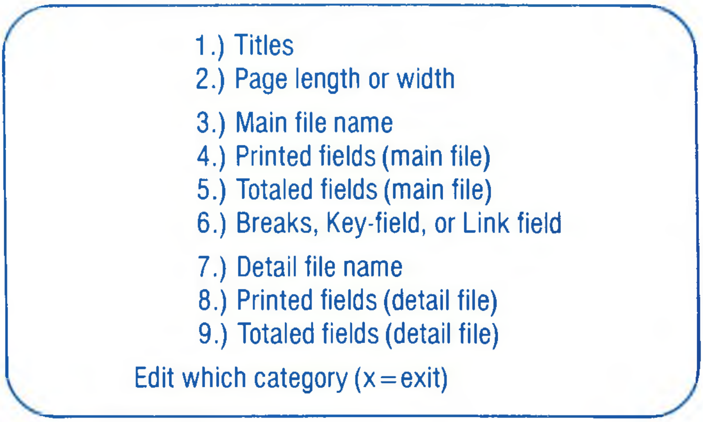
In both the main file and detail file sections, "f1,f2,f3,..." are the fields printed and "tr1,tr2,tr3,..." are the fields defined as total fields of each file. (NOTE: The whole "DETAIL FILE" section will be omitted if a detail file is not used. If the user wishes to print this report format information, he should simply type a Y when asked.

9.4 edit an Old format

The *edit Old format* sub-command allows the user to edit an existing report format. If the user invokes this sub-command, T.I.M. will list the formats contained in the Report Library and display the prompt:

```
Edit which format number (x=exit)
```

The user should enter the number corresponding to the format he wishes to edit, and then T.I.M. will display the following menu and prompt:



```
            1.) Titles
            2.) Page length or width

            3.) Main file name
            4.) Printed fields (main file)
            5.) Totaled fields (main file)
            6.) Breaks, Key-field, or Link field

            7.) Detail file name
            8.) Printed fields (detail file)
            9.) Totaled fields (detail file)
      Edit which category (x=exit)
```

Here the user enters the number corresponding to the category to be edited. This procedure may be repeated as often as desired before permanently resaving the format in the Report Library.

If the user decides to use the same library format name when resaving, T.I.M. will ask if he wishes to replace the existing format. If the answer is yes, the old library format will be replaced.

9.5 Print a report

When the user desires to print a report whose format is stored in the Report Library, he should type a **P** when T.I.M. asks for the "*Report*" sub-command.

Printing a report using a Report Library format is a very simple process. First, the user will enter the format number of the report he wants printed. He may first use the *List a library format* sub-command to determine which report format he wishes to print. Next T.I.M. will prompt:

```
Send to printer, terminal or disk (p,t,or d,cr=p,x=exit)
```

Here the user decides if he wishes hard copy from his printer, if he wishes the report printed on his terminal, or if he wishes to have the report written to an ASCII file on disk. This ASCII file can then be read by the users word processor and edited or printed. After the appropriate entry, T.I.M. will display the prompt:

```
Print totals only (y or n,cr=n)
```

This option allows the user to print only summary information, not data records. An example of this is a user who just wishes departmental totals to date, but does not need the detail of a full report. If the report is being sent to the printer, the program will prompt:

```
Pause between pages (y or n,cr=n)
```

If the user wishes to pause between each page he should type a **Y**. This is necessary if you are printing on single pages rather than continuous feed paper.

9.5.1 Aborting printing

After T.I.M. begins the printing of a report, the message:

```
Press "X" to stop printing
```

will appear on the user's terminal. If the user wishes to abort the report generation process and return to the "*Report*" sub-command menu, he should press and hold down the **X** key until the printing process is terminated. At this time, the user may abort the printing process completely, or resume when needed.

9.6 eXit to main menu

If the user wishes to exit the *Report generation* sub-command and return to the main T.I.M. menu, he should enter an **X**.

9.7 Points to remember

1.) The Report generation command will generate a tabular report of T.I.M. data.

2.) Fields will be automatically spaced across the page, with field titles and page numbers printed at the top of each page.

3.) Up to 20 fields may be totaled in one report using one file.

4.) Break-fields may be defined to group information so that sub-totals may be printed on a report.

5.) Report formats may be edited so that the entire report need not be defined again.

6.) Reports may be printed on the terminal, on paper or sent to a disk for word processor editing.

10.0 Select records from a file

The "*Select*" command is used to select certain groups of information from a complete data base. For example, the user may wish to select all people in a certain zip code area.

The *Select records* command has 6 sub-commands as shown below:

```
D = Define a new library format
E = Erase a library format
L = List a library format
R = Replace fields in selected records
S = Select records from a file
X = eXit to main menu
```

10.1 Define a new library format

The *Define a new library format* sub-command is used to define a record selection format. If the user wishes to use this sub-command, he should enter a **D**. T.I.M. will then query the user with:

```
Enter the origin data filename (x=exit)
```

The origin data file is the file we are going to select records from. After the filename has been entered, T.I.M. will display a field list of the origin file. Displayed under the field list will be the prompt:

```
Enter search criteria (x=exit, h=help, ?=fields)
```

This is an important prompt. By entering an H, a complete help menu is displayed which outlines the features of the "*Select*" search criterias. Typing a question mark (?) will redisplay the fields of the origin file. These features will be explained in the next 4 sections.

10.1.1 Defining the search criteria

The search criteria is one or more relational statements connected together with logical operators. Each relational statement, unless it is a "DEL" or "NONDEL" statement, must consist of three parts:

1.) A field number to instruct T.I.M. which field is being compared. This may also be a part of a field.

2.) A relational operator to define the intended relation. The relational operators defined are "greater than" (GT), "less than" (LT), "equal to" (EQ), "not equal to" (NE), "greater than or equal to" (GE), "less than or equal to" (LE), and "between two values" (BV).

3.) A data operand to be compared to the defined field. This operand can optionally be a second field.

10.1.1.1 Relational operators

The eight types of relational statements which T.I.M. supports are as follows:

1) DEL	All deleted records
2) NONDEL	All non-deleted records
3) FLD #×relation× ×constant×	Field relates to constant
4) FLD #×relation×FLD #	Field relates to field
5) FLD # BV×cl×:×c2×	Field between two values
6) SUBFLD # EQ×constant×	Part of field equals constant
7) NSUBFLD # EQ×constant×	No part of field equals constant
8) SUB[s/l]FLD # EQ×constant×	Part of field starting at "s" with length "l" equals constant

The first two relational statements are used to choose deleted or non-deleted records from the file. These relational statements may be used in longer search masks (e.g. NONDEL, AND FLD 1 EQ FLD 2,).

The third statement compares the contents of a field to a constant value (e.g. FLD 7LT 1000, OR FLD 10 EQ JOHN DOE).

The fourth statement compares the contents of two fields (e.g. FLD 2 LT FLD 3). Note: This relational statement may be used only if the two fields being compared are the same length.

The fifth relational statement compares a field to two values. If the field is between the values, then the statement is satisfied. For example:

```
FLD 1 BV 10:20
```

This statement would check to see if the contents of field 1 are between 10 and 20. Text comparisons may also be used as shown below:

```
FLD 3 BV ADAMS:SMITH
```

This statement would find all records in which the contents of field 3 are between "ADAMS" and "SMITH".

The sixth, seventh and eighth statements check to see if the field specified contains a certain constant. Examples are shown below:

1.) **SUBFLD 6 EQ b,**—This statement would find every record that contains a "b" anywhere in field 6.

2.) **NSUBFLD 1 EQ DOE,**—This statement would find all records that do not contain a "DOE" anywhere in field 1.

The eighth relational statement is similar to the sixth and seventh but includes the ability to search a specific part of a field. The start and length values specify where to start the search and how much to look at. This is shown below:

```
SUB[5/1 5]FLD 3 EQ TRUCK
```

This statement searches for the word "TRUCK" in field 3, starting at position 5 and searching the following 15 positions. Therefore the following fields would satisfy the search mask:

BIG TRUCK
LITTLE TRUCK WHEELS
RED FIRE TRUCK

while:

TRUCK
A TRUCK

would not, since the phrase "TRUCK" is not contained in the correct field position. Note: The SUBFLD or NSUBFLD using the start and length option may be used only on alphaııumeric fields.

These relational statements are further explained in the following sections.

Spaces serve as delimiters to separate the parts of the relational statement. An example of a relational statement is shown as follows:

```
FLD 4 GT 100.00
```

This particular relational statement instructs T.I.M. to find all records where the data in field number 4 is greater than (GT) $100.00. Notice that the dollar sign ($) is not entered into the search criteria.

10.1.1.2 Logical operators

Suppose we are looking for all overdue debts of over $100.00. This would be expressed as:

```
FLD 4 GT 100.00, AND FLD 5 LT 1/31/81
```

This search criteria contains two relational statements connected by the logical operator "AND". It instructs T.I.M. to find all records where the data in field 4 is greater than (GT) $100.00, AND, the data in field number 5 is less than (LT) 1/31/81. The logical operators that are available in T.I.M. are "AND" and "OR". These operators can be used to connect up to 10 relational statements together to form complicated search criteria. Commas must be used to separate the relational statements from the logical connector (as shown in the example).

A sub-field comparison may also be implemented. Suppose for example, that the user wanted to build a file of all non-deleted customers with a last name of FOLD. This could be done using the sub-field command as follows:

```
SUBFLD 2 EQ FOLD, AND NONDEL
```

To use the sub-field option simply put "SUBFLD" before the field number in the relational statement as shown above. It should be noted that all comparison data should be entered in the same case (upper or lower) as the data file entries, if the comparison data is alphanumeric in nature, rather than just numeric. The above example searches for all non-deleted records in which part of field 2 equals "FOLD".

As explained earlier, the sub-field option can also search parts of a field. For example, if the user wanted to check the first three characters in field 1 to see if they equalled "BOX", he could use the mask shown below:

```
SUB[1/3]FLD 1 EQ BOX, AND NONDEL
```

The numbers in the brackets indicate where to start looking inside the field and how many characters to check. In our example, the "1" means start comparing at the first character, while the "3" means compare the 3 characters after the start.

10.1.1.3 Dynamic data entry option

A final option implemented is a dynamic data entry mode for the search mask. This mode is especially useful for date comparisons. For instance, if the user wishes to bill his customers monthly he would need a format like:

```
NONDEL, AND FLD 5 LE 1/31/80,
```

This format would work for January but could not be saved in the Select Library and re-used in February because the date is incorrect. A way to circumvent this problem is to save the format in the Select Library without the date and enter the correct date each month. Two asterisks (**) in the data operand section of a relational statement allow the user to do this. Therefore, the above example would appear as:

```
NONDEL, AND FLD 5 LE **
```

Thus, every month, the user need only enter the library format number, and T.I.M. will ask the user to enter the date. This dynamic data entry option is not restricted to date fields only and may be used anytime a data operand changes with time.

10.1.1.4 Search criteria summary

There are a few important things to remember when defining a format:

1.) Since T.I.M. interprets the search criteria into another form it must be entered in a very structured format. Spaces must be used to delimit the relational operators. Commas must be used to delimit relational statements from logical operators as shown in the examples.

2.) The entire search criteria must not exceed 250 characters in length or a length error will be given by T.I.M. If a length error results, the file may be built in two stages using two shorter search criterias.

3.) The search criteria is interpreted and executed in a left-to-right sequence. Remember this when linking relational statements together to form complex search criteria.

4.) All relational statements which include the sub-field mode must use the equals (EQ) operator as it would not make sense to say a sub-field is greater than or less than some value.

A table of example search criteria are shown below:

SEARCH CRITERIA	EXPLANATION
SUB[2/1]FLD 1 EQ Y	The second character in field #1 equals the letter "Y"
NONDEL	All non-deleted records
NONDEL, AND FLD 3 BV 0:1000	All non-deleted records where field # is between 0 and 1000
SUBFLD 3 EQ AVE	All records where field #3 contains the characters "AVE"
FLD 1 GT 100, AND FLD 2 LT JOE	All records where field #1 is greater than 100 and, at the same time, field #2 comes before "JOE"
FLD 4 NE North	All records which do not have the word "North" in field #4
FLD 6 NE 10, OR FLD 7 NE 20	All records which have any entry except 10 in field #6 or any records which have any entry except 20 in field #7
FLD 9 NE NULL	All records which are not null (blank) in field #9

10.1.2 Entering the new format

Once this format is entered, T.I.M. will take apart and interpret the search criteria checking for errors. If an error is found, T.I.M. will indicate where it occured and what type of error it is. If no error is found, T.I.M. will display the fields of the origin file and display the prompt:

```
Enter the key-field to select by (cr#seq)
```

Here, the user should specify which order the file should be searched for records conforming to the search criteria.

After the key-field is specified, T.I.M. will clear the terminal and display the prompt:

```
Enter the format name (20 chrs.,x=exit)
```

The user should enter a name to identify the newly defined select criteria. After the name has been entered, control will return to the *Select records* sub-command menu.

10.2 Erase a library format

If the user decides he no longer needs a particular Select Library format, he may erase it using this sub-command. After the user enters an **E**, T.I.M. will ask:

```
Enter the format number to erase (x=exit)
```

The user should enter the number of the format to be deleted from the Select Library. After the number is entered, command will return to the *Select records* sub-command menu.

10.3 List a Library format

If the user wishes to list a specific format in the Select Library, he should enter an L. T.I.M. will query with:

```
Enter the library format number (x=exit)
```

The user should enter the format number of the format he wishes to see. The Select Library format will be displayed as follows:

```
FORMAT #   :  n            NAME        :  format name
DATA FILE  :  d:filename   KEY-FIELD   :  k

XXX
```

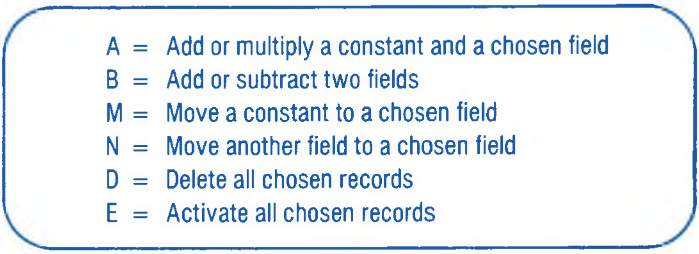
where "n" is the Select Library format number: "format name" is the given format name: "d:filename" is the drive letter and filename of the origin file; and "k" is the key-field number to search through. "XXX" is the actual selection criteria (eg. FLD 1 EQ 100).

10.4 Replace fields in selected records

By using this option, the user may make changes to records chosen from a file using the "*Select*" command. This sub-command is very useful when the user wishes to change, say, one or two fields in a great number of records. Examples of this might be a global record delete or activate, or perhaps finding all accounts overdue by 30 days and adding a 2% interest charge to the "AMOUNT DUE" field.

After any changes have been made, all calculated and total fields will be recomputed. Therefore changing an operand field of a calculated or total field will change the calculated and total fields also. No sequential, calculated, or total fields may be chosen as one of the fields to be changed. Date fields may be moved from one type of date to another (i.e. American to European) and the new dated converted accordingly.

The six methods of record change are displayed below and outlined as follows.



```
A = Add or multiply a constant and a chosen field
B = Add or subtract two fields
M = Move a constant to a chosen field
N = Move another field to a chosen field
D = Delete all chosen records
E = Activate all chosen records
```

10.4.1 Add or multiply a constant and a chosen field

This option will allow the user to add or multiply any field by a numeric constant. After choosing this option, T.I.M. will display a list of the fields of the file and ask which field is to be changed. Finally, T.I.M. will ask for the operator and constant to be used.

10.4.2 Add or subtract two fields

This option will allow the user to add or subtract the contents of any two numeric or dollar fields. This option is activated the same way as the previous option except that two fields are chosen.

10.4.3 Move a constant to a chosen field

This option allows the user to move a constant into any field of the selected record. The constant is verified in accordance with the type of data the chosen field contains. For example, an alphanumeric constant cannot be moved into a numeric field.

10.4.4 Move another field to a chosen field

This option allows the user to move the contents of one field into another. A numeric field may be moved into a dollar field and visa-versa, but all other moves must be of fields of the same type and length.

10.4.5 Delete all chosen records

This option will delete all records chosen by the select criteria.

10.4.6 Activate all chosen records

This option will activate (make deleted records non-deleted) all records chosen by the select criteria.

10.5 Select records from a file

When the user wishes to select records from a file, he should enter an S for the "*Select*" sub-command. T.I.M. will display a list of the formats stored in the Select Library and ask:

```
Enter the library format number (x=exit)
```

The user should enter the appropriate format number and then T.I.M. will prompt:

```
Use defined file named <d:filename> (y or n, cr=y)
```

Here, "d:filename" is the drive letter and filename of the file to be searched through. If the filename and drive are correct, the user should press ⏎. If the filename or drive has been changed, the user should enter an **N**, and T.I.M. will ask the user to re-enter the correct origin filename. After the file is chosen, T.I.M. will prompt:

```
Send output to terminal, disk, or printer (t,d,p,cr=t)
```

This is the destination of the selected records. These three options are explained next.

10.5.1 Sending output to terminal

If the user desires to send his "*Select*" output to the terminal, he should enter a **T** when asked. T.I.M. will then search the origin file for all records which satisfy the "*Select*" criteria. Everytime a record is found, it will be displayed on the terminal, and T.I.M. will display the prompt:

```
Continue, print, or exit (c,p, or x, cr=c)
```

At this time, the user may continue searching for more records, print the one being displayed, or exit the "*Select*" process.

10.5.2 Sending output to disk

If the user decides to create a new file of the records which conform to his selection criteria, he should enter a **D** when asked where to send the "*Select*" output. This new file of records will be a subset of the contents of the original file. T.I.M. will then ask:

```
Enter the destination filename (x=exit)
```

Here, the user should enter the name of the file where the selected records will be sent. T.I.M. will then query the user:

```
Enter comment (cr=none)
Enter a four-letter security code (cr=none)
```

These two queries are just like those asked when creating a new file using the "*Create*" command. After these two queries are answered, T.I.M. will search through the origin file and send any records to disk that conform to the search criteria. The prompt:

```
Press "X" to stop select
```

will be displayed while T.I.M. is searching the origin file. The search may be aborted by typing an **X**.

As the search is taking place the message:

```
Record n out of t with s selected
```

will be displayed on the terminal. Here, n is the record being searched, t is the total number of records in the file, and s is the total number of records which satisfy the search criteria.

When the selection process is over, the message:

```
n records put in file <filename>
```

will be displayed on the terminal. Here "n" is the number of records which did satisfy the selection criteria and were written into the file called "filename".

10.5.3 Sending output to printer

If a printed list of the "*Select*" output is desired, this option should be used. Once this option is chosen, the message:

```
Report format or list format (r or l,cr=r)
```

will be displayed. If a list format is chosen, each record selected will be printed in a vertical list fashion, starting with the first field printed on the first line and ending with the last field printed on the last line.

If a report format is desired, T.I.M. will allow the user to define up to 2 lines of titles (which will automatically be centered at the top of the page) and which fields he wishes to print on the report. The report option is an abridged version of the "*Report generation*" command. This option has no summaries, truncation lengths or breaks. The fields to print are asked for as follows:

```
Enter report field number n (cr=end)
```

Up to 16 fields may be defined in the "*Select*" report. During the report generation, you may exit by pressing an **X**. After the report has been generated, T.I.M. will print a record count of all records which were printed.

10.6 eXit to main menu

If the user wishes to exit the "*Select*" command, he may enter an **X**. This will return control to the main T.I.M. menu.

10.7 Points to remember

1.) The *Select records* command will go through an **entire** file and find all records that conform to the search criteria you wish.

2.) Up to 10 search criteria may be linked by using "AND"s and "OR"s.

3.) Once these records are selected, they may be sent to the terminal, to the printer or to a disk file.

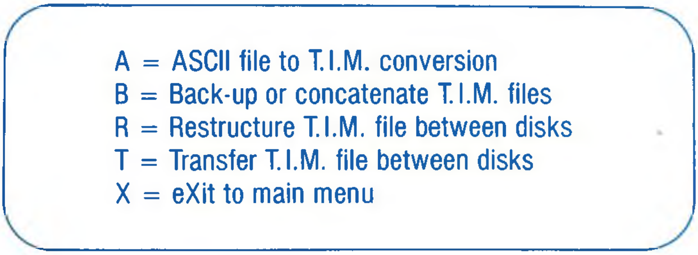
4.) If sent to the printer, the selected records may be output in a list or tabular report fashion.

5.) If sent to disk, a new file will be created and have just the selected records in it.

6.) Another option after the selection would be to do a global operation to the records. In other words, you may perform an operation on **every** record at once rather than updating them one at a time. Operations include addition, subtraction, multiplication, division, increment, decrement, global delete or activate.

11.0 T.I.M. Utilities

The *T.I.M. Utilities* are used to perform infrequent file operations. These are used less often than the maintenance commands. This command has 5 sub-commands as displayed below:



```
A = ASCII file to T.I.M. conversion
B = Back-up or concatenate T.I.M. files
R = Restructure T.I.M. file between disks
T = Transfer T.I.M. file between disks
X = eXit to main menu
```

11.1 ASCII file to T.I.M. conversion

This utility is designed to convert a standard ASCII file (fields delimited by commas, and each record ending in a carriage return and line feed.) into a T.I.M. file. To use this utility, the user should enter the destination filename (T.I.M. filename) and the origin filename and extension (ASCII filename). Next the record length and number of fields in the ASCII file must be entered. Finally, the user should specify which ASCII field is to be converted into which T.I.M. field. This is done in the same way as the *Restructure T.I.M. data file* utility explained in 11.4.

11.2 Back-up or concatenate T.I.M. files

The *Back-up or concatenate T.I.M. files* has two separate purposes. First, it can break down a very large T.I.M. file into smaller pieces for back-up purposes. This is especially useful for backing up a hard disk onto floppies. Second, it can combine T.I.M. files to create a larger file. This is not limited only to files created by the back-up sub-command.

Any two T.I.M. files can be combined if they are of the same format.

11.2.1 Back-up a file

To back-up a file you must enter three pieces of information; the source file, the destination file, and the disk capacity.

T.I.M. will prompt the user with:

```
Enter the name of the source file (x=exit)
Enter the drive (a—p,cr=b)
```

The user must enter the name of the file that is to be backed-up. An “x” will abandon this command.

Next, T.I.M. will prompt the user with:

```
Enter the destination filename (x=exit)
Enter the drive (a—p,cr=b)
```

The user now enters the name of the files and drive which will be transferred to the smaller disks. These files will all have the extension “.BAK” to distinguish them from regular T.I.M. files.

T.I.M. will then prompt the user with:

```
Enter the disk capacity in kbytes (eg 160,320,500)
```

The user must enter how large his disks are in kilobytes. This number allows the program to determine how many disks will be needed to back-up the file.

T.I.M. will now prompt:

```
Disk capacity in Kbytes: n1
File length in Kbytes: n2
Disks needed for back-up: n3

Press return to continue or X to exit
```

where "n1" is the diskette capacity which the user entered; "n2" is the file length of the source file; and "n3" is the number of disks the user will need to back-up his file. If the user wishes to continue he should hit return. An "x" will aabandon this sub-command.

If the user should continue, the prompt:

```
Put disk # n in drive d and press return
```

will be redisplayed for each destination used. "n" is the number of the back-up disk currently being written to and "d" is the drive of the destination file.

The program will then prompt with:

```
Record n1 out of n2
```

where "n1" is the record being backed-up and "n2" is the total number of records in the file.

Once the back-up is completed the disks should be stored safely. If the file should ever need to be restored using these back-ups, the "*Concatenate*" sub-command should be used.

It should be noted that no INDEX files are backed-up. Therefore, if a file is restored it must be re-sorted to reconstruct the key-fields.

11.2.2 Concatenate files

To combine two T.I.M. files into one larger file the "*Concatenate*" sub-command is used.

T.I.M. will then prompt the user with:

```
Enter the filename and extension (x=exit)
Enter the drive (a—p,cr=b)
```

Here the user must enter the filename **and extension**. If the user is restoring a T.I.M. file from the "*Back-up*" sub-command, the extension will be "BAK". If the user is combining two T.I.M. files into one T.I.M. file the extension would be "DAT". For example, you would enter "TEST.DAT" as the filename if you were adding the T.I.M. file "TEST" onto another T.I.M. file. An **x** will abort the sub-command.

T.I.M. will first prompt the user with:

```
Enter the destination filename (x=exit)
Enter the drive (a—p,cr=b)
```

This is the file which will be appended to. It must be a T.I.M. file which is already in existence. An **x** will abandon the sub-command.

Once the process is finished the prompt:

```
n records in file f
```

Where "n" is the total number of records in the destination file and "f" is the filename.

11.2.3 eXit to utility menu

By pressing **x**, the program will return to the "*Utility*" sub-command menu.

11.3 Restructure T.I.M. data file

This is a very powerful command used for restructuring the format of a T.I.M. file. For example, fields can be added, deleted, lengthened, shortened, or changed in type or precision. This is often necessary since all possible problems or applications cannot be foreseen at the definition stage.

In order for a file to be restructured, a second file must be created using the "*Create*" command. This second file will become the new file when the restructuring is completed. It should therefore contain all necessary fields. Once this new file is created, the user may procede with the "*Restructure*" sub-command.

T.I.M. will prompt the user with:

```
Enter the origin filename (x=exit)
Enter the drive (a—p,cr=b)
```

The user should enter the name and drive of the old file that is to be restructured. An **x** will abandon the sub-command.

T.I.M. will then prompt the user with:

```
Enter the destination filename (x=exit)
Enter the drive (a—p,cr=b)
```

The user should enter the name and drive of the new or restructured file. This file must already exist. Typing an **x** will abandon the sub-command. T.I.M. will now prompt the user with:

```
Put disks in appropriate drives and press return
```

The user should now make sure that the old file and the new file are in the coomputer. One of the files may replace the program disk if necessary.

T.I.M. will now display all the fields in the origin file on the user's screen. The user will now determine which field from the old file will go into which field in the new file. For each field in the new file T.I.M. will prompt the user with:

```
Enter the origin for <f> (cr=none)
```

The user will enter the field number of the old file to put into field "f" (where "f" is the field name) of the new file. If no origin field is desired, press ↵.

The program will now convert the old data file to the new format. The new file should then be checked to make sure it is correct before the old file is destroyed.

It should be noted that the program will attempt to make field type conversions if possible. For example, if an alphanumeric field containing a number is put in a numeric field, the conversion will take place correctly. Conversely, if a date field is put into a numeric field, the numeric field would end up with a zero in it.

11.4 Transfer a T.I.M. file between disks

Since each T.I.M. file on a disk has associated header information in a separate file, files cannot be merely copied using the operating system utilities (such as copy). This sub-command allows for the transfer of T.I.M. files and all associated information.

T.I.M. will prompt the user with:

```
Enter the origin filename (x=exit)
Enter the drive (a—p,cr=b)
```

The user should enter the name of the file and the drive to transfer. Pressing an **x** will abandon this sub-command.

The records and all associated file information will then be transferred to the destination file.

This is the only method that should be used to move T.I.M. files.

11.5 eXit to main menu

When the user desires to end the "*Utilities*" command, he should enter an **X**. This will return control to the main T.I.M. menu.

11.6 Points to remember

1.) ASCII files may be converted to T.I.M. files.

2.) Large files (like on hard disks) may be backed-up onto several floppies.

3.) Multiple T.I.M. files may be hooked together (concatenated) if they have the same file structure.

4.) T.I.M. files may be restructured (i.e. fields added, deleted, lengthened or shortened) without having to retype the entire file again.

5.) Files may be transferred (copied) to other disks. This is the only acceptable way to transfer T.I.M. files.

12.0 eXit to operating system

To return to the operating system, the user should enter an **X** when the main T.I.M. command menu is displayed.

A-1.0 T.I.M. file structure

This section gives a brief overview of the T.I.M. file structure. It is not necessary to read this section unless the user wishes to interface a custom program to a T.I.M. file.

There are three parts to a T.I.M. file: 1.) The directory which contains file description information 2.) The data file which contains the actual data records 3.) The index file which describes how to access the file in sorted order. These three files are explained in detail below.

A-1.1 The T.I.M. directory

There is a T.I.M. file directory on every T.I.M. data disk. This file is named TIMDIR.DAT and serves two major purposes: 1.) It tells T.I.M. what data files are on a disk 2.) It describes each file. This description includes the field titles, lengths and types; the calculations of all calculated fields, which fields are keys, as well as general file information (eg. record length, number of records in the file, number of records not yet merged, etc.).

Each T.I.M. directory contains a description for all of the data files on that particular disk. The first record of the directory contains the number of T.I.M. files represented in that directory. Each T.I.M. file is represented by a group of 14 records containing particular file information. Refer to the source listings at the end of this appendix for more information.

A-1.2 The data file

Each data file is made up of fixed length records. These records can be anywhere from 2 to 2400 characters in length. The last character in the record is the delete flag and is either an 'N' for non-deleted or a 'D' for deleted.

Each field in the record is of a fixed length and the data is either left or right justified depending on the field type as explained below:

1.) Number field	Right justified
2.) Dollar field	Right justified
3.) Calculated field	Right justified
4.) Total field	Right justified
5.) Sequential field	Right justified
6.) Alphanumeric field	Left justified
7.) Inverted name field	Left justified
8.) Date field	Left justified

A delimiter is also placed in the field if it does not fill the entire field length. This is to speed in the stripping of spaces once the record is retrieved. This delimiter is an ASCII 7C Hex (the vertical bar). It is on the left of the number, dollar, calculated, total, and sequential fields, and on the right of the alphanumeric, inverted, and date field.

Data may be inverted in the field depending on what type of field it is. The various inversions are explained below:

DATE FIELD—Date fields are always stored in the form yy/mm/dd with leading zeros if necessary. To restore this date into the American format just flip the year digits to the end.

INVERTED NAME FIELD—Inverted name fields are stored last name and then first name. For example, "John Doe" would appear as "Doe John" in the file. To re-invert this name to normal just search for the first space from the left and flip the two halves.

A-1.3 Index files

Index files are used when accessing a T.I.M. file in some sorted order. For every key-field in a file there will be an index file. The name of the index file will be the same as the data file with the extension '.IX?' where the question mark will be a number or letter. This symbol is dependent on which field the index file represents. For example, if field #1 of the file TEST is a key-field, there will be an index file named TEST.IX1. Below are the symbols which represent the extension additions for all 36 fields:

123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ

The index file itself is made up of records which are 254 bytes in length. The first record contains the minor key-field numbers used for that particular key-field. All records after the first contain 127 two byte integers representing record numbers. These record numbers are used to access the main data file in order by that particular key.

To obtain more specific information refer to the source listings at the end of this appendix. These source listings are written in MicroSoft BASIC and are copyrighted. If you use these subroutines in your programs you should contact Innovative Software, Inc.

```
1000 '
1010 ' T.I.M. (C) COPYRIGHT 1981-1982—Innovative Software, Inc.
1020 '
1030 '
1040 '***********************************************************************
1050 '
1060 ' Following is a group of subroutines used to access T.I.M. III
1070 ' files.  This includes the ability to read from a file in sequential
1080 ' or sorted order and how to write to a file.  Remember that all
1090 ' fields are justified according to their type (either right or
1100 ' left).
1110 '
1120 '***********************************************************************
1130 '
1140 DEF FNA$(X)=RIGHT$(STR$(X),LEN(STR'(X))-1)
1150 DEF FNL$(X$)=LEFT$(X$,INSTR(X$,CHR$(124))-1)
1160 '
1170 DEFINT A-Z
1180 '
1190 '***********************************************************************
1200 '
1210 ' --- the dimensions are explained below ---
1220 '
1230 ' a$() = holds field titles
1240 ' b$() = holds field data after record is read or before
1250 '        record is written
1260 ' c$() = holds three byte field types as follows:
1270 '        byte 1 - field type (aldsn$ct)
1280 '        byte 2 - a space in all field types except total fields
1290 '                 where it is a 0,1 or 2 depending on which
1300 '                 total field it represents.
1310 '        byte 3 - in numeric type fields (n$stc) this byte represents
1320 '                 the decimal precision of the number (0-4).  In
1330 '                 date fields it will be an A or an E depending if
1340 '                 it is an American date of a European Date.
1350 ' d$() = field statement used to read or write a record
1360 ' l()  = field lengths (1-60)
1370 ' v$() = calculated field equations in the form:
1380 '        <destination = fld1 operator fld2> or
1390 '        <destination = fld1 operator C constant>
1400 ' slc() = screen position of a field in integer format.
1410 '         the tens of thousands is the screen (1 or 2)
1420 '         the hundreds is the line number (1-20)
1430 '         the tens is the column number (1-80)
1440 '         (eg. 11210 is screen 1, line 12, column 10
1450 ' tot$() = total field operands in the form fld1,fld2,fld3
1460 '
1470 '***********************************************************************
1480 '
1490 DIM A$(39),B$(39),C$(39),D$(11),L(39),V$(19),SLC(39),TOT$(2)
1500 '
1510 GOSUB 3930:'get system parameters (including cursor controls
1520 '           erase screen, screen attributes, etc.)
```

```
1530 '
1540 INDEX$="123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ!@#+-":'index filename
1550 '
1560 WIDTH 255:WIDTH LPRINT PW
1570 '
1580 '***********************************************************************
1590 '
1600 ' This short section of code gets a filename which is returned in
1610 ' F$ with drive DR$.  It then looks to see if the file exists
1620 ' and if not returns an error.  If so it gets the headers
1630 ' including field titles, lengths, and types.
1640 ' If the file is confidential, and access code is asked for.
1650 '
1660 '***********************************************************************
1670 '
1680 GOSUB 3260:'GET FILE
1690 IF LEN(F$)=0 THEN END:'no name entered
1700 GOSUB 4480:'SEARCH
1710 IF REC>0 GOTO 1730
1720   PRINT:PRINT "File not found in T.I.M. directory":
       PRINT:GOTO 1680
1730 IF P$<>"0000" THEN GOSUB 5640:'get access code
1740 RESET
1750 '
1760 ' OPEN DATA FILES
1770 '
1780 '***********************************************************************
1790 '
1800 ' This routine opens a file with record length RW (record width) which
1810 ' came from the headers.  It is fielded using the field lengths in
1820 ' groups.  When a record is read the data will be in d$() and must be
1830 ' unbundled.  The GET RECORD subroutine does this.
1840 '
1850 '***********************************************************************
1860 '
1870 OPEN "R",1,DR$+":"+F$+".DAT",RW
1880 FOR I=0 TO 3
1890  A1=A1+L(I):A2=A2+L(I+4):A3=A3+L(I+8):A4=A4+L(I+12):A5=A5+L(I+16)
1900 NEXT I
1910 FIELD 1,L(0)+L(1)+L(2)+L(3) AS D$(1),L(4)+L(5)+L(6)+L(7) AS D$(2),
  L(8)+L(9)+L(10)+L(11) AS D$(3),L(12)+L(13)+L(14)+L(15) AS D$(4),
  L(16)+L(17)+L(18)+L(19) AS D$(5)
1920 FIELD 1,A1 AS D1$,A2 AS D2$,A3 AS D3$,A4 AS D4$,A5 AS D5$,L(20)+L(21)+
  L(22)+L(23) AS D$(6),L(24)+L(25)+L(26)+L(27) AS D$(7),L(28)+L(29)+L(30)+
  L(31) AS D$(8),L(32)+L(33)+L(34)+L(35) AS D$(9),L(36)+L(37)+L(38)+
  L(39) AS D$(10),1 AS D$(11)
1930 '
1940 '
1950 ' REINVERT DATA FIELDS
1960 '
1970 '***********************************************************************
1980 '
1990 ' This routine reinverts data read from a record.  Data is inverted
```

```
2000 ' so it will sort correctly.  The data is sent in C$ with its
2010 ' type in C1$ and its precision or date type in P1$.
2020 '
2030 '***********************************************************************
2040 '
2050 B=LEN(C$):A=INSTR(C$,CHR$(124)):IF A=0 GOTO 2070
2060   IF INSTR("ADI",C1$)>0 THEN C$=LEFT$(C$,A-1) ELSE C$=MID$(C$,A+1)
2070 IF INSTR("AN$CTS",C1$)>0 THEN RETURN
2080 IF C1$="D" GOTO 2130
2090   B=LEN(C$):A=INSTR(C$," "):IF A=0 THEN RETURN
2100   C$=RIGHT$(C$,B-A)+" "+LEFT$(C$,A-1)
2110   RETURN
2120 '
2130 IF P1$="A" THEN C$=MID$(C$,3,6)+LEFT$(C$,2):'AMERICAN DATE
2140 IF P1$="E" THEN C$=RIGHT$(C$,3)+MID$(C$,3,3)+LEFT$(C$,2):'EUROPEAN
2150 RETURN
2160 '
2170 ' INVERT FIELD
2180 '
2190 '***********************************************************************
2200 '
2210 ' This routine does the opposite of the above.  It inverts data before
2220 ' it is written to disk.  C$ contains the data, TYP$ contains the
2230 ' field type, P is the numeric precision, and P1$ holds the date type.
2250 '
2260 '***********************************************************************
2270 '
2280 R=INSTR(C$,CHR$(95)):IF R=0 THEN R=LEN(C$)+1
2290 C$=LEFT$(C$,R-1)
2300 IF TYP$="A" THEN RETURN
2310 IF TYP$="D" GOTO 2420
2320 IF INSTR("N$CTS",TYP$)>0 GOTO 2490
2330 '
2340 Q=LEN(C$):R=Q:IF R<1 THEN R=1
2350 WHILE MID$(C$,R,1)<>" " AND R>1
2360   R=R-1
2370 WEND
2380 IF R<=2 THEN RETURN
2390 C$=RIGHT$(C$,Q-R)+" "+LEFT$(C$,R-1)
2400 RETURN
24
2420 R=INSTR(C$,"/"):S=INSTR(R+1,C$,"/"):M=LEN(C$)
2430 IF S-R<3 THEN C$=LEFT$(C$,R)+"0"+RIGHT$(C$,M-R)
2440 IF R<3 THEN C$="0"+C$
2450 IF P1$="A" THEN C$=RIGHT$(C$,2)+LEFT$(C$,6)
2460 IF P1$="E" THEN C$=RIGHT$(C$,2)+MID$(C$,4,2)+"/"+LEFT$(C$,2)+"/"
2470 RETURN
2480 '
2490 GOSUB 2510:'ADJUST PREC
2500 RETURN
2510 '
2520 ' ADJUST NUMERAIC PRECISION
2530 '
2540 '***********************************************************************
2550 '
```

```
2560 ' This routine adjusts the number of decimal places of a number in
2570 ' the string C$ with P as the decimal precision (0-4).
2580 '
2590 '**********************************************************************
2600 '
2610 WHILE LEFT$(C$,1)=" "
2620   C$=RIGHT$(C$,LEN(C$)-1)
2630 WEND
2640 D=INSTR(C$,"."):IF D=0 THEN D=LEN(C$)+1:C$=C$+"."
2650 C$=LEFT$(C$,D+P):D=LEN(C$)-D:D=P-D
2660 C$=C$+STRING$(D,"0")
2670 IF P=0 THEN C$=LEFT$(C$,LEN(C$)-1)
2680 IF LEN(C$)=0 THEN C$="0"
2690 RETURN
2700 '
2710 ' WRITE RECORD
2720 '
2730 '**********************************************************************
2740 '
2750 ' This routine writes a record whose data is contained in B$().
2760 ' L() contains the field lengths and C$() contains the field types.
2770 ' DEL$ is the delete flag either N or D.  CURREC is the record
2780 ' to write to.
2790 '
2800 '**********************************************************************
2810 '
2820 L=0
2830 FOR J=1 TO 10
2840  G$=""
2850  FOR K=1 TO 4
2860   IF L>FLDS-1 GOTO 2920
2870   B$=SPACE$(L(L)):TYP$=LEFT$(C$(L),1):P1$=RIGHT$(C$(L),1):P=VAL(P1$)
2880   C$=B$(L):GOSUB 2160:'INVERT
2890   IF INSTR("N$CTS",TYP$)>0 AND LEN(C$)<L(L) THEN C$=CHR$(124)+C$
2900   IF INSTR("AID",TYP$)=0 THEN RSET B$=C$
       ELSE LSET B$=C$+CHR$(124)
2910   L=L+1:G$=G$+B$
2920  NEXT K
2930  LSET D$(J)=G$
2940 NEXT J
2950 LSET D$(11)=DEL$:'DELETE FLAG
2960 PUT 1,CURREC
2970 RETURN
2980 '
2990 ' GET RECORD
3000 '
3010 '**********************************************************************
3020 '
3030 ' This subroutine gets a record from a file and returns the fields
3040 ' in B$().  The delete status is in DEL$ and the record to get is
3050 ' sent in CURREC.  L() contains the field lengths and C$()
3060 ' contains the field types.
3070 '
3080 '**********************************************************************
3090 '
```

```
3100 GET 1,CURREC
3110 L=0
3120 FOR J=1 TO 10
3130  M=1
3140  FOR K=1 TO 4
3150   IF L>FLDS-1 GOTO 3220
3160   C1$=LEFT$(C$(L),1)
3170   P1$=RIGHT$(C$(L),1)
3180   C$=MID$(D$(J),M,L(L))
3190   GOSUB 1940:'REINVERT
3200   B$(L)=C$
3210   M=M+L(L):L=L+1
3220  NEXT K
3230 NEXT J
3240 DEL$=D$(11):'DELETE FLAG
3250 RETURN
3260 '
3270 ' GET FILENAME
3280 '
3290 '**********************************************************************
3300 '
3310 ' This routine returns a filename in F$ and drive in DR$.  F$ will
3320 ' be returned empty if no file is entered.  The routine expects
3330 ' available drives in ADR$ (from the headers) and default drive
3340 ' in DDR$ (also from the headers).
3350 '
3360 '**********************************************************************
3370 '
3380 PRINT "Enter the filename (x=exit) ";
3390 LINE INPUT F$:IF F$="X" OR F$="x" OR LEN(F$)=0 THEN F$="":RETURN
3400 IF LEN(F$)>8 GOTO 3380
3410 IF DDR$<>"V" THEN DR$=DDR$:RETURN:'DEFAULT DRIVE
3420 PRINT "Enter the drive (a-p,cr=b) ";
3430 DR$=INPUT$(1):IF ASC(DR$)=13 THEN DR$="B"
3440 DR$=CHR$(ASC(DR$) AND 95):PRINT DR$:IF INSTR(ADR$,DR$)=0 GOTO 3420
3450 RETURN
3460 '
3470 ' WRITE SCREEN
3480 '
3490 '**********************************************************************
3500 '
3510 ' This routine writes a record on the screen in the defined format.
3520 ' SCR contains which screen to display.
3530 '
3540 '**********************************************************************
3550 '
3560 PRINT Z$;:'ERASE SCREEN
3570 '
3580 FOR I1=0 TO 39
3590   IF I1>=FLDS GOTO 3680
3600   IF SCR=1 AND INT(SLC(I1)/10000)=2 GOTO 3680
3610   IF SCR=2 AND INT(SLC(I1)/10000)=1 GOTO 3680
3620 '
3630   L=VAL(MID$(STR$(SLC(I1)),3,2))+2:'LINE
3640   C=VAL(MID$(STR$(SLC(I1)),5,2)):'COL
```

```
3650   GOSUB 3720:'CURSOR
3660 '
3670   PRINT A$(I1);B$(I1);STRING$(L(I1)-LEN(B$(I1))," ");
3680 NEXT I1
3690 L=23:C=1
3700 GOSUB 3720:'address cursor
3710 RETURN
3720 '
3730 ' CURSOR POSITIONING ROUTINE
3740 '
3750 '**********************************************************************
3760 '
3770 ' This routine addresses the cursor.  It expects the line in L and
3780 ' the column in C.  Also the GET PARAMETERS routine must have
3790 ' already been accessed.
3800 '
3810 '**********************************************************************
3820 '
3830 A$=L11$:L1=L+L0:C1=C+C0
3840 IF L1<BL0 THEN L1=L1+OL0
3850 IF C1<BC0 THEN C1=C1+OC0
3860 IF LCR=1 THEN SWAP L1,C1
3870 IF BAF=0 THEN A$=A$+CHR$(L1) ELSE A$=A$+FNA$(L1)
3880 A$=A$+L12$
3890 IF BAF=0 THEN A$=A$+CHR$(C1) ELSE A$=A$+FNA$(C1)
3900 A$=A$+L13$
3910 PRINT A$;
3920 RETURN
3930 '
3940 ' GET SYSTEM PARAMETERS
3950 '
3960 '**********************************************************************
3970 '
3980 ' This routine gets all of the system related parameters from the disk.
3990 ' These are listed below:
4000 '
4010 '    Z$=Screen erase,  DDR$=Default drive,  PW=Printer width
4020 '    ADR$=Available drives,  TDAT$=Today's date
4030 '
4090 '    LAR$=Left arrow key,  RAR$=Right arrow key,  UAR$=Up arrow key
4100 '    DAR$=Down arrow key,  CDL$=Field delete key
4110 '    CIN$=Character insert key,  CRD$=Character delete key
4120 '    LFL$=Last field insert key,  TDT$=Today's date insert key
4130 '    HLP$=Help key,  EXT$=Exit key,  INI$=Tab key
4140 '
4150 '**********************************************************************
4160 '
4170 OPEN "R",1,"TIMPARAM.DAT",128
4180 FIELD 1,16 AS A1$,16 AS A2$,40 AS A3$,2 AS A4$,6 AS A5$,6 AS A6$,
     6 AS A7$,6 AS A8$,2 AS A9$,2 AS A10$,16 AS A11$,8 AS A12$
4190 GET 1,1
4200 Z$=CHR$(12)
4210 DDR$=LEFT$(A2$,1)
4240 TW=CVI(A9$):PW=CVI(A10$):ADR$=A11$
4250 J=INSTR(A12$," "):IF J=0 THEN J=9
```

```
4260 TDAT$=LEFT$(A12$,J-1)
4270 '
4360 FIELD 1,1 AS A1$,1 AS A2$,1 AS A3$,1 AS A4$,1 AS A5$,1 AS A6$,
     1 AS A7$,1 AS A8$,1 AS A9$,1 AS A10$,1 AS A11$,1 AS A12$,10 AS A13$,
     6 AS A14$,6 AS A15$
4370 GET 1,2
4380 LAR$=A1$:RAR$=A2$:UAR$=A3$:DAR$=A4$:CDL$=A5$:CIN$=A6$
4390 CRD$=A7$:LFL$=A8$:TDT$=A9$:HLP$=A10$:EXT$=A11$:INI$=A12$
4410 OLO=CVI(MID$(A13$,1,2)):BLO=CVI(MID$(A13$,3,2))
4420 OCO=CVI(MID$(A13$,5,2)):BCO=CVI(MID$(A13$,7,2))
4430 FIELD 1,121 AS A1$,2 AS A2$
4440 GET 1,3
4450 PNT=VAL(A2$):'PRINTER SELECTION
4460 CLOSE 1
4470 RETURN
4480 '
4490 'SEARCH T.I.M. DIRECTORY FOR A FILE
4500 '
4510 '**********************************************************************
4520 '
4530 ' This routine searches for a T.I.M. file.  It expects the filname in
4540 ' F$ with no extension and the drive in DR$.  If the file is found
4550 ' the headers are also retrieved and REC contains the header position
4560 ' of the file.  If REC is 0 then the file does not exist.
4570 '
4580 '**********************************************************************
4590 '
4600 REC=0
4610 FOR I=1 TO LEN(F$):'UPPER CASE
4620  IF ASC(MID$(F$,I,1))>64 THEN MID$(F$,I,1)=CHR$(ASC(MID$(F$,I,1)) AND 95)
4630 NEXT I
4640 F$=F$+SPACE$(8-LEN(F$))
4650 OPEN "R",1,DR$+":TIMDIR.DAT"
4660 FIELD 1,2 AS A1$
4670 GET 1,1
4680 IF EOF(1) THEN FLS=0:CLOSE 1:RETURN
4690 '
4700 FIELD 1,2 AS A1$,2 AS A2$,2 AS A3$,2 AS A4$,2 AS A5$
4710 GET 1,1
4720 FLS=CVI(A5$)
4730 '
4740 FIELD 1,8 AS N1$
4750 J=2
4760 FOR I=1 TO FLS
4770   GET 1,J
4780   IF N1$=F$ THEN REC=J:TREC=J
4790   J=J+14
4800 NEXT I
4810 IF REC=0 THEN CLOSE 1:RETURN
4820 '
4830 ' GET FILE HEADERS
4840 '
4850 '**********************************************************************
4860 '
4870 ' The parameters returned by the GET HEADERS routine are
```

```
4880 ' explained in the dimension section of this listing.  Those not
4890 ' explained are listed below:
4900 '
4910 '   NM$=Filename                    RW=Record width
4920 '   SQ=Sequential fld count         FLDS=Number of fields
4930 '   RECS=Number of records          LAD$=Last date records were added
4940 '        merged into file
4950 '   LUD$=Last date updated          P$=Password (0000 if none)
4960 '   VER$=Version number             TRECS=Number of records not merged
4965 '   PT$=Key-field flags (0 or 1)
4970 '
4980 '**********************************************************************
4990 '
5000 FIELD 1,8 AS A1$,2 AS A2$,2 AS A3$,2 AS A4$,2 AS A5$,4 AS A6$,
     4 AS A7$,8 AS A8$,60 AS A9$,1 AS A10$,2 AS A11$,8 AS A12$,8 AS A13$
5010 GET 1,REC
5020 NM$=A1$:RW=CVI(A2$):SQ=CVI(A3$):FLDS=CVI(A4$):RECS=CVI(A5$):LAD$=A12$
5030 LUD$=A13$:P$=A6$:VER$=A7$:TNS$=A10$:TRECS=CVI(A11$)
5040 '
5050 FIELD 1,128 AS A1$
5060 GET 1,REC+1
5070 K=0:GOSUB 5530:'TITLES
5080 '
5090 GET 1,REC+2
5100 K=8:GOSUB 5530:'TITLES
5110 '
5120 GET 1,REC+3
5130 K=16:GOSUB 5530:'TITLES
5140 '
5150 GET 1,REC+4
5160 K=24:GOSUB 5530:'TITLE
5170 '
5180 GET 1,REC+5
5190 K=32:GOSUB 5530:'TITLE
5200 '
5210 FIELD 1,80 AS A1$,40 AS A2$
5220 GET 1,REC+6
5230 FOR I=1 TO 40
5240   L(I-1)=CVI(MID$(A1$,(I*2)-1,2))
5250 NEXT I
5260 PT$=A2$:'KEYS
5270 '
5280 FIELD 1,120 AS A1$
5290 GET 1,REC+7
5300 FOR I=1 TO 40
5310   C$(I-1)=MID$(A1$,(I*3)-2,3)
5320 NEXT I
5330 '
5340 FIELD 1,120 AS A1$
5350 GET 1,REC+8:G$=A1$
5360 GET 1,REC+9:H$=A1$
5370 FOR I=1 TO 10
5380   V$(I-1)=MID$(G$,(I*12)-11,12)
5390   V$(I+9)=MID$(H$,(I*12)-11,12)
5400 NEXT I
```

```
5410 '
5420 FIELD 1,80 AS A1$,48 AS A2$
5430 GET 1,REC+10
5440 FOR I=1 TO 40
5450   SLC(I-1)=CVI(MID$(A1$,(I*2)-1,2))
5460 NEXT I
5470 TOT$(0)=FNL$(A2$):'total field #1
5480 '
5490 FIELD 1,48 AS A1$,48 AS A2$
5500 GET 1,REC+11
5510 TOT$(1)=FNL$(A1$):TOT$(2)=FNL$(A2$):'total fields 2 and 3
5520 CLOSE 1:RETURN
5530 '
5540 ' DISASSEMBLE TITLES
5550 '
5560 FOR I=1 TO 8
5570   A$(I+K-1)=MID$(A1$,(I*16)-15,15)
5580   J=INSTR(A$(I+K-1),"¶"):IF J=0 THEN J=16
5590   A$(I+K-1)=LEFT$(A$(I+K-1),J-1)
5600  B$=STR$(I+K):B$=RIGHT$(B$,LEN(B$)-1)+"  "
5610  A$(I+K-1)=LEFT$(B$,3)+A$(I+K-1)
5620 NEXT I
5630 RETURN
5640 '
5650 ' GET ACCESS CODE
5660 '
5670 '**********************************************************************
5680 '
5690 ' This routine accepts an access code and checks to see if it is
5700 ' the proper one.  If not, it gives the user three tries and then
5710 ' returns to the system.  The access code is expected in P$.
5720 '
5730 '**********************************************************************
5740 '
5750 G=1:PRINT "This is a confidential file. Enter the access code ";
5760 WHILE G>0
5770   C$="":FOR H=1 TO 4
5780     C$=C$+INPUT$(1):PRINT"*";
5790   NEXT H
5800   IF C$=P$ THEN G=0:GOTO 5830
5810   G=G+1:IF G=4 THEN END
5820   PRINT:PRINT "Error ! Enter access code again ";
5830 WEND
5840 PRINT
5850 RETURN
5860 '
5870 '*********************************************************************
5880 '
5890 ' This routine opens an index file so records may be retrieved in
5900 ' sorted order.  The variable 'INDEX' holds what key field to
5910 ' open.  This routine whould be placed after the open data file
5920 ' routine.
5930 '
5940 '*********************************************************************
5959 '
```

```
5960 IF MID$(PT$,INDEX,1)="1" GOTO 5990:'SEE IF KEY-FIELD
5970   PRINT"ERROR - THIS IS NOT A KEY-FIELD"
5980   ERR=1:RETURN
5990 D$=MID$(INDEX$,INDEX,1)
6000 OPEN "R",3,DR$+":"+F$+".IX"+D$,254
6010 FIELD 3,254 AS P$
6020 RETURN
6030 '
6040 '*********************************************************************
6050 '
6060 ' This routine returns the actual record number of a record in
6070 ' sorted order.  You send the record number you want in order
6080 ' (for example, the 10th one in order) in the variable 'RELREC'
6090 ' and the actual record number is returned in 'CURREC' which
6100 ' can then be used by the 'GET RECORD' subroutine.
6110 '
6120 '*********************************************************************
6130 '
6140 R1=INT(RELREC/127)+SGN(RELREC MOD 127):'WHICH INDEX RECORD WE NEED
6150 IF R1<>INDXREC THEN GET 3,R1+1:INDXREC=R1
6160 CURREC=CVI(MID$(P$,1+(2*((RELREC-1) MOD 127)),2))
6170 RETURN
```

A-2.0 File recovery techniques

There are very few times during file processing that records can be lost. Here we will discuss some recovery techniques that will save lost data under most circumstances.

A-2.1 Loss of the TIMDIR.DAT file

The loss or destruction of the file 'TIMDIR.DAT' on your data disk will result in a "File does not exist in T.I.M. directory" error even though your data file may be on the disk. This file is necessary to inform T.I.M. that the file is on the disk.

To correct this problem you must re-create the file description using the *Create a new file* command. This new file must exactly match the original file in regards to field length and type. Once this is accomplished you have a file description saying that there are 0 records in the new file although there may be thousands. To set the number of records you ***must*** now go to section A-2.2.

Once the number of records has been set you may look at your file using the *ADD/inspect/update* command. If the first record is correct but the second record appears to be jumbled, then you have not defined the field lengths to be exactly the same as the original file.

A-2.2 Loss of records

If there are more records in your file than the program indicates, you may try to recover the lost records using the techniques discussed in this section. There is a program called TIMRCNT.EXE located on your Example Files Disk, which lets the user tell T.I.M. how many records are in a file. You may run this command by typing:

```
TIMRCNT ↵
```

You must then tell T.I.M. the filename and number of records in the file. Once the number of records is set, you may check the file using the *Add/inspect/update* command. If there are blank records at the end of the file, then the record count has been set past the end of the file and must be lowered.

A-2.3 Loss of an index file

An index file tells how the file may be accessed in order (eg. alphabetical order). These files have the same filename as the original with a different extension. The extension will be ".IX?" where the question mark is a number or a letter. These files can be re-created at any time using the *Sort a file* command.

Primer introduction

The Primer contains four example applications of T.I.M. III. They are a mailing list example, a medical patient visit/payments example, a sales commissions example and a search firm example. The first two examples are very detailed so a beginning user will have no problem learning the system. The last two examples graduate to more complex applications but are still very easy to follow.

Each example application is introduced and the hypothetical application is outlined. From there, step-by-step instructions and comments lead the user through the process of defining the application files, reports, and input routines. T.I.M. is menu-driven throughout the program, so pretend you are out to eat when operating it. Command menus will be frequently displayed so all you have to do is pick the command you wish from the menu. (Just like ordering a K.C. Strip steak off a restaurant menu.)

Throughout the primer as well as the rest of the T.I.M. manual, all messages and entry onto the CRT will be printed in blue ink so as to differentiate it from the explanation and instruction.

When T.I.M. instructs you to type something into the computer, what you should type will be in **boldface** print.
The <enter> means press ↵ at the point indicated.

Lastly, the example files you will define and create will already be on the data disk supplied with the T.I.M. system, but be under different names. The "ML-LIST" file you will create will already be created and called "ML-LIST1" and the "VISITS" and "PAYMENTS" files you will define will be under the names of "VISITS1" and "PAY1", respectively. The last two example files, "SALES" and "SEARCH1" will also be on the EXAMPLE FILES disk. These files will already have records in them and will be ready for you to experiment with.

In addition to the example files, example Report, List and Select formats will be in their appropriate Libraries for demonstration and instruction.

Example 1: Mailing list application

Background information:

Assume you are a user who wishes to manage a mailing list file of 15,000 businesses. You wish to write form letters to groups of these businesses, as well as print mailing labels. Because of the volume of mail you send out, bulk rates are desired. These bulk rates are obtained by sorting the names in order of zip codes.

Thinking out the application:

Experience has taught most of us to look before we leap. This philosophy applies to computers as well as anything else. First let's jot down all the information we want T.I.M. to keep track of. (Feel free to consult section *0.2 T.I.M.'s Data base management system* if you are unfamiliar with the terms files, fields or records; or the different field types.) The information categories (fields) we want to manage might be as follows:

CONTACT NAME
COMPANY NAME
STREET ADDRESS
CITY, STATE
ZIPCODE
SIC CODE
AUX. CODE
DATE ENTERED

Now let's look at these 8 fields (categories). We need to know how long to make each field and the field type of each field. We decide that all the fields except "DATE ENTERED", "ZIPCODE" and "CONTACT NAME" will be Alphanumeric (A) fields. The "CONTACT NAME" field will be an Inverted name (I) field, the "ZIPCODE" will be a Numeric (N) field, and the "DATE ENTERED" will be a Date (D) field. These eight fields, their lengths and types are shown on the next page.

Field Name	L	T	
CONTACT NAME	25	I	
COMPANY NAME	45	A	
STREET ADDRESS	35	A	
CITY, STATE	30	A	
ZIPCODE	9	N	0
SIC CODE	10	A	
AUX. CODE	10	A	
DATE ENTERED	8	D	A

Getting the application "ON-LINE"

Enough said. Let's get this file defined and working. Turn on your computer, put the Program Disk in drive A.

```
A> TIM ⏎
```

The T.I.M. banner will be displayed and you will be asked to enter the date. Enter the date with preceding zeros on one-digit days or months. The slashes will be put in automatically.

Now you need to enter the disk drive where your data files will reside. Let's say they are on drive B, so type:

```
Enter the default drive (a—p,v=variable) B
```

The main T.I.M. menu will now come up on your screen. Look at those commands! Don't feel lost, remember what we are going to do. We are going to define the parameters and create a new file. Notice one command is called "Create a new file". That's the one we want. Type:

```
Enter command C ↵
```

This will load in the *Create* command. The *Create* sub-command minu will be displayed as follows:

```
C = Create a file that matches another file
D = Define a new file format
R = Redefine the screen format of a file
X = eXit to main menu
```

We now have a few choices. We want to create a new file from scratch so we will use the *Define a new file format*. sub-command. The computer will prompt and you will type as follows:

```
Enter the create sub-command D
```

Defining the field information

You will then see the help screen for the *"Create"* command. This screen gives you all the information you need to define a new file. To redisplay this help menu, all you have to do is hit an **H**. Notice at the bottom of the screen, the line

```
1 <SL=2400> Title:
```

is written. This is the line where the field titles, lengths and types will be entered to define our new file. The number "1" on the left designates the field number we are now defining. (Remember there can be up to 40 fields in a file.) The SL=2400 between the angle brackets means that the space left in our record is 2400 characters. This number will decrease proportionally as each new field is defined. The word "TITLE" designates the first entry: the field title. Our first field title is "CONTACT NAME".

```
1 <SL=2400> Title: CONTACT NAME ↵
```

for this entry. See what happened? The cursor moved to the right and a new prompt, LENGTH, is displayed. In our mailing list file, the length for our first field, "CONTACT NAME" was 25, so enter this number. Type:

```
Length: 25 ↵
```

Our cursor is now to the right of the prompt called "TYPE". Here is where you enter the field type abbreviation. Our field type is an Inverted name so we type:

```
Type: I ↵
```

Now the cursor goes back to the left and the same process is repeated for each field defined. Notice that the space left (SL) after our first defined field is now 2375. This number comes from the original amount (2400) less the last defined field length (25). Notice also that the number "2" is written before the " <SL=2375> " designating that we are now defining field 2. Let's define the next three fields. Type the information as follows:

```
2 <SL=2375> Title: COMPANY NAME ↵
            Length: 45 ↵
            Type: A ↵
3 <SL=2330> Title: STREET ADDRESS ↵
            Length: 35 ↵
            Type: A ↵
4 <SL=2295> Title: CITY, STATE ↵
            Length: 30 ↵
            Type: A ↵
5 <SL=2265> Title: ZIPCODE ↵
            Length: 9 ↵
            Type: N ↵
```

There goes the curve ball. Because this is a numeric field, an additional prompt, "PRECISION" appears. Here you need to enter the decimal precision of the numeric field defined. Because zip codes are whole numbers, the decimal precision is zero decimal places, so type:

```
Precision: 0 ↵
```

Be sure to enter the number zero, not the letter "O". Continue defining the next two fields. Type:

```
6 <SL=2256> Title: SIC CODE ↵
            Length: 10 ↵
            Type: A ↵
7 <SL=2246> Title: AUX CODE ↵
            Length: 10 ↵
            Type: A ↵
```

There is another curve ball around the corner. Define the "DATE ENTERED" field by typing:

8 <SL=2236> Title: **DATE ENTERED** ↵
Length: **8** ↵
Type: **D** ↵

Notice that an additional prompt "DATE TYPE" is displayed on the screen. Here you must enter which type of date you wish to use, American or European. An **A** is entered for American and an **E** for European. Type:

Date type: **A** ↵

You have now defined all eight fields for your new file. To continue on, type:

9 <SL=2228> Title: ↵

This ends the field definition process.

Defining file information:

T.I.M. will now ask:

```
Enter the new filename you wish to create (x=exit)
```

Let's call this mailing list file "ML-LIST". Enter this name by typing:

```
Enter the new filename you wish to create (x=exit) ML-LIST ⏎
```

The following prompt will be displayed on the screen:

```
Is this a confidential file (y or n, cr=n) ⏎
```

We won't use a password to access this file so we pressed ⏎.

T.I.M. now will ask you to enter a comment. This comment can be up to 60 characters long and may be used to explain more about the file. So, for our comment, type:

```
Enter comment (cr=none) TEST MAILING LIST FILE ⏎
```

Defining key-fields:

At this time, T.I.M. will write the newly defined file information onto the disk, and then ask you if you want to define a key-field. A key-field is a field by which the file may be ordered. In other words, if you wish to have this mailing list in order of zip code, the "ZIPCODE" field is a key-field. Type:

```
Do you wish to define a key-field (y or n) y
```

Now all the fields of your file are displayed on the terminal. The "ZIPCODE" field is field number 5 so type:

```
Enter the major key-field number (cr=end) 5 ⏎
Enter the minor key-field number 1 (cr=end) ⏎
```

For reporting and list functions, it would be nice to have our "ML-LIST" file also sorted by the field "CONTACT NAME". "CONTACT NAME" is field #1, so type:

```
Do you wish to define another key-field (y or n,cr=n) y ↵

Enter the major key-field number (cr=end) 1 ↵
Enter the minor key-field number 1 (cr=end) ↵
```

T.I.M. will then write this information onto the disk and prompt:

```
Define custom screen or use standard screen (c or s, cr=s,x=exit)
```

We won't worry about custom screens on this file so type:

```
Define custom screen or use standard screen (c or s, cr=s,x=exit) s
```

We did it! Now your file is defined and created. Let's proceed and add some people to the file. Exit the **Create a new file** command by typing:

```
Enter the create sub-command X
```

Adding records to our file

Now that we have our mailing list file defined and entered, we need to put information in it. This information is called data. You should now have the main T.I.M. menu in front of you on the screen. Look over the commands. We want to add records to our file so we will use the command *Add/Inspect/Update a record.* To use this command, type:

```
Enter command A
```

T.I.M. will display the message:

```
One moment while program is loading
```

Then T.I.M. will display the prompt:

```
Enter the filename (x=exit)
```

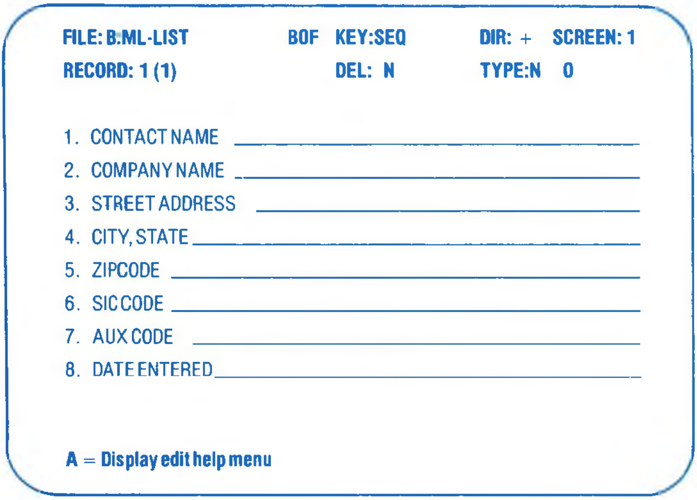
The filename of our mailing list file is "ML-LIST" so type:

```
Enter the filename (x=exit) ML-LIST ↵
```

You will now get the menu of the *Add/Inspect/Update* command on your terminal. We want to add records so you should type:

```
Enter sub-command (cr=step,h=help) A
```

We are ready to go. Your screen should look like the one on the next page.



```
FILE: B:ML-LIST           BOF   KEY:SEQ       DIR: +    SCREEN: 1
RECORD: 1 (1)                   DEL:  N       TYPE:N     0

1.  CONTACT NAME  ______________________________________
2.  COMPANY NAME  ______________________________________
3.  STREET ADDRESS  ____________________________________
4.  CITY, STATE ________________________________________
5.  ZIPCODE  ___________________________________________
6.  SIC CODE  __________________________________________
7.  AUX CODE  __________________________________________
8.  DATE ENTERED _______________________________________


A = Display edit help menu
```

Notice the status lines at the top of the screen. They look like this:

```
FILE: B:ML-LIST           BOF   KEY:SEQ       DIR: +    Screen: 1
RECORD: 1 (1)                   DEL:  N       TYPE: I
```

These status lines will tell you all sorts of good information. The first piece of information states which file is being added to. The "**RECORD: 1**" tells you that you are looking at the first record in your file. (After you add 1 record to your file this number will go to 2). The "**DEL: N**" tells you that the current record being displayed is non-deleted (will be explained later). And the "**TYPE: I**" tells you that the field type of the field that the cursor is on is an Inverted Name field. The rest of the status line will be explained later.

Let's start by typing the first record. Type the following so we can add our first record.

1. CONTACT NAME **William Fold** ⏎

Notice that the cursor is on line 2, the "COMPANY NAME" line. Notice also that the field type in the status line has changed to "A" designating an alphanumeric field. Type the following entries and watch the status line change on each new field type.

```
2.  COMPANY NAME R.D. Leather Co. ⏎
3.  STREET ADDRESS 123 Main Street ⏎
4.  CITY, STATE Hollywood, CA ⏎
5.  ZIPCODE 91212 ⏎
6.  SIC CODE 676 ⏎
7.  AUX CODE S ⏎
8.  DATE ENTERED
```

Now type today's date and press ⏎ into the date entered field. Wasn't that easy. Now it is your turn. Let loose on adding a few records. If you have any questions press F9 to get the Help Menu displayed on the screen. If you wish to change any information that you have typed, you may use the direction keys to move the cursor to the right position. When you are done adding records and the next blank record is being displayed, press:

F10 [F7]

This will display the prompt:

```
Enter sub-command (cr=step, h=help)
```

To get the *Add/Inspect/Update* menu back, type:

```
Enter sub-command (cr=step, h=help) h
```

Now you can exit this command by typing:

```
Enter sub-command X
```

T.I.M. will then ask you if you wish to merge the new records. The merge will put the new records into the sorted orders of the file. In our example "ML-LIST" file, fields 1 (CONTACT NAME) and 5 (ZIPCODE) are ordered. So if you type:

```
Do you wish to merge the records now (y or n) y ↵
```

the newly added information will be ordered. This will allow reports, mailing lists, etc. to be generated in the order of one of these fields.

Example 1: Define and generate a report

Getting to the Report command

You now have the main T.I.M. menu on your screen. We want to generate a report of the people in our "ML-LIST" file. By doing this, we can proof all the names to make sure there were no mistakes. Once proofed and OK, we will generate a mailing label list.

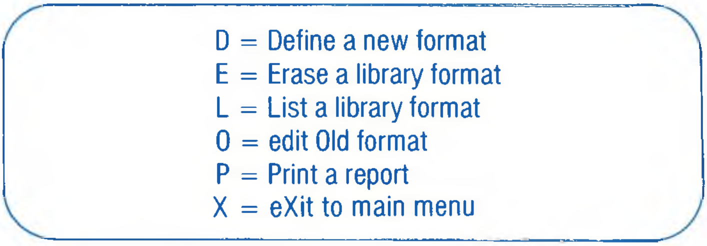
First things first. We need to generate a report so we will use the *Report generation* command on the main T.I.M. menu. To do this, type:

```
Enter command R
```

T.I.M. will then access the disk and display the message:

```
One moment while program is loading
```

Next T.I.M. will display the *Report generation* command menu. It will look like this:



```
D = Define a new format
E = Erase a library format
L = List a library format
O = edit Old format
P = Print a report
X = eXit to main menu
```

What is the Report Library?

Read over these report sub-commands. Notice that a "library" is mentioned. T.I.M. uses "libraries" throughout the program to make life easier on the user (you). This is how the libraries work. Assume you wish to print a report. First you would define what fields you want printed, which fields you might want totaled, how to title it, which files to print from and so on. Then, you would save this information (called a format) in the Report Library. You would give the whole format a name so you would remember it from others you may have stored. Once stored in the Report Library, you can print it many times without re-defining it. Also, you may edit the library format and change some of its information if needed.

Let's define the ML-LIST report

To define a report format, type:

```
Enter report sub-command D
```

when asked for the report sub-command. T.I.M. will now say:

```
Enter title #1(cr=end)
```

Here we will enter the titles of our ML-LIST report. Type:

```
Enter title #1 (cr=end) ML-LIST INFORMATION REPORT ↵
Enter title #2 (cr=end) / ↵
Enter title #3 (cr=end) ↵
```

Typing the (/), will cause the current date to be inserted into the title line when printed.

Now T.I.M. will ask for the report page width as follows:

```
Enter the page width (cr=131)
```

Enter this width by typing:

```
Enter the page width (cr=131) ↵
```

Next, T.I.M. will prompt:

```
Enter the page length (cr=66)
```

In response, type:

```
Enter the page length (cr=66) ↵
```

This will give a standard 66 line (11-inch) page length. Now T.I.M. will display the prompt:

```
Enter the main filename (x=exit)
```

The main filename in our example is "ML-LIST" because it is the name of the only file we are going to print in our report. Type:

Enter the main filename (x=exit) **ML-LIST** ⏎

T.I.M. will display all of the field information (Field titles, lengths and types) on your screen and then display the prompt:

Enter field number 1 to print (#=Rec#,cr=end)

Now is the time when you need to tell T.I.M. which fields of your file you wish to print on this report. Let's say we want to print just the fields 1,2,3,4,5,6, and 8 (CONTACT NAME, COMPANY NAME, ADDRESS, CITY, STATE, ZIP-CODE, SIC CODE, DATE ENTERED). Everything OK? Well not quite. You may remember we defined a printer width of 131 characters, and the combined width of all five of these fields is 162 characters. Looks like we have to make some adjustments. T.I.M. allows the user to truncate (cut off) a field if he wishes, during report printing. Let's use this feature. Type:

Enter field number 1 to print (#=Rec#,cr=end) **1** ⏎
Truncation length (cr=25) **20** ⏎

Upon printing, this will truncate field 1 after 20 characters. Then type:

Enter field number 2 to print (#=Rec#,cr=end) **2** ⏎

This will truncate field 2 after 30 characters. Continue defining the fields to print without using truncations by typing:

Enter field number 3 to print (#=Rec#,cr=end) **3** ⏎
Truncation length (cr=35) **30** ⏎

Enter field number 4 to print (#=Rec#,cr=end) **4** ⏎
Truncation length (cr=30) **20** ⏎

```
Enter field number 5 to print (#=Rec#,cr=end) 5 ⏎
Truncation length (cr=9) 7 ⏎

Enter field number 6 to print (#=Rec#,cr=end) 6 ⏎
Truncation length (cr=10) 4 ⏎

Enter field number 7 to print (#=Rec#,cr=end) 8 ⏎
Truncation length (cr=8) ⏎
```

We are now done defining fields so type:

```
Enter field number 8 to print (#=Rec#,cr=end) ⏎
```

T.I.M. will re-display the field information of our "ML-LIST" file and display the following prompt:

```
Enter field number 1 to total (cr=end)
```

We are not going to define any total fields in this report so type:

```
Enter the major break-field number (cr=none) ⏎
```

Similarly, we are not going to use the major break-field option, so type:

```
Enter field number 1 to total (cr=end) ⏎
```

T.I.M. will then display the file information again and follow it with the prompt:

```
Enter the key-field number to print by (cr=sequential)
```

The key-field entry is a very important entry. As mentioned earlier, a key-field is a field in a T.I.M. file that may be accessed in sorted order. In our example, we defined two key-fields when we created the "ML-LIST" file. These two fields were "CONTACT NAME" and "ZIPCODE". Looking at the field information on your screen, notice that each of these two fields have a lower case letter K (k) to the left of their field number. This k is used throughout the program to designate a key-field in these field lists.

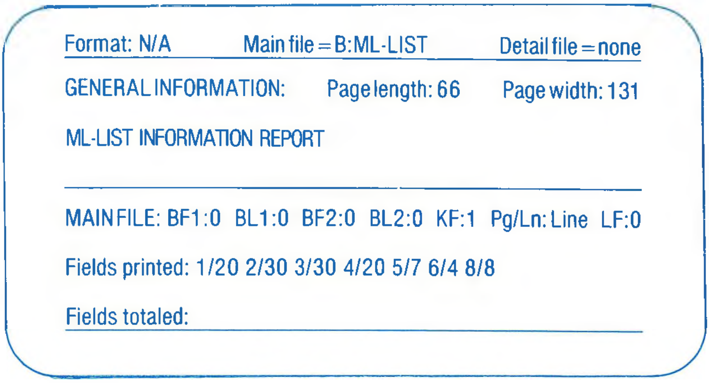
Let's print this report by order of "CONTACT NAME". The "CONTACT NAME" field number is 1 so type:

```
Enter the key-field number to print by (cr=seq) 1 ⏎
```

Next, T.I.M. will prompt you by asking you to enter the link-field. We won't use this option so type:

```
Enter the link-field if a second file is used (cr=none) ⏎
```

T.I.M. will now clear the screen and display your completed report format on the terminal. It will look like the following:



```
Format: N/A            Main file=B:ML-LIST          Detail file=none
GENERAL INFORMATION:        Page length: 66      Page width: 131
ML-LIST INFORMATION REPORT

MAIN FILE: BF1:0  BL1:0  BF2:0  BL2:0  KF:1  Pg/Ln: Line  LF:0
Fields printed: 1/20 2/30 3/30 4/20 5/7 6/4 8/8
Fields totaled:
```

Everything you have defined is displayed in this format. At the bottom of the format the following prompt will be displayed.

```
Save format, edit, or abandon (s,e, or a,cr=s)
```

At this time you may save this format in the Report Library or edit it if you made a mistake. Let's assume you are perfect, and we'll save it. Type:

```
Save format, edit, or abandon (s,e, or a,cr=s) s ⏎
```

Let's enter the name we want to call our new report format. Type:

Enter the library format name (20 chrs.,x=exit) **ML-LIST 1st REPORT** ↵

T.I.M. will now save this format on disk and return to the report sub-command menu. If there were mistakes in it, you could correct them by using the *edit Old format* sub-command.

Printing our first report:

Well, let's print this report. To print, use the *Print a report* sub-command by typing:

Enter report sub-command **P**

The contents of the Report Library will be displayed on the screen and you will be asked to enter the format number to print from. Enter the number of your defined report format and then press the ↵ key. For example, if your format (called ML-LIST 1st REPORT) is report format number 2 in the Report Library, you would type:

Enter format number to print from (x=exit) **2** ↵

T.I.M. will then display the message:

Send output to printer, terminal or disk (p,t, or d,cr=p,x=exit)

Type:

Send output to printer, terminal or disk (p,t, or d,cr=p,x=exit) ↵

After you type ↵, T.I.M. will query two more times as follows:

```
Print totals only (y or n, cr=n)
```

Here, you type:

```
Print totals only (y or n, cr=n) ↵
```

Next, T.I.M. will display the prompt:

```
Pause between pages (y or n, cr=n)
```

As before, you type:

```
Pause between pages (y or n, cr=n) ↵
```

and your report will be generated on your printer. This report should look like the one on the next page.

Success is sweet!

```
                                                       ML-LIST1 INFORMATION REPORT
                                                                02/05/82


    CONTACT NAME                COMPANY NAME                   STREET ADDRESS                CITY, STATE         ZIPCODE SIC  DATE ENTERED
------------------------------------------------------------------------------------------------------------------------------------------
Harvey Adams         Adams Weldings                 5647 Jarboe                    Tucson, AZ              63321 239        07/21/81
James Bond           Private Investigation Corp.    007 Gold St.                   New York, NY            13223 125        07/21/81
Daniel Boone         Racoon Furs Inc.               1245 Skunk Lane                Coonsville, KY          46333 233        07/21/81
Jaque Custodian      Janitorial Services            699 Drumm St.                  Littleton, CO           85921 188        07/21/81
Eileen Dover         Dover Insurance Corp.          355 E. 54 St.                  Dallas, TX              75229 556        07/21/81
Ben Gay              Gymnastic Equipment            2601 Rub Ave.                  Columbia, MO            64113 223        07/21/81
Daniel Harris        Harris & Sons Insurance Co.    125 E. 113th Terr.             Kansas City, MO         64114 663        07/21/81
Hank Hyatt           Hyatt Regency Hotel Corp.      1356 Central Express Way       Dallas, TX              75229 329        01/21/81
Henry Jackson        Jackson Insurance Agency       2525 Madison                   Tempe, AZ               84521 239        09/09/81
Elton John           Elton Sound & Music Corp.      Treble Clef Hills              Beverly Hills, CA       95553 342        09/09/81
James Nasium         Nasium Construction Co.        231 Olympic Blvd.              Shawnee Mission, KS     63123 233        07/21/81
Howard Nelson        Union 76 Oils                  313 Westover                   Kansas City, KS         64131 123        07/21/81
Ferita Photographer  Photography & Prints           135 Olympus Drive              Dallas, TX              75229 333        07/21/81
Fred G. Sanford      Sanford & Sons Antiques        135 Junk Road                  Beverly Hills, CA       94263 231        07/21/81
Thomas J. Spiff      Spiff Paints Inc.              39 W. 102 Terr.                Shawnee Mission, KS     63123 123        07/21/81
Sam Squidd           Squidds & Sons Insurance Inc.  1225 Sea Lane                  Springfield, MO         64113 559        07/21/81
Dan Tanna            Tanna Investigation Inc.       125 Flamingo Drive             Las Vegas, NV           98225 266        07/21/81
```

Ex.—20

Example 1: Defining and generating a list

Getting to the List command:

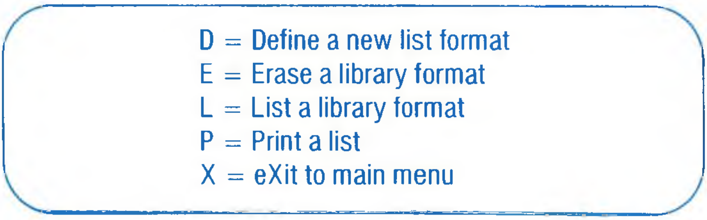
You now have the main T.I.M. menu on your screen. We want to use the names and addresses in our "ML-LIST" file for printing of mailing labels. To accomplish this you need to use the *List generation* command. Type:

```
Enter command L
```

T.I.M. will then access the disk and display the message:

```
One moment while program is loading
```

Now we are where we want to be. The *List Generation* command menu is on your screen and looks like this:



```
D = Define a new list format
E = Erase a library format
L = List a library format
P = Print a list
X = eXit to main menu
```

The List Library:

The List Library is functionally the same as the Report Library. It stores list formats for later printing.

Let's define a mailing list format

To define a list format, type:

```
Enter list sub-command D
```

when asked for the list sub-command. T.I.M. will now display the message:

```
Enter the filename to generate lists from (x=exit)
```

We are going to generate lists from our "ML-LIST" file so type:

```
Enter the filename to generate lists from (x=exit) ML-LIST ↵
```

T.I.M.'s list generator was designed for ease of use as well as versatility. To accomplish these two tasks, the idea of a "list" was used. While defining a list format you will define overall list specifications and specific list specifications. To define a list format, first you define the overall information such as number of labels across the page, number of blank lines between each list and the starting tabs of each list. Next you define specifically which fields are printed where in each list. These same fields will be printed on each list across the page (assuming you use more than one-up labels).

Now T.I.M. will prompt:

```
Enter number of lists across page (1-4)
```

At this time we need to enter a number from 1 to 4. This is the number of lists (or labels) abreast on the page. Type:

```
Enter number of lists across page (1-4) 3 ↵
```

Now T.I.M. will ask for the starting tab of the first list. We want the tab to be 1 so type:

```
Enter starting tab of first list (cr=1) 1 ⏎
```

Type in the next two list tabs as follows:

```
Enter starting tab of second list 37 ⏎
Enter starting tab of third list 75 ⏎
```

After the list tabs have been entered, you will type:

```
Enter number of lines between lists 3 ⏎
```

Now T.I.M. will display the field information on our "ML-LIST" file, and ask:

```
Enter the key-field number to print from (cr=seq)
```

We want the list printed in order of the "CONTACT NAME" field so type:

```
Enter the key-field number to print from (cr=seq) 1 ⏎
```

T.I.M. will now display the *List generation* Help menu so you may define which fields you wish to print, and where you want to print them on each list. First, the prompt:

```
Line #:
```

will come up on the screen. Here you enter the line number where you want the first field printed. Type the following:

```
Line #: 1 ⏎
```

Next, the prompt:

```
Column #:
```

will be displayed on the same line. Type:

```
Column #: 1 ⏎
```

Next, the prompt:

```
Field #:
```

will be displayed. Type:

```
Field #: 1
```

Then the prompt:

```
Field width:
```

will be displayed. Type:

```
Field width: 25 ↵
```

Lastly, the prompt:

```
Text:
```

will be written on the screen. We don't want any extraneous text on our labels so type:

```
Text: ↵
```

This will cause the whole field definition process to begin again. On our last field definition we just specified that on line 1 of each of our lists across the page, field 1 (CONTACT NAME) will be printed starting at column 1 and be truncated after 25 characters. Let's continue to define our list format as follows by typing:

```
Line #: 2  Column #:  1  Field #: 2  Field width: 39  Text: ↵

Line #: 3  Column #:  1  Field #: 3  Field width: 35  Text: ↵

Line #: 4  Column #:  1  Field #: 4  Field width: 28  Text: ↵

Line #: 4  Column #:30  Field #: 5  Field width:  9  Text: ↵
```

We are now done defining specific field information so type:

> Line #: ⏎

for the new line # entry. T.I.M. will now display the prompt:

> Enter the List Library format name (20 char., x=exit)

In response to this query type:

> **ML-LIST 1st TEST** ⏎

T.I.M. will now bring you back to the *List Generation* menu.

Printing a mailing list:

O.K., let's print our list. To print a list, type:

> Enter sub-command **P**

when the *List generation* sub-command is on the screen. T.I.M. will display the contents of the List Library and your newly defined "ML-LIST 1st TEST" should be displayed. T.I.M. will prompt:

> Enter the format number to print from (x=exit)

In response, type the number corresponding to your new format and then press ⏎. For instance, if your list format was number 3, you would type:

> Enter the format number to print from (x=exit) **3** ⏎

T.I.M. will then retrieve the format from the List Library and prompt:

> Use defined file named <B:ML-LIST> (y or n, cr=y)

and you will type:

```
Use defined file named <B:ML-LIST> (y or n, cr=y) y
```

T.I.M. will access the disks for all needed information and then you will type:

```
Send to printer or terminal (p or t, cr=p) p
```

We don't want to pause between lists so type:

```
Pause between lists (y or n,cr=n) ↵
```

in response to the next query. The last option will allow you to begin printing the file after the first record. This option may be used to finish an interrupted mailing list generation. For this query, you will type:

```
Start at which record number (cr=1, x=exit) ↵
```

The last prompt before printing allows a very useful feature. The prompt is shown below:

```
Paper ready (t=test, cr=print)
```

By entering a "T", T.I.M. will print a string of characters so that you may line up your paper or mailing labels. Let's use this feature just to see it work. Type:

```
Paper ready (t=test, cr=print) t
```

Now the prompt will be displayed again for further paper alignment or for actual list generation. Let's generate our lists. Type:

```
Paper ready (t=test, cr=print) ↵
```

Your list should look like the one on the next page. You may exit this command by typing an "X" during printing.

Harvey Adams
Adams Weldings
5647 Jarboe
Tucson, AZ 63321

James Bond
Private Investigation Corp.
007 Gold St.
New York, NY 13223

Daniel Boone
Racoon Furs Inc.
1245 Skunk Lane
Coonsville, KY 46333

Jaque Custodian
Janitorial Services
699 Drumm St.
Littleton, CO 85921

Eileen Dover
Dover Insurance Corp.
355 E. 54 St.
Dallas, TX 75229

Ben Gay
Gymnastic Equipment
2601 Rub Ave.
Columbia, MO 64113

Daniel Harris
Harris & Sons Insurance Co.
125 E. 113th Terr.
Kansas City, MO 64114

Hank Hyatt
Hyatt Regency Hotel Corp.
1356 Central Express Way
Dallas, TX 75229

Henry Jackson
Jackson Insurance Agency
2525 Madison
Tempe, AZ 84521

Elton John
Elton Sound & Music Corp.
Treble Clef Hills
Beverly Hills, CA 95553

James Nasium
Nasium Construction Co.
231 Olympic Blvd.
Shawnee Mission, KS 63123

Howard Nelson
Union 76 Oils
313 Westover
Kansas City, KS 64131

Ferita Photographer
Photography & Prints
135 Olympus Drive
Dallas, TX 75229

Fred G. Sanford
Sanford & Sons Antiques
135 Junk Road
Beverly Hills, CA 94263

Thomas J. Spiff
Spiff Paints Inc.
39 W. 102 Terr.
Shawnee Mission, KS 63123

Sam Squidd
Squidds & Sons Insurance Inc.
1225 Sea Lane
Springfield, MO 64113

Dan Tanna
Tanna Investigation Inc.
5 Flamingo Drive
Las Vegas, NV 98225

Ex.—27

Example 1: Selecting records from the "ML-LIST" file

Getting to the "Select records" command:

Getting to the *Select records* command is just like any other. When the main T.I.M. menu is displayed on the terminal, type:

```
Enter command S
```

The *Select records* sub-command menu will be displayed as follows:

```
D = Define a new library format
E = Erase a library format
L = List a library format
R = Replace fields in selected records
S = Seleact records from a file
X = eXit to main menu
```

What does the Select command do?

The *Select records from a file* command is one of the most powerful T.I.M. commands. This command will allow the user to search an entire file for certain field information and draw out only those records which satisfy a selection criteria. For instance, you could select only those records from your "ML-LIST" file whose contact name fell between Abe Bell and Carol Mason. In this example, the records containing John Brown and Bill Henderson would be selected but the record containing Nancy Pope would not. Similarly, multiple search criteria may be defined. An example of this would be the selection of all zip codes between 64001 and 64200 but not the zip code 64114.

The *Select records form a file* command allows the same library library commands as the Report and List generation commands. You may store information permanently in the Select Library and use it over and over without re-definition.

The *Select records from a file* command allows three different output routines for the records which conform to the search criteria. The records may be sent one by one to the terminal, to the printer in a report or list, or to the disk and made into a new file.

Selecting between values

To start with, let's begin by defining a library format (select criteria). When the *Select records* sub-command menu is displayed, type:

```
Enter select sub-command D
```

for *Define a new library format.* T.I.M. will then ask:

```
Enter the origin data filename (x=exit) ML-LIST
```

The origin data filename is the filename of the file we are going to select records from. In this example we entered "ML-LIST" file, and display the prompt below:

```
Enter search criteria (x=exit, h=help, ?=fields)
```

Entering an X will end the process, entering an "H" will display a help menu and entering a question mark (?) will redisplay the field information. Let's look at our Help menu. Type:

```
Enter search criteria (x=exit, h=help, ?=fields) h ↵
```

We will get a whole screen of useful information. This help menu displays all of the field relations, logical and relational operators as well as giving some examples. You may refer to this Help menu as well as the field list during normal select definition.

In our case, let's select all the records whose contact name falls between two names: Abe Bell and Carol Mason. To do this, we can use the "Between Values" operator. The "CONTACT NAME" field is field #1, so what we want is all records with the contents of field #1 between Abe Bell and Carol Mason. To enter this as the search criteria, type:

```
FLD 1 BV Abe Bell:Carol Mason ↵
```

T.I.M. will then display the prompt:

```
Enter the key-field to select by (cr=seq)
```

Here, just as in the Report or List command, you need to enter the order to go through the file. We will go through and search the file by order of "CONTACT NAME" so we type:

```
Enter the key-field to select by (cr=seq) 1 ↵
```

Now we will enter the name we wish to call our new format. Type:

```
Enter the format name (20 chrs.,x=exit) ML-LIST Bell:Mason
```

T.I.M. will then bring you back to the *Select* sub-command menu. Let's go through the ML-LIST file, find all records which fit this criteria, and send them to the printer. Type:

```
Enter select sub-command S
```

And you will get the current contents of the Select Library. Pick your newly defined format, the "ML-LIST Bell:Mason" format and type its library format number and press ↵. For example, if it is library format #1, type:

```
Enter format number to select by (x=exit) 1 ↵
```

T.I.M. will then display the message:

```
Use defined file named <B:ML-LIST> (y or n, cr=y)
```

We will use the defined file so type:

```
Use defined file named <B:ML-LIST> (y or n, cr=y) ↵
```

T.I.M. will access the disk and display the prompt:

```
Send output to terminal, disk or printer (t,d,p,cr=t)
```

We want to send our output to the printer so type:

```
Send output to terminal, disk or printer (t,d,p,cr=t) p
```

We now have two choices. We can list the selected records in a tabular report format or in a vertical list format. Let's use the report format. Type:

```
Report or list format (r or l,cr=l) r
```

The report generator in the *Select* command is a simple one which allows two report titles and no summaries. For the first title, type:

```
Enter report title number 1 ML-LIST FILE ↵
```

and for the second title, type:

```
Enter report title number 2 All between Abe Bell and Carol Mason ↵
```

T.I.M. will now display a field list. You can pick which fields you wish to have printed in your *Select* report. Type:

```
Enter field number 1 to print (cr=end) 1 ↵
Enter field number 2 to print (cr=end) 2 ↵
Enter field number 3 to print (cr=end) 3 ↵
Enter field number 4 to print (cr=end) ↵
```

We can't print any more fields because they won't fit on a 131 page. T.I.M. will now generate this report.

Multiple selection

Now let's define another library format and send the selected records to disk: When the *Select* sub-command is being displayed, type:

```
Enter select sub-command D
```

Now type the origin filename again.

```
Enter the origin data filename (x=exit) ML-LIST ⏎
```

to get the Help menu displayed. In this example, let's find all zip codes between 64001 and 64200 but not the zip code 64114. The search criteria would be typed in as follows:

```
FLD 5 BV 64001:64200, AND FLD 5 NE 64114
```

Notice that the "ZIPCODE" field is field #5. Type:

```
Enter key-field number to select by (cr=seq) 5 ⏎
```

for the key-field number to select records by. Now enter the format name. Type:

```
Enter the format name (20 chrs.,x=exit) ML-LIST BV AND NE ⏎
```

T.I.M. will go back to the "Select" sub-command menu. To start our selection type:

```
Enter select sub-command S
```

Type in the appropriate Select Library format number and a return. Next type:

```
Used defined file <ML-LIST> (y or n, cr=y) y ⏎
```

so that T.I.M. will use the defined file "ML-LIST". Here we go a different route than our last select. We want to send the records which conform to the search criteria to a new file on disk. When the prompt:

```
Send output to terminal, disk or printer (t,d,p, x=exit)
```

is displayed, type:

```
Send output to terminal, disk or printer (t,d,p, x=exit) d
```

T.I.M. will display the prompt:

```
Enter the destination filename (cr=end)
```

Here we need to enter a name for the new file we are going to put the selected records in. This new file will then be a subset of the original file. Type:

```
Enter the destination filename (cr=end) NEW-LIST ↵
```

then type:

```
Enter comment (cr=none) ML-LIST SUBSET ↵
Enter password (cr=none) ↵
```

for the comment and security code. T.I.M. will then display the message:

```
Record Selection in progress
```

T.I.M. will select all the records which conform to your designated search criteria and copy them to a new file called "NEW-LIST". This file may then be treated as any other T.I.M. file and inspected, searched, or have lists or reports printed from it.

Example 2: Patient records and payment files

Background information:

Assume you are a doctor who wishes to manage the patient files of each visit, fee and payment. Before your computer was installed you were writing everything down by hand and your secretary was transfererring it to each patients history and ledger cards. Billing at the end of every month was done by going through all the patients' files and figuring who paid, when and how much. To ease this paperwork headache will take more than aspirin and bed rest.

Thinking out the application

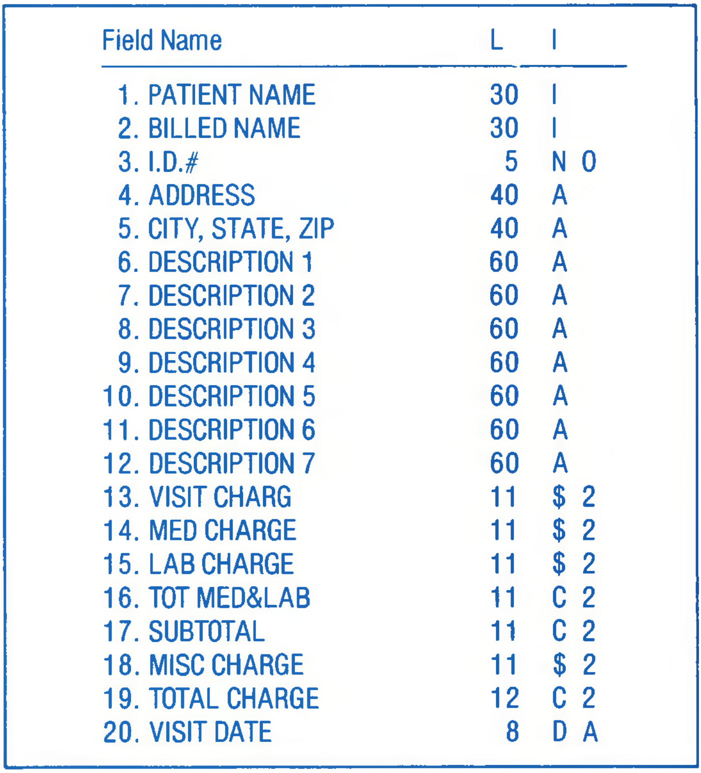
Just as in our mailing list application, let's think about what we are trying to do. It looks like we need two separate but related groups of information. One group would be the patient's visit information and the other would be the patient's payment information. Let's put each of these two groups in their own file. The patient visit file could be called "VISITS" and contain the following fields.

PATIENT NAME
BILLED NAME
BILLING ID#
ADDRESS
CITY, STATE
VISIT TEXT
VISIT CHARGES
MEDICAL CHARGES
LAB CHARGES
MISC CHARGES
TOTAL CHARGES
VISIT DATE

The information in the patient payments file called "PAYMENTS" might be as follows:

PATIENT NAME
BILLING I.D.#
DATE
PAYMENT

Now let's look at these two files. Starting with the "VISITS" file, we need to figure out the field lengths and field types for this file. They might be as follows:



Field Name	L	I	
1. PATIENT NAME	30	I	
2. BILLED NAME	30	I	
3. I.D.#	5	N	0
4. ADDRESS	40	A	
5. CITY, STATE, ZIP	40	A	
6. DESCRIPTION 1	60	A	
7. DESCRIPTION 2	60	A	
8. DESCRIPTION 3	60	A	
9. DESCRIPTION 4	60	A	
10. DESCRIPTION 5	60	A	
11. DESCRIPTION 6	60	A	
12. DESCRIPTION 7	60	A	
13. VISIT CHARG	11	$	2
14. MED CHARGE	11	$	2
15. LAB CHARGE	11	$	2
16. TOT MED&LAB	11	C	2
17. SUBTOTAL	11	C	2
18. MISC CHARGE	11	$	2
19. TOTAL CHARGE	12	C	2
20. VISIT DATE	8	D	A

Notice that there has been 420 characters set aside for descriptive visit text. This should allow plenty of room for good patient record keeping per visit. Also notice the charges are all Dollar Fields ($) and the totals are Calculated (C) Fields.

The "PAYMENTS" file is an easy one. The field descriptions and types might be as follows:

Field Name	L	T	
BILLED NAME	30	I	
PATIENT NAME	30	I	
I.D.#	5	N	O
DATE	8	D	A
PAYMENT	11	$	

Getting the application on line

Let's get this application up and running. Put your program disk in drive A and boot your system. Go to the *Create a new file* command and type a

Enter the Create sub-command **D**

so we can begin to define the new file format. Enter the first title by typing:

```
1 <SL=2400>   Title: PATIENT NAME ↵
              Length: 30 ↵
              Type: I ↵
```

Continue defining the next 11 fields up through "DESCRIPTION 7" just as you did in the mailing list file. After you get don e with the first one and the following 11, the T.I.M. prompt will be as follows:

```
13 <SL=1835> Title:
```

We now need to enter the visit charge field. Type:

```
13 <SL=1835> Title: VISIT CHARG ⏎
             Length: 11 ⏎
             Type: $ ⏎
```

By entering the “$” as the field type, the field will automatically have two decimal places. Enter the “MED CHARGE” and the “LAB CHARGE” fields the same way by typing:

```
14 <SL=1824> Title: MED CHARGE ⏎
             Length: 11 ⏎
             Type: $ ⏎

15 <SL=1813> Title: LAB CHARGE ⏎
             Length: 11 ⏎
             Type: $ ⏎
```

Now we come to the “TOT MED&LAB” field. Enter the field title and length by typing:

```
16 <SL=1802> Title: TOT MED&LAB ⏎
             Length: 11 ⏎
```

Now enter the field type as a Calculated Field by typing:

```
Type: C ⏎
```

You will be asked for the precision of the caluclated field. The precision is two decimal places (for dollar and cents) so you would type:

```
Precision: 2 ⏎
```

T.I.M. will now clear the screen and write a list of all the fields you have defined so far. You will then enter the first operand field. This is field number 14, the ‘MED CHARGE” field. Type:

```
Enter the field # of the first operand (x=exit) 14 ⏎
```

Now you must enter the operator sign. We are going to add the contents of field 14, 'MED CHARGE" to field 15 "LAB CHARGE" to get a total. Therefore the sign of the calculation is a plus (+) sign. Type:

```
Enter the operator (+,—,*,/,x=exit) + ⏎
```

Now you must enter the next operand field. It is field 15 so type:

```
Enter the field # of the second operand (x=exit) 15 ⏎
```

You have made it through your first calculated field. If you have two fields you want to multiply, a calculated field could be used there too. (An example of this might be price times quantity to get an extension.) You may add, subtract, multiply or divide using the four operators +,—,*, and /. Let's finish defining the fields in this file. For the "SUBTOTAL" field type:

```
17 <SL=1791> Title: SUBTOTAL ⏎
             Length: 11 ⏎
             Type: C ⏎
             Precision: 2 ⏎

Enter the field # of the first operand (x=exit) 13 ⏎
Enter the operator (+,—,*,/,x=exit) + ⏎
Enter the field # of the second operand (x=exit) 16 ⏎
```

For the "MISC CHARGE" field type:

```
18 <SL=1780> Title: MISC CHARGE ⏎
             Length: 11 ⏎
             Type: $ ⏎
```

For the "TOTAL CHARGE" field type:

```
19 <SL=1769> Title: TOTAL CHARGE ⏎
             Length: 12 ⏎
             Type: C ⏎
             Precision: 2 ⏎
```

Then you would type:

```
Enter the field # of the first operand (x=exit) 17 ↵
Enter the operator (+,—,*,x=exit) + ↵
Enter the field # of the second operand (x=exit) 18 ↵
```

Lastly, to define the "VISIT DATE" field type:

```
20 <SL=1757> Title: VISIT DATE ↵
             Length: 8 ↵
             Type: D ↵
             Date Type: A ↵
```

We are now done defining fields and you will have the prompt

```
21 <SL=1749> Title:
```

on your screen. If you wish to examine all the fields, type a question mark "?" for the field title query. Everything looks in order so let's end the field definition process by typing:

```
21 <SL=1749> Title: ↵
```

T.I.M. will then query:

```
Are you finished defining (y or n)
```

And you would type:

```
Are you finished defining (y or n) y
```

Now you can enter the filename for your new file. Let's call it "VISITS" so type:

```
Enter the new filename you wish to create VISITS ↵
```

Assume we want to make this file accessable only by people who knew the correct password. To do this, type:

```
Is this a confidential file (y or n ,cr=n) y
Enter a four letter code (cr=unprotected) ABCD ↵
```

Now "ABCD" is your password to this confidential file. Enter the comment by typing:

```
Enter comment (cr=none) PATIENT VISIT INFO FILE ↵
```

Now T.I.M. will write the new file on the disk and ask you if you want to define a key-field. Let's define two key-fields so the file will be ordered two ways, one by "PATIENT NAME" and one by "I.D.#". To do this type:

```
Do you wish to define a key-field (y or n,cr=exit) y
```

when you are asked if you want to define a key-field and then type:

```
Enter the major key-field number (cr=end) 1 ↵
Enter the minor key-field number 1 (cr=end) ↵

Do you wish to define another key-field (y or n,cr=n) y

Enter the major key-field number (cr=end) 3 ↵
Enter the minor key-field number 1 (cr=end) ↵

Do you wish to define another key-field (y or n,cr=n) n
```

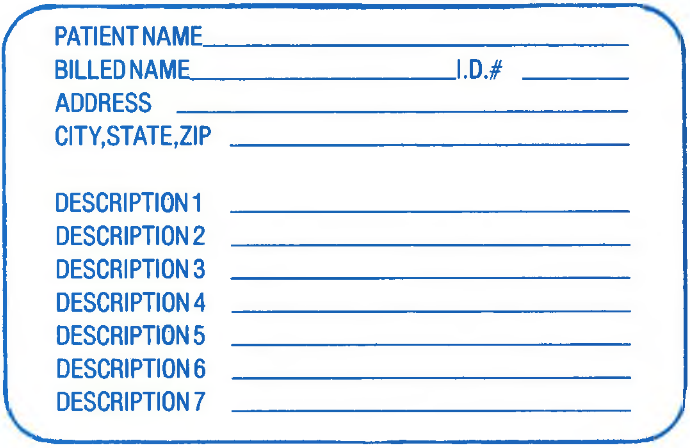
T.I.M. will then write all this information on the disk and write the following:

```
Define custom screen or use standard screen (c or s,cr=s,x=exit)
```

For our file, let's define a screen so data entry will be easier with this file. Enter a C.

Defining a custom screen:

Now we will define a custom screen format for our "VISITS" file. Our custom screen will group all related inputs so data entry will not be confusing. You define the screen format by moving the cursor to the desired spot on the CRT screen. You will have the option of defining one or two screens of field entries. Let's say we want screen 1 to contain all the visit information and look like this:



PATIENT NAME____________________________

BILLED NAME__________________ I.D.# ________

ADDRESS ______________________________

CITY,STATE,ZIP ___________________________

DESCRIPTION 1 ___________________________

DESCRIPTION 2 ___________________________

DESCRIPTION 3 ___________________________

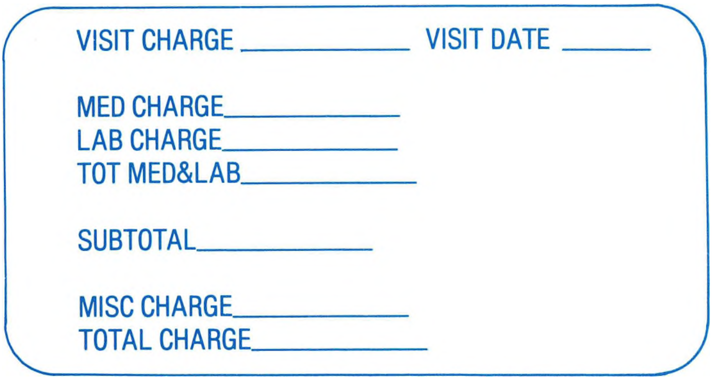
DESCRIPTION 4 ___________________________

DESCRIPTION 5 ___________________________

DESCRIPTION 6 ___________________________

DESCRIPTION 7 ___________________________

Screen 2 will contain all the charges information and look like the one following:



VISIT CHARGE ____________ VISIT DATE ______

MED CHARGE____________
LAB CHARGE____________
TOT MED&LAB____________

SUBTOTAL____________

MISC CHARGE____________
TOTAL CHARGE____________

To define these screens use the following screen definition commands.

SCREEN DEFINITION COMMANDS

← = Cursor left

→ = Cursor right

↑ = Cursor up

↓ = Cursor down

INS = Insert field at this position

F2 = Delete field

F3 = Go to first field

F4 = Go to next field

F5 = Go to previous field

F8 = Next HELP menu

F7 = Exit screen definition

B = Blank all fields
E = go to End of screen
F = go to First field
P = insert Previously defined screen
DEL = Delete current line
S = go to Start of Screen
T = Toggle to next screen
SPACE = Insert blank line
TAB = Tab 10 spaces

The bottom two lines of the screen will scroll through the help menus for screen editing. Notice also that there is a status line at the top of the screen. This line will look like the following as you begin your screen definition of the "VISITS" file.

```
Field:1   CONTACT NAME   Length:25   Line:1   Column:1   Screen:1
```

This status line explains all the field information of your current field as well as cursor position.

Now let's get started. Move the cursor (it is now in the upper left corner of the screen) down a couple lines using the ↓ key. Now press

INS

This will insert field 1, which was displayed in the status line, at the position where the cursor was. Move the cursor down one line and press

INS

This places the second field at that spot. Move the cursor over about five spaces to the right of field 2 using the → key. Press

INS

See how it placed field 3, "I.D.#" at that spot. Continue placing all the fields which are to be on screen 1. When you have finished, Press

T

and you will get a new blank screen. This is screen 2. Do as you did previously and define all the field positions. When you are satisfied, press

F7

to exit the screen definition stage.

On to *Add/inspect/update a record.*

Example 2: Define and generate a report

What to do

In the patient visits/payments example, (#2) we have two related files. One file contains the patient visit information including charges, and one file contains the payment information. We need to define a report which will link these two files and their information together. This will let us compare charges to payments so statements may be sent.

Let's define the report

From the main T.I.M. menu, type:

```
Enter command R
```

to get to the *Report generation* command. We need to define a report, so type:

```
Enter report sub-command D
```

when asked to enter the Report sub-command. T.I.M. will begin the report definition by asking:

```
Enter title #1 (cr=end)
```

Type the following for your report titles:

```
Enter title #1 (cr=end) PATIENT CHARGES AND PAYMENTS ↵
Enter title #2 (cr=end) LINKED FILE REPORT ↵
Enter title #3 (cr=end) USING VISITS AND PAYMENTS FILES ↵
Enter title #4 (cr=end) / ↵
Enter title #5 (cr=end) ↵
```

Use the standard page width and length so type:

```
Enter page width (cr=131) ↵
Enter page length (cr=66) ↵
```

for these two entries. Now T.I.M. will display the prompt:

```
Enter the main filename (x=exit)
```

The main filename in this example will be the "VISITS" file. Later the "PAYMENTS" file will be entered as the detail file. Type:

```
Enter the main filename (x=exit) VISITS ↵
```

Because we made our VISITS file confidential, T.I.M. will now ask us to enter the file's password as follows:

```
This is a confidential file. Enter the access code.
```

Our access code is "ABCD" so type:

```
This is a confidential file. Enter the access code. ABCD
```

T.I.M. will access the disk and then display all the field information on our "VISITS" file. T.I.M. will prompt:

```
Enter field number 1 to print (#=Rec#,cr=end)
```

It is now time to enter the fields to print across our report, as well as their truncation lengths (if used). Type:

```
Enter field number 1 to print (#=Rec#,cr=end) 2 ↵
Truncation length (cr=30) 15 ↵
```

This will print field #1 (PATIENT NAME) and truncate it after 15 characters. Now type the rest:

```
Enter field number 2 to print (#=Rec#,cr=end) 1 ↵
Truncation length (cr=30) 15 ↵
```

```
Enter field number 3 to print (#=Rec#,cr=end) 3 ↵
Truncation length (cr=5) ↵

Enter field number 4 to print (#=Rec#,cr=end) 20 ↵
Truncation length (cr=8) ↵

Enter field number 5 to print (#=Rec#,cr=end) 13 ↵
Truncation length (cr=11) ↵

Enter field  number 6 to print (#=Rec#,cr=end) 14 ↵
Truncation length (cr=11) ↵

Enter field number 7 to print (#=Rec#,cr=end) 15 ↵
Truncation length (cr=11) ↵

Enter field number 8 to print (#=Rec#,cr=end) 17 ↵
Truncation length (cr=11) ↵

Enter field number 9 to print (#=Rec#,cr=end) 18 ↵
Truncation length (cr=11) ↵

Enter field number 10 to print (#=Rec#,cr=end) 19 ↵

Truncation length (cr=12) ↵
```

We are now done defining fields so type:

```
Enter field number 11 to print (#=Rec#,cr=end) ↵
```

T.I.M. will now display the prompt:

```
Enter field number 1 to total (cr=end)
```

In this report, we are going to total all the charges of the main file, as well as the payments from the detail file. Now we will define which fields in our main file we want totaled. We want the "SUBTOTAL", "MISC CHARGE" and "TOTAL CHARGE" fields totaled, so type:

```
Enter field number 1 to total (cr=end) 17 ⏎
Enter field number 2 to total (cr=end) 18 ⏎
Enter field number 3 to total (cr=end) 19 ⏎
```

You are defining the fields to total, so type:

```
Enter field number 4 to total (cr=end) ⏎
```

Now T.I.M. will prompt:

```
Enter the major break-field number (cr=none)
```

A break-field is a field where T.I.M. will print sub-totals after multiple entries. In our example, if a patient visits the doctor three times in a month, he could have three different visit records each containing the visit charge. By defining a brake-field, we can total all visit records of a person. We will enter field #2, (BILLED NAME) as the break-field so we can get totals at each new billed party. Type:

```
Enter the major break-field number 2 ⏎
```

T.I.M. will display the prompt:

```
Enter minor break-field number (cr=none)
```

We won't use a minor break-field in this report, so type:

```
Enter minor break-field number (cr=none) ⏎
```

T.I.M. will now display the prompt:

```
Skip line(s) or page after each break (l or p,cr=l)
```

We will skip a line on the report after each break so type:

```
Skip line(s) or page after each break (l or p,cr=l) l
```

Because you chose lines after each break, T.I.M. will prompt:

```
Enter number of blank lines after a major break (cr=1)
```

Let's use 5 lines between each break, so type:

```
Enter number of blank lines after a major break (cr=1) 5 ⏎
```

Now you will be asked:

```
Enter the link-field if a second file will be used (cr=none)
```

In our example, the link-field in our visits file is field #3, (I.D.#). The link-field is a field common to both files printed (if two fields are printed on one report). The link-field allows T.I.M. to print the main file record (in our case a visit charge), and look up any records in the detail file which match the link-field. This way the charges of one patient may be grouped and linked to his payment records. Type:

```
Enter the link-field if a second file will be used (cr=none) 3 ⏎
```

Now T.I.M. knows that a second file will be used in this report. The query:

```
Enter the detail filename (x=exit)
```

will be displayed. Here, type:

```
Enter the detail filename (x=exit) PAYMENTS
```

A field list of this file will be displayed and then you will be asked:

```
Enter the field to link the detail file to the main file
```

The link-field must be the same type of field as the last link-field (I.D.#) so type:

```
Enter the field to link the detail file to the main file 3 ⏎
```

Now we get to the field definition stage of the second file. T.I.M. will prompt:

```
Enter field number 1 to print (#=Rec#,cr=end)
```

Here we enter the fields in the detail file we wish to print, as well as their truncation lengths. Type the following:

```
Enter field number 1 to print (cr=end) 2 ⏎
Enter truncation length (cr=30) ⏎
Enter field number 2 to print (cr=end) 4 ⏎
Enter truncation length (cr=8) ⏎
Enter field number 3 to print (cr=end) 5 ⏎
Enter truncation length (cr=11) ⏎
```

We are done so type:

```
Enter field number 4 to print (cr=end) ⏎
```

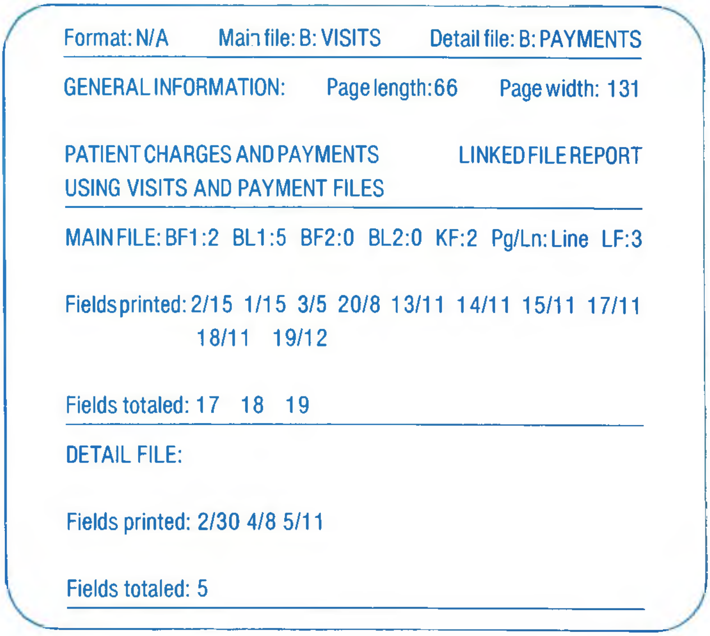
We will total the "PAYMENT" field so type:

```
Enter field number 1 to total 5 ⏎
```

for the field number to total. Then type:

```
Enter field number 2 to total ⏎
```

We are now done defining our report and T.I.M. will display the report format on the screen. It will look like the following:



```
Format: N/A        Main file: B: VISITS        Detail file: B: PAYMENTS
GENERAL INFORMATION:      Page length:66        Page width: 131

PATIENT CHARGES AND PAYMENTS               LINKED FILE REPORT
USING VISITS AND PAYMENT FILES
MAIN FILE: BF1:2  BL1:5  BF2:0  BL2:0  KF:2  Pg/Ln: Line  LF:3

Fields printed: 2/15  1/15  3/5  20/8  13/11  14/11  15/11  17/11
                18/11   19/12

Fields totaled: 17   18   19
DETAIL FILE:

Fields printed: 2/30 4/8 5/11

Fields totaled: 5
```

You will then be asked:

```
Save, edit, or abandon report format (s,e,a,cr=s)
```

You want to save it so type:

```
Save, edit, or abandon report format (s,e,a,cr=s) s
```

Now enter the library name as follows:

REP. #2 ⏎

Your new format will be saved in the report library. If you made a mistake, use the *edit Old format* sub-command.

Printing our report:

To print the report, type a **P** when asked to enter the report sub-command. After you type the P, the contents of the report library will be displayed. You must find the right format (called REP. #) and type its Report Library format number and then press ⏎. T.I.M. will then display:

```
This is a confidential file. Enter the access code.
```

Just as before, type:

```
This is a confidential file. Enter the access code. ABCD
```

Next T.I.M. will query:

```
Send to printer, terminal or disk (p,t or d,cr=p,)
```

and you will type:

```
Send to printer, terminal or disk (p,t or d,cr=p,) ⏎
```

A ⏎ will instruct T.I.M. to send the output to the printer. Next, T.I.M. will prompt with:

```
Print totals only (y or n,cr=n) ⏎
```

By hitting return you are instructing the program to print records and totals. Last T.I.M. will prompt with:

```
Pause between pages (y or n,cr=n) ⏎
```

A ⏎ here says not to pause between pages.

Your report should now be printing and look like the one on the next page.

```
                                    PATIENT CHARGES AND PAYMENTS
                                         LINKED FILE REPORT
                                    USING VISITS1 AND PAY1 FILES
                                       FROM EXAMPLE FILES DISK
                                              02/11/82

 BILLED NAME     PATIENT NAME    I.D.# VISIT DATE VISIT CHARG MED CHARGE  LAB CHARGE   SUBTOTAL   MISC CHARGE TOTAL CHARGE
                  PATIENT NAME               DATE      PAYMENT
-----------------------------------------------------------------------------------------------------------------------------
Kett Blakeley    Kett Blakeley     47    03/07/80       10.00      56.00       45.00      111.00        0.00       111.00
Kett Blakeley    Lester Blakeley   47    11/04/80       10.00     650.00       54.00      714.00        0.00       714.00
Kett Blakeley    Sara blakeley     47    08/01/80       20.00      11.00       22.00       53.00        0.00        53.00
                                                                                      ----------- ----------- ------------
          <TOTAL>                                                                          878.00       $0.00       878.00

MJ Record count = 3

              Kett Blakeley               03/07/81       50.00


Perry Carson     Perry Carson      652   06/07/81       10.00      46.00       65.00      121.00        0.00       121.00
Perry Carson     Terry Carson      652   09/09/81       10.00      25.00      200.00      235.00        0.00       235.00
                                                                                      ----------- ----------- ------------
          <TOTAL>                                                                          356.00       $0.00       356.00

MJ Record count = 2

              Perry Carson                06/07/81      121.00
              Perry Carson                09/01/81      100.00
              Terry Carson                09/01/81      100.00
                                                    -----------
<TOTAL #2>                                                 $321.00

June Carter      Jennett Carter    523   04/18/81       10.00     100.00        0.00      110.00        0.00       110.00

              Jennett Carter              04/18/81       10.00
              Jennett Carter              05/20/81       10.00
              June Carter                 03/21/81       50.00
              Jennett Carter              05/20/81       10.00
                                                    -----------
<TOTAL #2>                                                  $80.00

Jack Hill        Jill Hill          23   08/04/81       10.00      15.00      423.00      448.00        0.00       448.00

              Jill Hill                   09/02/81      100.00


Sam Lemmonay     Sam Lemmonay      363   11/26/81       10.00      42.00       46.00       96.00        0.00        96.00

              Sam Lemmonay                11/26/81       98.00


Danial K. Lesco Tim Lesco          651   09/08/81       10.00      52.00       65.00      127.00        0.00       127.00
Danial K. Lesco Tim Lesco          651   09/08/81       10.00      52.00       65.00      127.00        0.00       127.00
                                                                                      ----------- ----------- ------------
          <TOTAL>                                                                          254.00       $0.00       254.00

MJ Record count = 2

              Tim Lesco                   09/08/81      100.00


Leslie Paul      Leslie Paul        88   03/14/81       10.00      25.00        0.00       35.00        0.00        35.00

              Leslie Paul                 03/14/81       15.00


Tom Thumbdeldum Tom Thumbdeldum    854   05/0        10.00       20.00       35.00       65.00        0.00        65.00

                                                                                      =========== =========== ============
     <GND-TOTAL>                                                                          2244.00       $0.00      2244.00
                                                                                      =========== =========== ============

GR Record count = 12



                                                    ===========
<G TOT #2>                                             $879.00
```

Example 3: Sales commisions files

Background information:

Manufacturer Representatives sell their products to consumers as well as companies that use the products to complete various projects. This example is modeled after the second case. We have a manufacturer rep. firm that is owned by Mr. Black and employs three salesmen. These salesmen are responsible for different aspects and products used in the firm's projects. Each salesman will receive commissions on the part of the project and pruducts for which he has responsibility. These commissions will be paid as the customer pays Mr. Black's firm and the profits are realized. All three salesmen are paid the same commission rate on their share of the project, but this rate may vary with time and circumstances. Mr. Black wishes to keep track of all project profits, profits realized, commissions paid and commissions due.

Thinking out the application:

Because this application is heavily slanted towards figures, calculated and total fields will be used to track all figures with a minimum amount of user entry. The proposed file structure is shown next:

FIELD TITLES	L	T		FIELD TITLES	L	T	
1.) PROJECT NAME	20	A		16.) A-BASE	10	C	2
2.) PROJ #	6	N	0	17.) B-BASE	10	C	2
3.) CONTRACTOR	10	A		18.) C-BASE	10	C	2
4.) ORD.RECD	8	D	A	19.) COMM RATE	5	N	2
5.) PROJ. PROFIT	12	$	2	20.) A-COMM	10	C	2
6.) MARGIN%	4	N	2	21.) B-COMM	10	C	2
7.) TOTAL COMM.	12	C	2	22.) C-COMM	10	C	2
8.) PROFIT1 RECD	12	$	2	23.) A-COMM PAID	10	N	2
9.) PROFIT2 RECD	12	$	2	24.) B-COMM PAID	10	N	2
10.) PROFIT3 RECD	12	$	2	25.) C-COMM PAID	10	N	2
11.) TOT. PROF. RECD	12	T	2	26.) A-COMM DUE	10	C	2
12.) PROFIT DUE	12	C	2	27.) B-COMM DUE	10	C	2
13.) A % OF PROJ.	4	N	2	28.) C-COMM DUE	10	C	2
14.) B % OF PROJ.	4	N	2	29.) TOT COMM PAID	12	T	2
15.) C % OF PROJ.	4	N	2	30.) TOT COMM DUE	12	T	2

The calculated fields and total fields are outlined below:

```
Field  7 = Field#  5 * Field#  6
Field 12 = Field#  5 - Field# 11
Field 16 = Field#  7 * Field# 13
Field 17 = Field#  7 * Field# 14
Field 18 = Field#  7 * Field# 15
Field 20 = Field# 16 * Field# 19
Field 21 = Field# 17 * Field# 19
Field 22 = Field# 18 * Field# 19
Field 26 = Field# 20 - Field# 23
Field 27 = Field# 21 - Field# 24
Field 28 = Field# 22 - Field# 25

Total field #11 = 8,9,10
Total field #12 = 23,24,25
Total field #13 = 26,27,28
```

The file structure is explained as follows:

- Fields 1 and 2 are the project name and number.
- Field 3 is the contractor (customer) who purchases the products.
- Field 4 is the date the order for said product is made.
- Field 5 is the profit (before commissions) made on the project.
- Field 6 is the percent of the project profit that is to be divided among the salesmen.
- Field 7 is total commission base for the project.
- Fields 8, 9 and 10 are fields for each profit parcel after Mr. Black's firm has collected a payment.
- Field 11 is the total profit received (the sum of fields 8,9 and 10).
- Field 12 is the difference of fields 5 and 11.

Fields 13, 14, and 15 are the allocations of project profit given to each of the three salesmen.
Fields 16, 17, and 18 are calculated fields, they are the base amounts that the three salesmen will have their individual commissions calculated from.
Field 19 is the salesmen's commission rate for the project.
Fields 20, 21 and 22 are the commissions due each of the salesmen.
Fields 23, 24, and 25 are the respective commissions paid each salesman.
Fields 26, 27, and 28 are the commissions due each salesman.
Field 29 is the total of the commissions paid to all the salesmen.
Field 30 is the total of the commissions due all the salesmen.

The information in the previous file can be utilized to generate various types of lists and reports to help Mr. Black organize and interpret his data. On the following pages are two example applications of T.I.M.'s Report generator and the corresponding Report Library formats. The first example application is a sales commission report that itemizes the sales commissions accrued and paid to each salesman. The second example is a job profit report that details the project name, contractor, and project profit for each job.

```
                                          SALES COMMISSIONS REPORT
                                    SALES COMMISSIONS/JOB PROFIT FILE
                                                 12/17/81


    PROJECT NAME     TOT. PROF. RECD  A - BASE    B - BASE    C - BASE  A-COMM PAID B-COMM PAID C-COMM PAID A-COMM DUE B-COMM DUE C-COMM DUE
-------------------------------------------------------------------------------------------------------------------------------------------
James Towers                10000.00    1505.00     9030.00     4515.00      150.50      903.00      451.50       0.00       0.00       0.00
James Towers                 4500.00    1187.50     1662.50     1900.00      118.75      166.25        0.00       0.00       0.00     190.00
James Towers                    0.00       0.00      760.00      760.00        0.00       38.00       38.00       0.00      38.00      38.00
3rd St. Viaduct             32500.00    3350.00    10050.00    20100.00        0.00        0.00      500.00     335.00    1005.00    1510.00
3rd St. Viaduct              6510.00       0.00        0.00     6510.00        0.00        0.00        0.00       0.00       0.00     651.00
3rd St. Viaduct             30000.00   15225.00     3150.00     7875.00      751.50        0.00        0.00     771.00     315.00     787.50
Howell Basin                95000.00    9975.00     9975.00     8550.00      997.50      997.50      855.00       0.00       0.00       0.00
Peters                      42500.00    4603.50     4306.50     5940.00      400.35      355.00      450.00      60.00      75.65     144.00
Peters                      80000.00    8600.00     4300.00     4300.00      860.00      430.00      430.00       0.00       0.00       0.00
Gibbs Tunnel                15000.00    1748.25     1748.25     1753.50      174.82      174.82      175.35       0.00       0.00       0.00
Gibbs Tunnel                60000.00    8400.00     6300.00     6300.00      540.00      400.00      400.00     300.00     230.00     230.00
                                                                       =========== =========== =========== ========== ========== ==========
          <GND-TOTAL>                                                      3993.42     3464.57     3299.85    1466.00    1663.65    3550.50
                                                                       =========== =========== =========== ========== ========== ==========

GR Record count = 11
```

```
                                         JOB PROFIT REPORT
                                  SALES COMMISSIONS/JOB PROFIT FILE
                                              12/17/81


 PROJECT NAME    PROJ # CONTRACTOR ORD. RECD PROJ. PROFIT MARGIN% TOTAL COMM.  PROFIT1 RECD PROFIT2 RECD PROFIT3 RECD TOT. PROF. RECD
------------------------------------------------------------------------------------------------------------------------------------
James Towers     133823 Dunn       02/21/82     35000.00     .43     15050.00      5000.00      5000.00         0.00        10000.00
James Towers     133823 Edwards    02/21/82     11875.00     .40      4750.00      2500.00      1000.00      1000.00         4500.00
James Towers     133823 Evans      02/21/82      3800.00     .40      1520.00         0.00         0.00         0.00            0.00
                                            ------------                                                         ---------------
        <TOTAL>                               $50,675.00                                                                14500.00

MJ Record count = 3


3rd St. Viaduct 232422 American    12/19/81     67000.00     .50     33500.00     23000.00      4750.00      4750.00        32500.00
3rd St. Viaduct 232422 Dunn        12/19/81     18600.00     .35      6510.00      6510.00         0.00         0.00         6510.00
3rd St. Viaduct 232422 Youngstown  12/19/81     75000.00     .35     26250.00     10000.00     10000.00     10000.00        30000.00
                                            ------------                                                         ---------------
        <TOTAL>                              $160,600.00                                                                69010.00

MJ Record count = 3


Howell Basin    235478 Sanford     03/23/82     95000.00     .30     28500.00         0.00     45000.00     50000.00        95000.00


Peters          312444 Hapler Bro  12/12/82     55000.00     .27     14850.00     15000.00     15000.00     12500.00        42500.00
Peters          312444 Younger     12/26/82     80000.00     .21     17200.00     20000.00     30000.00     30000.00        80000.00
                                            ------------                                                         ---------------
        <TOTAL>                              $135,000.00                                                               122500.00

MJ Record count = 2


Gibbs Tunnel    834753 Haven       10/10/82     15000.00     .35      5250.00         0.00      7500.00      7500.00        15000.00
Gibbs Tunnel    834753 Hypergloss  12/26/82     60000.00     .35     21000.00     20000.00     20000.00     20000.00        60000.00
                                            ------------                                                         ---------------
        <TOTAL>                               $75,000.00                                                                75000.00

MJ Record count = 2


Gibbs Tunnel    834793 Underwood   10/10/82     65000.00     .27     17550.00     25000.00     20000.00     20000.00        65000.00
                                            ============                                                         ===============
    <GND-TOTAL>                              $581,275.00                                                               441010.00
                                            ============                                                         ===============

GR Record count = 12
```

Ex.—62

Example 4: Search firm applicant records

Background information:

Personnel search firms maintain lists of people and their skills who they have talked with in the past and who might be willing to consider a new job. The number of names in this data base might easily exceed five thousand, a difficult task of data management without a computer. When a new job order is received it must be compared against the known skills of existing applicants from the database to find those that seem qualified or nearly qualified. This is usually done by memory, a very inefficient method indeed. When a recruiter contacts a potential applicant again, he/she must update information on them such as job skills and salary. File folders on the applicants become a mishmash of scraps of paper written by various recruiters, making it difficult to determine what the actual current skills are. Mailings are also occasionally sent to all applicants to keep their interest up.

Thinking out the application

We need to set up a file to contain various types of information on the applicants. For the purpose of this example, let's assume that this search firm specializes in general office help, such as typists, secretaries, clerks, etc. Since the purpose of this file is ony to help recruiters find people who are probably qualified, we do not need to maintain large amounts of information on each person. All we need to know are the person's basic skills and salary, along with a few other minor determinates. We might break these down into three categories of information: Personal information, current company information, and work skills.

Personal Information:

Name
Address
City, State
Zip code
Phone (Home)
Phone (Work)
Spouse
Spouses' Occupation

High school
City, State
College
Years
Degree(s)
Special training
Other interests
Relocate/Where

Current company information:

Company
Years there
Current title

Job Description
Current salary

Work skills:

Typing rating
WPM
Mistakes
Shorthand Rating
WPM
Word processing rating
WPM

Filing experience
Telephone experience
Receptionist experience
Ten keypad experience
Current work type
Past work type
Comments

While this could be put into separate files, we will place it all into one for convenience sake. The following is the file structure:

Field Name	L	T	
1. NAME	25	I	
2. ADDRESS	30	A	
3. CITY, STATE	25	A	
4. ZIP	5	N	0
5. PHONE (HOME)	12	A	
6. PHONE (WORK)	12	A	
7. SPOUSE	25	I	

```
 8. OCCUPATION              40  A
 9. HIGH SCHOOL             30  A
10. CITY, STATE             25  A
11.  COLLEGE                25  A
12.  YEARS                   2  N 0
13.  DEGREE(S)              30  A
14.  COLLEGE                25  A
15.  YEARS                   2  N 0
16.  DEGREE(S)              30  A
17.  SPECIAL TRAINING       60  A
18.  OTHER INTERESTS        60
19.  RELO/WHERE             50  A
20.  CURRENT COMPANY        30  A
21.  YEARS THERE             2  N 0
22.  CURRENT TITLE          30  A
23.  JOB DESCRIPTION        60  A
24.  CURRENT SALARY          8  $ 2
25.  TYPING RATING           4  N 1
26.  WPM                     3  N 0
27.  MISTAKES                2  N 0
28.  SHORTHAND RATING        4  N 1
29.  WPM                     3  N 0
30.  WP RATING               4  N 1
31.  WPM                     3  N 0
32.  FILING EXP?             3  A
33.  PHONE EXP?              3  A
34.  RECEPT EXP?             3  A
35.  TEN KEYPAD EXP?         3  A
36.  WORK TYPE              25  A
37.  PAST WORK TYPE         60  A
38.  COMMENT                60  A
39.  COMMENT                60  A
40.  COMMENT                60  A
```

Fields 11 through 13 are repeated in 14 through 16 in case the applicant attended two colleges. Rating fields 25, 28, and 30 have numeric truncations of one for ratings of 7.5, 9.5, etc.

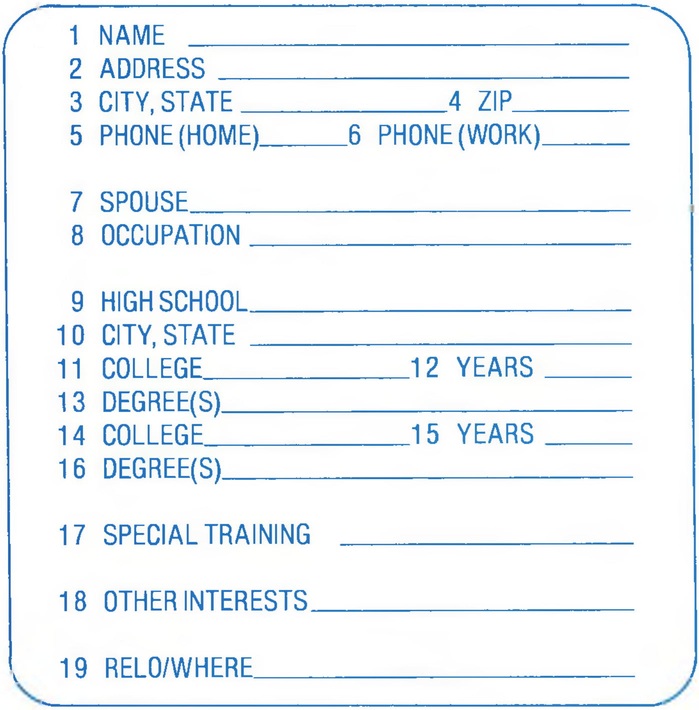
Getting the application on line

Since this is our fourth example, we will not go step by step for this creation process. Define the fields as indicated and then define the following key fields:

Major key	1		
Major key	4	Minor key	1
Major key	20	Minor key	1

We will define a custom screen to allow for easier data entry. All personal information will be on screen 1 and company and work information on screen 2.

SCREEN 1:



1 NAME ____________________

2 ADDRESS ____________________

3 CITY, STATE ____________ 4 ZIP ________

5 PHONE (HOME) ______ 6 PHONE (WORK) ______

7 SPOUSE ____________________

8 OCCUPATION ____________________

9 HIGH SCHOOL ____________________

10 CITY, STATE ____________________

11 COLLEGE ____________ 12 YEARS ______

13 DEGREE(S) ____________________

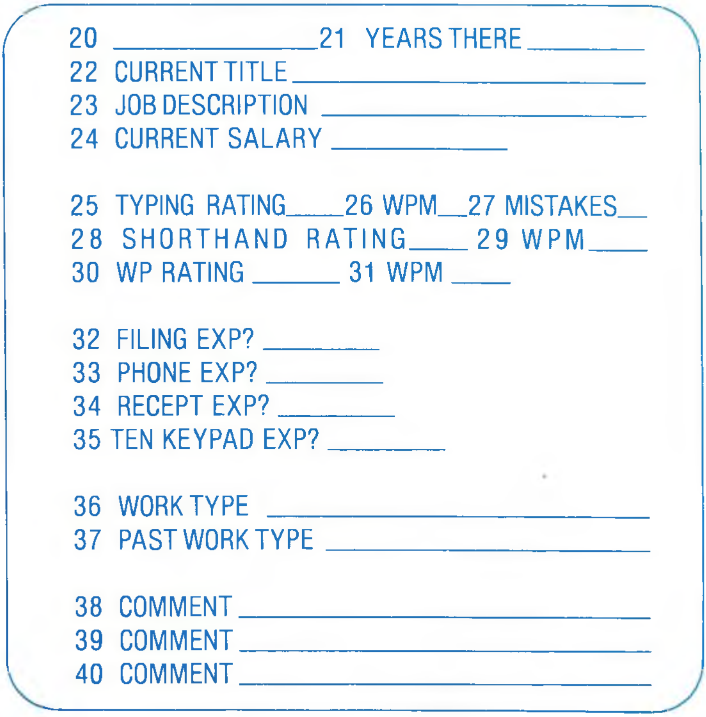
14 COLLEGE ____________ 15 YEARS ______

16 DEGREE(S) ____________________

17 SPECIAL TRAINING ____________________

18 OTHER INTERESTS ____________________

19 RELO/WHERE ____________________



20 ______________21 YEARS THERE ________
22 CURRENT TITLE ________________________
23 JOB DESCRIPTION ______________________
24 CURRENT SALARY ____________

25 TYPING RATING____26 WPM__27 MISTAKES__
28 SHORTHAND RATING____29 WPM____
30 WP RATING ______ 31 WPM ____

32 FILING EXP? ________
33 PHONE EXP? ________
34 RECEPT EXP? ________
35 TEN KEYPAD EXP? ________

36 WORK TYPE __________________________
37 PAST WORK TYPE ______________________

38 COMMENT ____________________________
39 COMMENT ____________________________
40 COMMENT ____________________________

We are now finished creating our file and you may add several example records for practice.

Example 4: Selecting records from the "SEARCH" file

What we want:

Let's say that an order has just come in from a client for a secretary with at least two years of college, at least 40 WPM on the typewriter and some experience as a receptionist. We would use the Select command to locate all people in our data base who meet this criteria.

The selection criteria:

First, we would find each of the individual requirements, and then put them all together.

At least two years college	FLD 12 GE 2,
At least 40 WPM typing	FLD 26 GE 40,
Experience as a receptionist	FLD 34 EQ YES,

If we put all the requirements together, we would have the following:

```
FLD 12  GE 2, AND FLD 26  GE 40, AND FLD 34  EQ YES,
```

These people may be sent to any of the outputs, such as terminal, printer or disk.

How about a request for someone with the following statistics:

Single
College degree
Office manager
Typing rating at least a nine
Shorthand rating at least an eight
Word Processing experience

Again let's define each criteria separately, then put them all together.

Single	FLD 7 EQ NULL
College degree	FLD 13 NE NULL
Office manager	FLD 22 EQ Office manager
Typing rating at least nine	FLD 25 GE 9
Shorthand rating at least eight	FLD 28 GE 8
Word processing experience	FLD 30 GT 0

When we put them all together they will look like this:

```
FLD 7  EQ  NULL, AND  FLD 13  NE NULL, AND  FLD 22  EQ Office manager,
AND  FLD 25  GE 9, AND  FLD 28  GE 8, AND  FLD 30  GT 0,
```

Let's do one last example, which follows:

Single
Some college
Might relocate
Curent salary between $8,000 and $10,000
Some experience in all of the following:
Typing
Shorthand
Filing
Receptionist

To draw these people from our data base, we would use the following search criteria:

```
FLD 7  EQ NULL, AND  FLD 14  NE NULL, AND  SUBFLD 19  EQ Yes, AND  FLD
24 BV 8000:10000, AND FLD 25  GE 1, AND  FLD 28  GE 1, AND  FLD 32  EQ
Yes, AND  FLD 34  EQ Yes,
```

This sums up our examples with *Select*. You may want to define a few more to sharpen up your knowledge of the various select operators.

Define and generate a list

What we want:

When a recruiter is preparing to call a potential applicant it is necessary for him/her to have some information about the person in front of them. For this purpose we will create a "data sheet" list format. It would be best to print a data sheet for each person selected by the search criteria to be distributed to the recruiters.

Defining the format:

We will want to print only one list across the page with 31 lines between lists. The format that you should use is shown on the next page:

Line#	Column#	Field#	Width	Text
1	1	1	25	
2	1	2	30	
3	1	3	25	
4	1	4	4	
5	1	5	12	
5	25	6	12	
6	1	7	25	
7	1	8	40	
8	1	9	40	
9	1	10	30	
10	1	11	25	
10	35	12	2	
11	1	13	30	
12	1	14	25	
12	35	15	2	
13	1	16	30	
14	1	17	60	
15	1	18	60	
16	1	19	50	

17	1	20	30
17	36	21	2
18	1	22	30
19	1	23	60
20	35	24	8
21	1	25	4
21	40	26	3
21	52	27	2
22	1	28	4
22	40	29	3
23	1	30	4
23	40	31	3
25	1	32	3
26	1	33	3
27	1	34	3
28	1	35	3
30	1	36	25
31	1	37	60
33	1	38	60
34	1	39	60
35	1	40	60

The above format will yield a usable data sheet. An excerpt from this list is shown on the next page. See if you can improve on it, especially with text entries for date of call, recruiter, etc.

```
Jane Hendricks
8816 Maple Dr.
Pineville, Oklahoma
69723
312/123-7786            312/444-6577
Adam Hendricks
Kelvin Auto Repair
Milliard Fillmore High
Pineville, Oklahoma
N.E. Oklahoma State               02
Associates Degree
                                  0

Emily Potter's Secretarial College
Needle point
No
Unemployed                         0


                                  0.00
6.4                                    45          1
7.2                                    80
0.0                                    0

yes
yes
yes
no


medium-sized office dealing with printing sales








Adam Johnson
1313 Mockingbird Lane
Souix Falls, North Dakota
76653
579/921-9823            344/762-3343
single

Great Bend City High
Great Bend, Montana
                                  0

                                  0

Devron Business School
Rodeo Rider in spare time
no
Evans Brodcasting Co.              7
Office Manager
manage an office
                                  14500.00
4.4                                    50          3
0.0                                    0
6.7                                    120

yes
yes
yes
yes
```

A Database Primer

Copyright ©1990 by Mark Ross

A good database or database management system (DBMS) can be very useful to you. Managers today are rarely in a 'hands on' situation. Instead, you use information given to you by others. Information becomes the input you use in making decisions. It is crucial to have good information. A well built DBMS will give you timely reports and information you need to make better decisions.

You get real benefits from a well organized DBMS:

1. Your data are well organized so that ...
    - you have a well defined structure for the data and you know where everything is.
    - you get repeatable results, because all users are accessing the same data and because standard reports provide the same analyses with new data.
    - you have well defined relationships among different kinds of data, allowing you to combine them in useful ways.
2. You get improved accuracy and confidence in your information because ...
    - data are stored where they're supposed to be, not in a pile on somebody's desk.
    - the data can be verified while they're being put into the system, not later, when you're trying to get an important report together.
    - there are virtually no arithmetic errors when summarizing data.
3. You get flexibility because ...
    - you can get new reports quickly, often in just minutes. They can be reviewed and changed quickly.
    - you can combine diverse data types from different sources, and create new kinds of information.
    - your data can be accessed by other tools such as graphics packages, word processors, and analytic tools such as statistical programs or spreadsheets.

The Database Structure

First, some definitions:

Data File: The collection of objects (the things you want to store) and their attributes. Example: sales data.

Data Record: Each occurrence of an object gets its own place in the data file. This record can be visualized as a row in the file. Example: individual items sold in a sales file.

Data Field: Sometimes called variables or columns, these are the attributes of the objects you are going to store. Example: an invoice number or customer ID in a sales file.

Relational DBMS: A collection of data files that have some common data, and that you want to combine. Example: sales, customer, and sales-rep files in a sales tracking system.

Data Structure: The description of the data files and their fields that are used in a particular database, in short, a description of the database.

How data are organized and kept in a database determine how the DBMS can be used to create information. A common mistake is to put as much data as possible in each record. For example, in a sales tracking system, you might be tempted to put the address for each customer in the same file as the data about individual sales transactions.

You would be better off keeping these in separate files. Why? Because you would be wasting space with duplicate data. Since you have a separate data record for each line item on an invoice, you would have to enter the address for each item sold. On the other hand, if you kept the address in a CUSTOMER file, you would only have to enter it once, when you first get a new customer. By only entering it once you eliminate opportunities for entering a wrong address. If you put it in the CUSTOMER file correctly the first time, you know that it's going to be correct every time you use it.

But how do you connect the data in the SALES file with the data in the CUSTOMER file? Technically, it varies from system to system, but they all require you to have some common data in the files you are relating. In the sales tracking example, you would have a CUSTOMER ID in both the CUSTOMER file and the SALES file. The system would combine the data from each file by matching the CUSTOMER ID each time you ask for a report.

Data Entry

Many people don't pay much attention to the data entry process when they think about putting together a DBMS. However, a good data entry program is essential to getting good information. After all, you build your information out of data, and you use that information to make decisions. It is important to have accurate information, which means having accurate data.

What you need is entry programs that check for obviously bad data. Let's continue with the sales tracking example. If you want to add a customer to the CUSTOMER file, you want to make sure that the CUSTOMER ID you're entering is not already being used. If the program doesn't check for duplicates, you would have to enter your data about the customer, print out the entire list, and then check it by hand. You don't have time for that, and you'd still probably miss some.

You also need to check the data when you are adding data to one file that needs to match data in another. Suppose that the sales tracking system has a SALES file to hold data about individual transactions. You want to make sure that your CUSTOMER ID in this file exists in the CUSTOMER file. Otherwise, you couldn't identify the customer or address, and wouldn't be able to collect your payment.

Another time to check the data is when you are entering numbers. You have to enter the SALES AMOUNT into your SALES file. But, you have to make sure that the value is reasonable. If your average sale is $200, you probably aren't going to have one as high as $100,000. A good entry program will let you set upper and lower boundaries on numeric data. It will also allow you to limit other kinds of data. For example, you might have stock numbers that start with the letter 'A', 'B', or 'C', followed by a three digit number between 100 and 400. Again, a good system will check for valid combinations.

Getting Information Out of the System

The purpose of any decision support tool is the creation and communication of information. A DBMS is no exception. You collect and organize data so that you can use it to create information. You might use that information for many purposes: operational control ("George, your group didn't meet quota last month, what happened?"), government reporting (taxes), as sales tools (to show how much better you're doing than the competition), or as decision making tools ("Widgets aren't doing too well, lets drop them.").

In order to use the information contained in the data, you must have some way to get it out with a reasonable effort, and within a reasonable time. Most off-the-shelf DBMS packages have some kind of system, or systems, for letting you do that.

There are many kinds of reporting tools. Most of them let you summarize the data in various ways. For example, you might want to look at total sales for the period along with breaks for customer subtotals. Virtually all commercial packages allow you to do that, with the option of showing each record for individual customers. To look at the sales of customers relative to each other, you might want to see each customer's sales as a percentage of total sales.

It is much easier for you to look at information that is selected or ordered in some way. Two of the most commonly used reporting tools in a DBMS do this for you. For example, you can look at all the customers grouped together alphabetically or by CUSTOMER ID. At the same time, you can select, or limit the report, so that you only see those who bought the product in the past month.

You will sometimes find it useful to look at more than one variable at a time. In the sales tracking example, you might want to see what kind of customers each sales representative sold to in addition to how much each sold. A cross tabulation table with sales representatives on one axis and sales volume by customer type or size groups on the other would give you that sort of information. Not all DBMS packages let can do this easily, or even at all.

There are times when you will want to combine data fields. Your SALES file might have fields for QUANTITY SHIPPED and QUANTITY BACKORDERED. For sales review and marketing management you really need to know how many were bought, not how many of those were shipped or backordered. Your DBMS should let you create a QUANTITY ORDERED field that does not have any actual data in it, but will sum the QUANTITY SHIPPED and QUANTITY BACKORDERED fields when you ask for a report. You only need to see the QUANTITY ORDERED field in the report.

You will need some reports on an ongoing basis. Virtually all reporting systems allow you to save a report format for reuse. As you get new orders, you can rerun the reports you have developed to track sales with the additional data in the SALES and CUSTOMER files. They will look similar to the old ones, but will give you the updated information.

There are times when you may want to create new data files from existing ones. After an order is shipped, you may decide not to keep a detailed record in the SALES file, but you may need to keep some information for long term tracking. You would probably want to keep some data on the number of items ordered by customer, but not need the data on the number shipped or backordered. Basically, you would want to store the QUANTITY ORDERED data but not the QUANTITY SHIPPED or QUANTITY BACKORDERED values. You would need to convert the computed value from the SALES file into an actual value in the history file. The more powerful DBMS programs have mechanisms for doing that. However, they usually require some programming expertise.

About the Author. Mark Ross is principal of Ross Econometrics, providers of decision support to management. Ross Econometrics has a tutorial disk available that contains working examples of decision support models and explanations. Call: (617) 489-5499, or write: Box 60; Waverley, MA 02179.