A MANUAL FOR ADAAS

Preliminary Version

February 1974

HIGHWAY SAFETY RESEARCH INSTITUTE
ANN ARBOR, MICHIGAN 48104

CONTENTS

FIGURES

TABLES

SECTION 1

INTRODUCTION AND SYSTEM OVERVIEW

INTRODUCTION

The Highway Safety Research Institute at the University of Michigan maintains an extensive collection of accident records that document important features of traffic collisions in many areas of the country. Due to the sheer volume of our accident records to date, it is necessary to utilize computerized storage and analysis of the data. The Statistical Research System was developed to provide the necessary analysis capability and is a descendent of the OSIRIS System originated at the University of Michigan. Some familiarity with the language of modern digital computer systems and the data analysis packages that may be used in conjunction with these systems is, of course, necessary for successful operation. If the necessary familiarity is overly complex, then the people who are most likely to benefit from the data will never achieve the proficiency required. In order to make the data as accessible as possible to those who can benefit from it, HSRI has developed ADAAS (Automated Data Access & Analysis System) to provide a simplified method of using the accident files in conjunction with the Statistical Research System. It is hoped that this feature will foster an increased usage of the accident data files to answer questions that arise in the course of highway safety related activities and to explore new problems that had not previously been uncovered.

HSRI obtains accident data from a variety of sources: the most prominent are the police agencies and the various Multidisciplinary Accident Investigation Teams. In this data, each accident is characterized by a number of factors that describe the driver, vehicle and environmental conditions recorded by the investigator. The volume of recorded accidents and the large number of factors for each accident make manual data handling procedures impossible. The accident data is consequently transcribed onto magnetic tape in a form that is usable on a digital computer.

If the stored information is to be of any value, it is necessary to have some technique for accessing the magnetic tapes and deriving from them the information that is desired by the data analyst. This function is performed for the user by the Statistical Research System (SRS)--a group of computer programs prepared for and maintained by HSRI for data manipulation and analysis. By proper use of SRS, the experienced user can create new data files, modify or update existing ones, and perform a number of analysis operations on existing files that range in complexity from listing a subset of the data for detailed scrutiny to complex statistical operations such as multivariate analysis of variance. The SRS in conjunction with the accident files comprises a powerful system for the collection and analysis of data.

The final component in the analysis hierarchy shown in Figure 1 is the computer and its executive supervisor that controls the operation of all functions resident on the machine. All HSRI accident files as well as the SRS are maintained on the University of Michigan's IBM 360/67 Computer under supervision of the Michigan Terminal System (MTS). MTS was developed for operation in a time-sharing mode, so that access to the computer could be gained from a variety of remote terminals for conversational usage of the computer in a real-time environment. By issuing the proper commands to MTS it is possible to sign on the system as a registered user and to perform all the operations necessary to access the desired data file, carry out the analysis required, and arrange for printout or storage of the results obtained.

While computerized storage and analysis is the only practical method of handling large data bases, it does introduce several difficulties for the user, who, in most situations, is not experienced in computer operations. Thus, the minute attention to detail required to operate computers tends to repel many potential users who are unaccustomed to this detail in human relationships. Computers are designed for detailed tasks, however, and there is no reason that they should not be given the task of

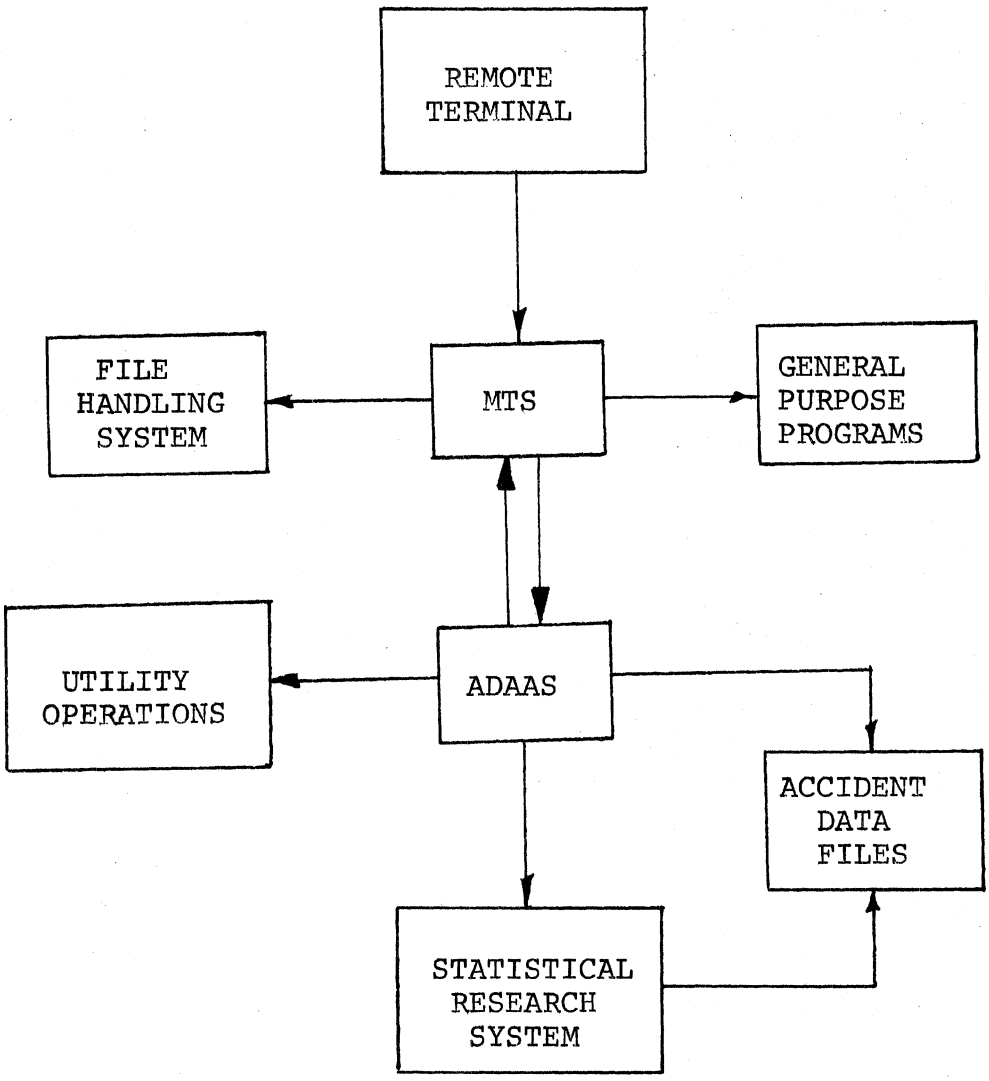

FIGURE 1 The Data Analysis Hierarchy

doing the operations that are difficult for the novice. The goal of the ADAAS System is therefore clear: use of the computer itself to perform most of the detailed operations necessary to carry out an analysis task using the HSRI accident files. In implementing this goal, however, it is difficult to allow for all the possible manipulations that can be performed with MTS and SRS. Consequently, ADAAS is presently designed to handle the routine operations normally encountered; the user is still encouraged to use the full capabilities of SRS to carry out more sophisticated analysis operations.

Before discussing the operational use of ADAAS, a presentation of the background information on MTS, SRS and the HSRI files necessary to successful use of the technique will be presented.

A SYSTEM OVERVIEW

As indicated in Figure 1, the HSRI accident data system is itself comprised of four major systems:

1. MTS - The Michigan Terminal System

   MTS is the controlling operating system for all tasks done at the University of Michigan Computing Center.

2. ADAAS - The Automated Data Access & Analysis System

   ADAAS is a sublevel operating system (within MTS) to supervise the tasks required for accessing the HSRI accident data files.

3. SRS - The Statistical Research System

   SRS is a package of analysis programs called by ADAAS to provide for analysis of the accident data.

4. HSRI Accident Files

   An extensive set of accident data files maintained by HSRI Data from country-wide sources and from a variety of investigative levels is incorporated.

It would be convenient (especially for the novice user) if the entire operation appeared homogeneous. To as large a degree as possible this has been accomplished: however, there are still many situations in which the division between systems becomes evident.

To understand what is happening in these situations it is useful for the user to have an appreciation of the role of each system in the overall operation. Consequently, a descriptive discussion of each system is presented in this section. The detailed instructions for operation of each system is the subject of later sections.

1. The Michigan Terminal System

The Michigan Terminal System (MTS) is an executive program responsible for interfacing the University of Michigan Computer with the needs and requirements of users. As such, it is the lowest level (or most basic) system that the user normally encounters. This means that if any serious errors or inconsistencies occur in the operation of higher level programs that the programs themselves cannot handle, control is returned ultimately to MTS.

Aside from handling abnormal conditions that may occur, the user generally relies on MTS to perform the following tasks:

a. Sign on and off the computer.
b. Maintain an accounting of the work done and costs incurred.
c. Provide for the storage of information.
d. Provide for the printing (retrieval) of information.
e. Supervise the operation of sub-systems designed to accomplish specific tasks.

The full capacity of MTS in accomplishing these tasks is beyond the scope of this text. See Reference [1] for a complete discussion.

2. The Automated Data Access & Analysis System

The data management program ADAAS operates within MTS to provide a convenient means of accessing the HSRI accident data files, performing the desired analysis operations, and printing the results obtained. User interaction with ADAAS is similar in many ways to interaction with MTS; that is, the user supplies one of a number of possible legal system commands. Upon decoding and

accepting the command, the requested operation is performed and the system returns to the user to wait for a new command.

Commands in ADAAS may be grouped into three general categories: (a) informational, (b) data access, and (c) data analysis. Informational commands partially implement the self-describing features of the ADAAS program and provide the user with current on-line system condition and operation information.

One of the most important ADAAS features is the data access command that permits users to access any available HSRI data file by specifying a data base keyword (i.e., an eight-letter codeword designating the file). Since most HSRI data files are stored on magnetic tape, ADAAS supervises any tape mounts that are necessary.

Once data access has been obtained, ADAAS may be called upon to perform a number of selected analysis operations. Here again the analysis programs are selected by specifying a four-letter codeword designation for the desired program. Once the necessary setup operations have been completed, ADAAS calls upon the HSRI Statistical Research System (SRS) to perform the actual analysis. When analysis is complete, control is returned to ADAAS. At this point further analysis can be performed on the file that is currently being accessed, or a new file may be used.

SECTION 2

THE MICHIGAN TERMINAL SYSTEM (MTS)

The University of Michigan computer system can service a large number of remote users concurrently, offering each a wide variety of services. The task of keeping track of all the programs in the machine and devoting some attention to each of them on a regular basis is handled by the MTS operating system. In order to request service from the computer, a potential user must first identify himself as a registered account holder and then communicate his requests via the MTS Command Language.

Two operational modes within MTS are possible: conversational or batch. In the conversational mode, a remote terminal is used to communicate directly with MTS through a typewriter-like keyboard in conjunction with a telephone. Each command is processed as it is received and the results are reported back to the user as they are generated. Since each command may be based on previously obtained results, the conversational mode of operation is highly interactive.

Batch mode, on the other hand, is non-interactive. All requests must be completely preplanned and submitted to a remote batch terminal as a single job. Some time later--minutes or hours, depending on program length and computer processing load--the results may be picked up at the batch terminal.

The command language and its usage is essentially the same for both batch and conversational modes. The primary difference is that MTS replies after most commands in the conversational mode confirming the action that it has just taken, notifying the user of certain errors in the command, or requesting additional information that was not supplied. This feature is very useful when errors are made in the instructions given to the system. For example, if certain words are misspelled and MTS is unable to interpret the command, an error message will be issued. In some cases the error can be identified and the user will then be prompted for new information. In addition, if MTS is asked to empty or destroy files (that may contain important information)

a confirmation request is issued before the operation is carried out, thereby allowing the user to change his mind.

Although the conversational mode is convenient, especially for the novice users, batch operation has definite advantages under certain conditions. Tasks requiring a large amount of setup information that must be run several times with only minor variations are good candidates for batch jobs. In addition, when computer loading is heavy, it is often more convenient to enter a batch job and pick up the results the next day than to attempt to complete the work at a terminal.

The discussion and presentation of examples in this manual will be oriented toward conversational usage, but operation in batch will be described. Our examples will consist of data entered by the user and data returned by the computer. In order to distinguish the two, user supplied input lines will be underlined while machine supplied output will not. In practice, this underlining must not be used.

A. TERMINALS

Communication with MTS is generally carried out by means of a terminal connected to the computer through telephone lines. The remote terminal may be as simple as a touch-tone telephone or as complex as a remote computer; at the intermediate level of complexity and expense are the typewriter-like terminals (such as Teletypes, the IBM 2741, and the IBM 1050) and certain cathode-ray display terminals. The Teletype terminal device may be either a Model 33, 35, or 37 and may be ASR (automatic send and receive) or KSR (keyboard send and receive). The ASR models are equipped for use with paper tape.

In addition to the many types of terminal devices, there are several different types of control units used to connect the devices to MTS. The two most commonly used transmission control units are the Memorex 1270 Transmission Controller and the PDP-8 Data Concentrator. Each transmission control unit handles terminal devices differently; therefore, there are restrictions in their use. The Data Concentrator handles a wider variety of

terminal devices than the Memorex 1270 and supports paper tape equipment of the ASR model Teletypes. In addition, the Data Concentrator supports a wide variety of more sophisticated terminal devices not described here.

Common terminal devices supported by MTS are described below:

1. The IBM 2741 Communications Terminal is very much like an ordinary electric typewriter. Its character set includes all commonly used text and programming symbols. Its output speed is about 14 characters per second and its carriage width is 130 characters.

2. The Model 33 Teletype is a commercial telecommunications unit which can be connected to the computer via telephone lines as the IBM 2741. Its output speed is 10 characters per second and its carriage width is 72 characters. It does not provide lower-case alphabetics. It is, however, the most common and the most inexpensive terminal available.

3. The GE TermiNet 300 is a high speed terminal that can operate at 10, 15, or 30 characters per second with a carriage width of up to 118 positions.

Physical characteristics of the various teletypes, the IBM 2741, and the GE TermiNet 300 are described in Appendix A.

Each of the terminal devices has a set of special control characters that instruct MTS to perform special functions (generally in reference to data input or output operations). These terminal control characters are shown in Table 1.

The table indicates the control characters to:

-DELETE THE LAST CHARACTER TYPED: On the Teletype, press CONTROL-H. Nothing will be printed on the paper as a result of this operation, and the printing element will not be backspaced. MTS automatically will have removed the last character typed. To delete the last n characters, enter CONTROL-H repeatedly, n times. On the IBM 2741, hit the backspace key. The print element will backspace, and the user may continue by typing over the deleted character(s).

TABLE 1

Terminal Control Characters

FUNCTION	TELETYPE (763-0300)	TELETYPE (763-1500)	IBM 2741 (763-0300)	TERMINET 300 (763-0300)	TERMINET 300 (763-1500)
Character Deletion	CTRL-H	CTRL-H	BACKSPACE	BS (backspace)	BS (backspace)
Line Deletion (Follow by line terminator)	CTRL-N	RUBOUT	UNDERSCORE (_)	CTL-N	DEL (delete)
Line Termination*	RETURN	RETURN	RETURN	RETURN	RETURN
Attention Interrupt	BREAK	BREAK	ATTENTION	INTERRUPT	INTERRUPT
End of File	CTRL-C	CTRL-C	¢	CTL-C	CTL-C

(NOTE: CTRL-CHAR or CTL-CHAR indicates the character formed by first depressing the Control Key, then the character indicated.

*Other characters are possible, but the indicated control keys are preferred.

-DELETE A LINE: All characters already typed in the current line: Press CONTROL-N followed by RETURN on the Teletype. MTS will return "LINE DELETED" and will type a pound sign (#) on the next line, indicating that a new line can be entered. On the IBM 2741, the shift key should be pressed down and the dash character key pressed simultaneously, (underscore character) followed by the return key, in order to delete a line. MTS will respond with: LINE DELETED.

-END-OF-LINE: TERMINATE A LINE OF TYPED CHARACTERS: On the Teletype, or the IBM 2741, the return key should be pressed. MTS will then accept all characters typed in the current line as the total contents of that line. MTS will then process this line and return a pound sign (#) indicating that another line can be typed by the user. During the running of an analysis program, the carriage will return to the left margin without printing a pound sign (#), indicating that another line can be entered if requested by the program.

-INTERRUPT ANY CURRENT PROCESSING ACTION: (For example, stop an analysis program or stop the copying of a file): Press the BREAK key on the Teletype or the ATTENTION key on the IBM 2741. When the Teletype BREAK key is depressed, the BRK-RLS (break-release) key on the console will light up if the machine is equipped with this feature and the keyboard will be disconnected. To reactivate the keyboard, press the BRK-RLS key. This operation is known as an ATTENTION INTERRUPT. A more detailed discussion of this very useful feature and the actions which it may invoke are described in Section 2-J.

-GENERATE AN END-OF-FILE (EOF): The MTS command "$ENDFILE" can be used. In addition, CONTROL-C can be used on the Teletype or "¢" can be used on the IBM 2741.

B. SIGNING ON AND OFF

The first steps in using MTS are to turn the terminal power on, make certain the device is in the "communicate" and not the "local" mode, and then dial the telephone number that establishes

the computer link. A list of telephone numbers that are appropriate for the terminal used is shown in Table 2.

TABLE 2

MTS Telephone Numbers

Memorex 1270	(313) 763-0300
Data Concentrator	(313) 763-1500
MTS STATUS	(313) 763-0420

If the modem used with the terminal has a duplex switch, it should be in the half-duplex position for the 3-0300 line and in the full-duplex position for the 3-1500 line. The exact method of dialing will depend on the terminal:

1. With a regular telephone and separate modem simply dial the number and place the headset in the modem cradle.
2. For the Bell Dataphone, press the TALK/CLEAR button, dial the number, press DATA when a steady tone is heard and put the headset in the cradle.
3. For Teletypes with a built in coupler and phone, press ORIG and dial the number.

When the phone connection is established, it is necessary to enter the word "GO" when using the 763-0300 line (this is not necessary for the 763-1500 line). MTS will output some descriptive header material that depends upon the terminal used and the communications devices connected to. Examples 1 and 2 show typical replies for a Teletype and for an IBM 2741, respectively, on the Memorex 1270.

EXAMPLE 1:
```
GO
MTS (LA35-0060)
WHO ARE YOU?
UMHYSRI A AA
#NEXT EXPECTED SHUTDOWN: 12:30 PM
#
```

EXAMPLE 2:
```
GO
MTS (LA04-0037)
#NEXT EXPECTED SHUTDOWN: 12:30 PM
#
```

In these examples, the EXPECTED SHUTDOWN message is informative and does not command user compliance. The only operator intervention required here is the entry of "GO" that identifies the type of terminal that is being used. The "WHO ARE YOU?" question in Example 1 is taken care of by an answerback device in the Teletype.

The final pound sign "#" in the initial MTS connect message indicates that you are properly connected to the system and that MTS is waiting to receive its first instruction. The first command that is issued must be a valid signon command with the appropriate user number obtained from the Computing Center or from HSRI. The machine will respond with a request for the user password as shown in Example 3 (IBM 2741) or Example 4 (Teletype):

EXAMPLE 3:
```
#$SIGNON MYID
#ENTER USER PASSWORD
?MMMMMMMMMMMMM
```

EXAMPLE 4:
```
#$SIGNON MYID
#ENTER USER PASSWORD
?_____
```

The password is a "combination lock" that prevents unauthorized use of your ID number and should consequently always be concealed. For an IBM 2741 the password may be entered in the obscured area provided by the computer. On a Teletype or other terminal, the duplex switch should be placed in the full-duplex position for the Memorex 1270 or the ESCAPE key depressed for the Data Concentrator and the password typed in. In either case, if the password is correct, MTS will respond with the confirmation message shown in Example 5.

```
EXAMPLE 5:   #**LAST SIGNON WAS:  (Time)    (Date)
             #USER "MYID" SIGNED ON AT  (Time)  ON  (Date)
             #
```

Rule 1: The first character of a command line to MTS is a "$". Although there are cases in which the dollar sign need not appear (e.g., see Section 2-K), it is always acceptable to include it.

Rule 2: The first MTS command must be a $SIGNON followed by a space and then the user ID.

When you get an ID number from the Computing Center, a password will already be assigned to it. This should be changed to something safer, more meaningful, and more easily remembered. The $SET command is used to change your password. To change your current password to a new one called NEWPASS use the command shown in Example 6.

```
EXAMPLE 6:   #$SET PW=NEWPASS
             #
```

The $SET command can be the part of any job. The password should be no longer than twelve characters and must not contain any blanks.

The last command to MTS must be a $SIGNOFF command. This indicates that the user is finished using the system. At signoff, a number of pertinent statistics that describe the job are printed out at the terminal. A typical signoff sequence is shown in Example 7.

```
EXAMPLE 7:   #$SIGNOFF
             #OFF AT 16:55.26          12-16-71
             #ELAPSED TIME             27.15 MIN.        $1.35
             #CPU TIME USED            20.371 SEC.       $1.77
             #CPU STOR VMI             10.083 PAGE-MIN   $ .53
             #WAIT STOR VMI             6.558 PAGE-HR.
             #DRUM READS              1382
             #APPROX. COST OF THIS RUN IS   $3.64
             #DISK STORAGE             24.567 PAGE-HR
             #APPROX. REMAINING BALANCE:   $33.03
```

Rule 3: The last command must be $SIGNOFF to tell MTS that you have finished using its facilities. MTS will return with statistics on the completed run showing how much the run cost, the time it required, etc.

If the full set of signoff statistics are not desired, it is possible to use the commands $SIGNOFF SHORT or $SIGNOFF $ to obtain an abbreviated printout. In particular, $SIGNOFF $ is quite useful and prints out only the cost of the run and the approximate remaining balance. Example 8 illustrates this procedure.

EXAMPLE 8:
```
#$SIGNOFF $
#OFF AT 09:04.40    01-11-72
#   $.50
# $20.98
```

So far, several words of the MTS command language have been encountered (SIGNON, SIGNOFF, SET). There are many more, some of which will be introduced in the sections that follow. In general, the command statements have a fairly free format. A safe rule is:

Rule 4: Don't leave a space between the dollar sign and the command word. In general you should leave a space between "words" when there is no explicit separator, such as in $SIGNON MYID. However, if there is a separator such as an equal sign (=), you should not leave any spaces around it. For example, in the phrase "PW=PASSWORD" there should be no blanks.

C. STORING INFORMATION IN MTS

Information is commonly stored in some device that is referred to as a file. However, this is an overworked term and may lead to some confusion. We have seen in the Introduction how the accident information is stored on magnetic tapes that are referred to as "Accident Files." Similarly, the Computing Center maintains programs and information on "Public Files" for utilization by all

computer users. Finally, the user himself may create "Private Files" on his own user account to store the results of the analysis operations that he has performed on the ADAAS system. These "files" are created and maintained by the user, and the Computing Center charges a storage fee that is proportional to the size of the storage space used (see Example 7). Storage size is measured in PAGES of information. One PAGE may contain up to 4096 characters and closely corresponds to 60 lines of data, each line containing 68 characters. An MTS PAGE is thus approximately equal to a full physical page of teletype output.

MTS records information on disk storage devices. A data file may be thought of as an area of this disk where information can be stored. Every file has a name (up to twelve characters long) which one uses to communicate to MTS about that particular file. Files may be PUBLIC (usable by everyone) or PRIVATE. We will discuss public files later; for now, attention will be directed to private files--those files created and manipulated by users for their particular purposes.

Private files belong to a specific user and may be accessed only by him unless he gives explicit permission to others. There are two types of private files, PERMANENT or TEMPORARY. Permanent files must be explicitly created by the user and exist until destroyed. Temporary files may be created by the user or by MTS when their names are first encountered by the system. More importantly for the user, they are automatically destroyed when the user signs off. Temporary files are distinguished by file names that begin with a minus sign (-). Thus, MYFILE is the name of a permanent file while -TEMP denotes a temporary file.

Files are composed of lines of information. If the information is put into the file by an SR System program, a line is generally the information contained in one SR System output record. If the information is put in the file by the user via a typewriter terminal, a line is the information typed before the line return code is entered (CONTROL-S or RETURN). The line number for a given line may have any value in the range -99999.999

to +99999.999. For most purposes, line numbers start with 1 and increase consecutively by integer values. Information for different line numbers may be entered or taken out of the file in any order.

Before using a file, it is necessary to create it. While MTS will automatically create temporary files when necessary, it is always possible to create a file with the $CREATE command. In the examples that follow, it will be assumed that the user has correctly signed on the system and that MTS has responded with a pound sign (#) and is ready to receive commands.

To create a permanent file called MYFILE it is only necessary to issue the command $CREATE, followed by the desired name. The procedure together with the MTS response is shown in Example 9.

EXAMPLE 9:
```
#$CREATE MYFILE
#FILE "MYFILE" HAS BEEN CREATED.
#
```

If the name of the file is not acceptable to the system, or if no file space is available on your account, an appropriate error message will be issued.

Rule 5: A file of modest size can be created by the $CREATE command followed by a space and then the name that you wish to attach to it. A permanent file will be retained for your use until destroyed.

Before storing information in a file, it is necessary to make sure it is the active file. The active file denotes the storage space where MTS puts information lines that are submitted to it. Any file that exists on the user account can be made the active file by using the $GET command. The procedure is shown in Example 10.

EXAMPLE 10:
```
#$GET MYFILE
#READY
#
```

Rule 6: The $GET command causes the file referenced to become the active file for the system. This step is necessary prior to adding information to, or correcting information in, an existing file.

If the file is to be used for storage of data generated by a program, then the information will be put into the file in a format that is under program control. If the information is to be entered by the user, then two courses of action are possible. If the lines are to be entered in a consecutive fashion, then we can use the $NUMBER command to automatically number the lines and enter the information in the active file. Example 11 shows a simple job that creates a file called FILEFACTS and then fills it with six lines of information.

EXAMPLE 11:

```
#$CREATE FILEFACTS
#FILE "FILEFACTS" HAS BEEN CREATED.
#$GET FILEFACTS
#READY.
#$NUMBER
#      1_THIS SAMPLE FILE CONTAINS SOME
#      2_USEFUL BUT IRRELEVANT
#      3_INFORMATION ON FILES IN MTS.
#      4_  1) A FILE IS A SET OF NUMBERED
       5_LINES STORED UNDER SOME UNIQUE
       6_NAME.
       7_$UNNUMBER
```

Rule 7: The command $NUMBER instructs MTS to number the lines of data that follow and to put these lines in the active file. MTS types the line number and an underline character (_) and assigns the contents of the line entered to the given line number in the active file. The numbering starts with 1 and increases in unit increments (i.e., 1,2,3; etc.)

Rule 8: Lines beginning with a dollar sign in the first position are considered to be commands and are executed by MTS. If it is necessary to enter a dollar sign into the first character of a line in the active file, a double dollar sign must be used. Thus, in Example 11, the line

```
#      7 $$UNNUMBER
```

would cause line 7 of the file to contain the phrase "$UNNUMBER".

Rule 9: The command "$UNNUMBER turns off the automatic line numbering feature.

Data may also be added to the active file by entering the line number, a comma, and the data line directly to MTS. Thus, if the file FILEFACTS was an existing file, the first line of Example 11 could be entered by the following set of commands:

EXAMPLE 12:

```
#$GET FILEFACTS
#READY
#1,THIS SAMPLE FILE CONTAINS SOME
#
```

The comma is a delimiter to separate the line number information from the contents of the line (which may begin with a number).

Rule 10: All lines of information which (1) start with a line number (either explicitly or via $NUMBER) and (2) do not have a single "$" following the number are put into the currently active file. Any line not starting with a line number or containing a single "$" after a line number is assumed to be a command to be executed immediately.

It is often necessary to modify, or revise, the contents of a file. A drastic modification is the elimination of all the information that currently exists in a given file. This is accomplished as shown in Example 13.

EXAMPLE 13:

```
#$EMPTY MYFILE
#FILE "MYFILE" IS TO BE EMPTIED. PLEASE CONFIRM.
?OK
#DONE
#
```

Rule 11: The $EMPTY command will delete all the contents of the file specified, but will not destroy the file or modify its size (i.e., a large file remains large!).

Rule 12: Any response to the confirmation request, other than "OK", "O.K.", or "!" will abort the command. To save time, the OK may be entered directly after the filename, i.e., $EMPTY MYFILE OK.

Less drastic modifications may be performed by the procedure for entering new data shown in Example 12. For instance, if we want to change the information in file "FILEFACTS" generated in Example 11 we could use the procedure:

EXAMPLE 14:
```
#$GET FILEFACTS
#READY.
#6,NAME ASSIGNED BY THE USER.
#2,
#3.5,YOU SHOULD LEARN THEM.
#
```

This sequence of instructions places a line numbered 3.5 between lines 3 and 4 and replaces the contents of line 6 with new information. Line 2 is deleted by terminating the line immediately after the comma. Note that it is not necessary to enter the information in line number order.

Remember from Rule 10, that MTS places all input lines that are interpreted as data lines into the currently active file. To protect files from having data stored inadvertently, it is good practice to release files from active status after the desired information has been entered. This is done with the $RELEASE command. Example 15 shows the use of the $GET and $RELEASE commands to assign the active file.

EXAMPLE 15:

```
#1,THIS IS AN INPUT DATA LINE !
#THERE IS NO ACTIVE FILE TO PUT THAT IN.
#$GET FILE
#READY
#1,THIS IS AN INPUT DATA LINE !
#$RELEASE
#1,THIS IS AN INPUT DATA LINE !
#THERE IS NO ACTIVE FILE TO PUT THAT IN.
#
```

Once the data in our files have been modified and are ready for use, it is often necessary to list the file on the terminal so that we can examine the entire contents and see what line number corresponds to each data line. Similarly, we might desire to copy the information to another file. These operations are performed by the $LIST or $COPY commands. Before discussing these commands, however, it is useful to introduce the concept of reference to portions of a line file.

In many operations, we are not interested in the entire contents of a file, but only that information contained in a given line number range. Thus, we might want to know what is contained in the first five lines of the file, or perhaps in the last five lines. The portion of the file that is of interest can be specified by appending the line number range of interest. That is,

```
MYFILE(1,5)
MYFILE(20.6,399.2)
```

With this technique the last line is denoted by LAST so the third and fourth lines from the end of MYFILE can be referenced by

```
MYFILE(LAST-4,LAST-3)
```

If the last line of the file is one of the line number delimiters, it can be omitted in the specification. Thus, the last five lines of MYFILE can be referenced by

```
MYFILE(LAST-4)
```

As a final note, the contents of the file from line number one to the end of the file are referenced by the filename with no appended delimiters. For example,

```
MYFILE
```

refers to the contents of MYFILE from line number one to the end of the file. Since line numbers may be less than one, this may not reference the entire file.

With this convention out of the way, the list and copy commands are straightforward. To list the contents of a file (line numbers and line content) use the MTS command

```
          $LIST MYFILE
or,       $LIST MYFILE(J)          J,K=Line Numbers
or,       $LIST MYFILE(J,K)
```

For example, we can list the contents of "FILEFACTS" to obtain the following printout.

```
EXAMPLE 16:  #$LIST FILEFACTS
             >      1     THIS SAMPLE FILE CONTAINS SOME
             >      3     INFORMATION ON FILES IN MTS.
             >      3.5   YOU SHOULD LEARN THEM.
             >      4      1) A FILE IS A SET OF NUMBERED
             >      5     LINES STORED UNDER SOME UNIQUE
             >      6     NAME ASSIGNED BY THE USER.
             #END OF FILE
             #
```

Rule 13: The $LIST command, followed by a file name (with or without line number modifiers as explained above) causes the line number and full contents of each line to be printed at the terminal.

Similarly, we can use the copy commands

```
          $COPY MYFILE
or,       $COPY MYFILE(J)          J,K=Line Numbers
or,       $COPY MYFILE(J,K)
```

to copy the contents of the file without the line number information to the terminal. Again using the FILEFACTS example, we could print out the contents from line 3.5 to the end of the file as shown in Example 17.

EXAMPLE 17:
```
#$COPY FILEFACTS(3.5)
>YOU SHOULD LEARN THEM.
>1) A FILE IS A SET OF NUMBERED
>LINES STORED UNDER SOME UNIQUE
>NAME ASSIGNED BY THE USER.
#
```

The $COPY command in general will transfer the contents of one file to another. Thus, we may use

EXAMPLE 18:
```
#$COPY FILEFACTS TO NEWFILE
#
```

to copy the contents of FILEFACTS to another file called NEWFILE.

Rule 14: The $COPY command transfers the contents of one file into another. In this process, the lines are numbered sequentially from 1 as they enter the new file regardless of what their line number may have been in the old file. The old file is retained unaltered.

Rule 15: When copying to a terminal or line printer, the first character of each line is examined to see if it is blank or if it is a carriage control character (0,-,+,&,9,1,2,4,6,8,;,<). If one of these characters is encountered, the appropriate carriage control action is taken (double space, new page, etc.), and the remainder of the line is printed as it is. If the first character is not recognized as a carriage control character, then the full line is printed.

The effect of Rule 15 can be seen in a comparison of Examples 16 and 17. In listing FILEFACTS, we see from Example 16 that line 4 begins with a blank and that the "1" lines up with "0" in line 3.5.

Copying the file, however, as shown in Example 17, indicates that the "1" now lines up with the "Y". This occurs because the blank at the beginning of the line 4 was interpreted as a no-action carriage control character. The true line contents may be obtained by using the $LIST command.

The final operation in file handling that we will discuss is the destruction of a file and its contents. Each file that is maintained on a user account takes up a part of the user's allocated file space and also accrues a tenancy charge that depends on the size of the file. For these reasons, it is desirable to destroy all files that are not needed. To do this, proceed as in Example 19.

EXAMPLE 19:
```
#$DESTROY MYFILE
#FILE "MYFILE" IS TO BE DESTROYED. PLEASE CONFIRM.
?OK
#DONE.
#
```

Rule 16: The $DESTROY command deletes the entire contents of the file specified, and removes the name of the file from the user catalog.

Rule 17: Any response to the confirmation request, other than "OK", "O.K.", or "!" will abort the command. To save time, the OK may be entered directly after the filename, i.e., $DESTROY MYFILE OK.

Information about the files currently maintained on a user ID may be obtained by use of the MTS command $FILESTATUS. This command has many functions and a diversity of operations to generate any piece of information about files that is available to the user; for a complete description of this command, see [1]. To obtain a list of the permanent files on a user account, use the $FILESTATUS command as shown in Example 20.

EXAMPLE 20:
```
#$FILESTATUS
 BATCH    CARDIC    SETUP    TBBL    ZFILE
#
```

Size information may be obtained for a given file by adding the name of the desired file and the modifier "FILE" as shown in Example 21.

EXAMPLE 21:
```
#$FILESTATUS TBBL FILE
TBBL          SIZE=9P, MINSIZE=2P, TRUNC=2P, MAXSIZE=255P
 LINES=48, AVLEN=65, MAXLEN=126, MAXHOLE=255
#
```

In the printout the following information is presented:

SIZE	Current size of the file (in pages)
MINSIZE	Minimum size of the file if all the information were compacted (no holes)
TRUNC	Minimum size that the file can be truncated to
MAXSIZE	Maximum possible size of the file
LINES	Number of lines
AVLEN	Average length of lines
MAXLEN	Length of the longest line
MAXHOLE	Size of the largest hole

The same information for all files on the user ID may be obtained by substituting an asterisk for the file name. That is,

```
#$FILESTATUS * FILE
```

D. RUNNING UTILITY PROGRAMS

Up to now we have discussed operations that have been performed under the direct control of MTS. It is possible to cause MTS to relinquish control to an operating program specified by the user. These programs are usually written in some programming language such as FORTRAN, BASIC, etc. The user may store one of these programs in a file called, for example, PROGRAM. To execute this program, it is only necessary to use the command $RUN as shown in Example 22.

EXAMPLE 22:
```
#$RUN PROGRAM
#EXECUTION BEGINS
[Machine under program control]
#EXECUTION TERMINATED.
#
```

Rule 18: The $RUN command causes the program stored in the referenced file to be loaded in the machine and executed. Operation of the machine is then under control of the program until execution is terminated and control is returned to MTS.

The real utility of the $RUN command for the non-programmer comes about from the fact that the University Computing Center maintains a large number of operating programs that may be accessed by any user. These programs are stored in the Public Files that were discussed at the beginning of this section. A public file has a name that begins with an asterisk (*); thus, *STATUS is the name of a public file that contains a program to present the current status of the user's account. A number of these files that are of interest to the ADAAS user are described below.

1. *STATUS

   Reports the amount of funds and storage assigned, used, and remaining.

EXAMPLE 23:

```
#$RUN *STATUS
#EXECUTION BEGINS
STATUS OF MYID AT 06/04/73     13:48.00    USED    MAXIMUM    REMAINING
CUMULATIVE CHARGE                ($)      254.25    300.00      45.75
PERMANENT DISK SPACE           (PAGES)       32        50         18
CURRENT SIGNONS                                1         1          0
CUMULATIVE TERMINAL TIME         (HRS)     13.53

#EXECUTION TERMINATED
#
```

2. *USERS

Reports the current load of users on the computing system in terms of terminal users, batch jobs, and non-MTS tasks.

EXAMPLE 24:
```
#$RUN *USERS
#EXECUTION BEGINS
THERE ARE 39 TERMINAL USERS, 5 BATCH TASKS, 39
AVAILABLE LINES, AND 12 NON-MTS JOBS USING 1208
VIRTUAL PAGES AND 274 REAL PAGES.  HARDWARE IS
CPU'S P1 P2, STORAGE A B C D E F G H,CCU'S 0 1
#EXECUTION TERMINATED
```

A total of 44 jobs are being processed by MTS at the moment, 39 from remote terminals and 5 entered on cards at the Computing Center for batch processing. The additional 12 non-MTS jobs are major system communication and storage management routines. Thus the multi-programming system is processing a total of 56 active programs. Of the total virtual memory of approximately 2200 pages capacity (900 each on the two drums and 384 pages of core memory), 1208 were being used when *USERS was being executed; 274 of the 384 pages of the core memory (the "real" pages) were being used by active programs. All major hardware items are in operation, including two central processors (P1 and P2), the eight magnetic core memory boxes (storage A,B,C,D,E,F,G and H) and the two channel control units (CCU's 0 and 1).

E. RUNNING USER-PRODUCED PROGRAMS

In addition to the public files discussed above and ADAAS which is the subject of this manual, there are a large number of programs that are produced and maintained by system users themselves for utilization by the entire computing community. These files are usually maintained in a read-only status and are useful in performing a wide variety of processing tasks. Because the programs are user-maintained, the responsibility for program operation and errors must be with the person who runs these programs although the originator in most cases is happy to be notified of any errors or difficulties. Please contact HSRI for

details on running these programs.

An extremely useful statistical analysis program is MIDAS, maintained by the University of Michigan Statistical Research Laboratory. This user-oriented, interactive program may be used to compute histograms, correlations, etc., and a wide variety of data handling and transformation tasks. While the ADAAS/SRS package is well suited to the analysis of large data sets, MIDAS has a great many advantages when small data sets are involved. Interested users are encouraged to contact HSRI for further information on this analysis package.

Additionally, HSRI maintains an internal system of programs for the secondary processing of data generated by the SR System or by other data sources. Of prime interest are the plotting programs to provide graphical representations of the data. Most important are a 3D plotter program to provide graphical output of bivariate frequency information (i.e., a histogram with two independent variables) and a mapping program to produce grey level maps of the United States by state, or of the State of Michigan by county. The 3D plotter produces CALCOMP graphical output, while the mapping programs output directly to the terminal device or line printer. An extremely useful histogram program is available under the name BARGRAPH. This program will produce histograms in a wide variety of formats from tabular input data.

F. ABBREVIATED MTS COMMANDS

When operating from a terminal in conversational mode on MTS, it is often possible to omit the dollar sign before the command and to use a shortened form of the command. For example:

```
#C FILEFACTS
```

is equivalent to

```
#$COPY FILEFACTS
```

The savings in user interaction time is obvious. In batch mode,

the full-length form of the command (with the dollar sign) should always be used.

In some cases the command may be shortened, but the dollar sign may not be omitted. Thus, entering the command U (for UNNUMBER) while the user is in the automatic numbering mode would cause the character U to be entered on the appropriate file line. The command "$U" is recognized as an MTS command to turn off numbering.

The acceptable short form of the MTS commands discussed in this section are indicated in the MTS command summary (presented in Section K) by underlining.

G. DEVICE CONTROL COMMANDS

In addition to the MTS commands that are prefixed by a dollar sign, there are a number of useful Device Control Commands that are prefixed by a percent sign (%) on the Memorex 1270 (CTRL-A, CTRL-A or !A!A on the Data Concentrator). These commands do not instruct MTS, but simply tell the terminal control device how to handle the information that is passed between the terminal and MTS.

The most frequent use of these commands is in regard to line length. A "line" of information in MTS may be up to 255 characters long while output data lines from the SRS are 132 characters long. Line printers and some terminals (IBM 2741, Model 38 Teletype, etc.) can handle 132 characters, but most teletype-like terminals will only handle 70-80 characters.

If a file with a line length longer than a given device will handle is output to a terminal, the device controller will simply output the available number of characters and throw the rest of the line away. For example, a bivariate table from the SRS that is copied to a Model 33 or Model 35 Teletype will simply be truncated and only the left portion of the table will be obtained.

MTS can be forced to print the entire 255 character line by the following command sequence.

EXAMPLE 25:
```
#%LEN=255
%OK.
#
```

Any desired length can be substituted for 255 of course. This procedure will result in the MTS "line" being output as a number of terminal lines. The logical form of the output is consequently modified, but the information is saved.

The right margin can be similarly modified by the %RMAR command. For example, the 132 character IBM 2741 could be made to appear like a 72 character device with the command sequence:

EXAMPLE 26:
```
#%RMAR=72
%OK.
#
```

The RMAR command can be used in combination with LEN to control the form of the output.

Finally, there is a device control command to prevent the computer from disconnecting the telephone line when a $SIGNOFF command is given. The sequence

EXAMPLE 27:
```
#%DON'T
%OK.
#
```

will accomplish this. After a signoff, the computer will resubmit the signon information to the terminal. This command is very useful when a single user wishes to consecutively sign on to different user ID's or when there is a queue of people waiting to utilize a single terminal.

H. BATCH MODE OPERATION

While the emphasis of this chapter has been on terminal operation, the form of the input commands to MTS are the same in both batch and terminal modes. The main difference is that errors in batch do not result in error comments and the opportunity to enter

new information: the task is simply terminated when an unrecoverable error is encountered.

The form of the signon command used in Section 2-B for terminal signon assumes a number of default options relative to the command. In batch mode the default options are somewhat different and it is not usually possible to ignore them. The general form of the signon command is

```
$SIGNON MYID PARLIST 'COMMENT'
```

The first two parts of the command are the signon command itself and the user ID. The comment phrase (enclosed in primes) is simply an identifying string that is used for run identification purposes. The user's name is a commonly used comment phrase. The PARLIST component is a set of keywords and keyword values such as that encountered in the $SET command (i.e., PW=NEWPASS). See Rule 4 for the formation of this PARLIST. Valid keywords and their default values for batch operation are shown in Table 3. Remember that the default value is obtained when the keyword phrase is simply omitted.

Example 28 shows a set of batch commands to copy a large file to the printer.

EXAMPLE 28:

```
$SIGNON MYID PAGES=100 'K.OMENT'
PASSWORD
$COPY LARGEFILE
$SIGNOFF
```

Similarly a batch job to run the program stored on the file EXPENSIVE at midnight (when the computer hopefully won't be too busy) is shown in Example 29.

EXAMPLE 29:

```
$SIGNON MYID T=5M PRIO=L 'I.M.RICH'
PASSWORD
$CREATE OUTPUT
$RUN EXPENSIVE PAR=OUTPUT
CARD 1  }
CARD 2  }      input data cards for EXPENSIVE
CARD 3  }
$ENDFILE
$COPY OUTPUT
$SIGNOFF
```

TABLE 3

Keyword Parameters for the $SIGNON Command

PARAMETER	BATCH DEFAULT	Suggested Values[2] SMALL JOBS	LARGE JOBS	
TIME = {t|tS|tM}[1]	30S	2M	5M	Maximum processing time to be used ($5/minute 10-01-73) t,tS-time in seconds (100,100S) tM-time in minutes (5M)
PAGES = p	50 pages	100	200	Maximum number of pages to be printed.
CARDS = c	0 cards	Default[3]	Default[3]	Maximum number of cards to be punched.
PRIO = {N|L}[1]	N	Default[3]	L	N=normal priority-job run by priority level. L=low priority-job run near midnight.
COPIES = n	1	Default[3]	Default[3]	Number of copies of output to be printed.

[1] {a|b|c|d} indicates that a choice may be made between the permissible options a,b,c, & d.

[2] These are only suggested values based on experience. The correct choice of values is, of course, the responsibility of the user.

[3] To use the default option, simply omit the keyword parameter from the signon command.

[4] CNTR is the route code for the Computing Center, Ann Arbor, Michigan.

Two methods may be used to enter a batch job to MTS. The first, and most historic, method is to punch the desired commands on cards and to submit them at a batch station. The printed or punched output will then be returned to the originating station (unless an alternate route was specified).

Unfortunately, most remote users do not have a batch station at their disposal. However, batch jobs can still be run subject only to the restriction that no batch printer is available for the job. This means that any printed output from the job will be routed to the Computing Center in Ann Arbor, Michigan (see Note (4), Table 3). If the batch job is designed so that no important printed output is generated, however, this restriction need not be serious. In Example 29 the output of the program EXPENSIVE was put in the permanent file OUTPUT and copied to the printer for convenience. Alternatively, the $COPY OUTPUT command could be deleted; after the job is run in batch, the user may sign on in conversational mode and copy OUTPUT to his terminal. A final example of this technique is shown in Example 30.

EXAMPLE 30:
```
$SIGNON MYID
PASSWORD
$COPY FILEA TO FILEB
$SIGNOFF
```

Here FILEA is copied to FILEB and no printed output is necessary if the job is set up correctly.

To enter a batch job from the terminal, prepare a file whose lines contain the batch job commands, one command per line, in the same order as the desired command sequence. In preparing the file by the techniques of Section 2-C, it is important to remember Rule 8. When the file (called BATCH below) is prepared, the batch job is initiated as in Example 31.

EXAMPLE 31:
```
#$COPY BATCH *BATCH*
>*BATCH* ASSIGNED RECEIPT NUMBER NNNNNN
#*BATCH* NNNNNN RELEASED
#
```

The number NNNNNN designates the job number, which identifies the output at the specified batch output station. To obtain the status of the job, use the command:

```
$RUN *HBQ PAR=NNNNNN
```

I. LIMITED SERVICE STATE

When computer loading is very heavy, users may be placed in a LIMITED SERVICE STATE (LSS) when they sign on to the system. This means that it is not possible to use the five commands:

$DEBUG $LOAD $RESTART $RUN $START

Of these five, only the $RUN command is of importance to users of this manual. Since it is necessary to use the $RUN command with the ADAAS system, significant operations cannot be performed in a limited service state.

Two procedures are possible to circumvent this difficulty. First, the user may attempt to get out of the limited service state by issuing the command:

```
$SET LSS=OFF
```

If the load is sufficiently light, and LSS can be turned off, MTS will simply respond with the pound (#) sign. If this happens, the $RUN command may be issued directly. If the load is not sufficiently light MTS will respond with:

```
CAN'T SET LSS OFF - LOAD TOO HIGH
```

If repetitive use of the LSS=OFF operation does not work, it is probably more time and cost effective to submit the task as a batch job. All the necessary setup operations for preparing and initiating a batch job may be performed even when LSS=ON.

Please remember that MTS is not a commercial time-sharing

system operating at a profit that can expand its facilities (within its income) to meet any needs. MTS is an educational/ research system that gets heavily loaded periodically when large numbers of students are solving course-assigned problems against a time deadline.

J. ATTENTION INTERRUPT PROCESSING

In conversational mode it is possible to issue an ATTENTION INTERRUPT that provides a means of requesting MTS to stop its current processing task and, possibly, to proceed with some other action. The means for issuing this interrupt is discussed in Section 2-A. This operation is very useful for stopping programs when a mistake has been made or when the output being generated is no longer desired.

The action which follows an attention interrupt depends on what was happening when the interrupt was issued. Three different possibilities are discussed below:

1. If no program is currently running, and certain MTS commands are in the process of execution ($LIST, $COPY, $FILESTATUS etc.) the current operation will be suspended, the message ATTN! will be issued, and MTS will return with a pound (#) sign signifying that a new command is expected.

2. In other cases, if a program is currently running in MTS that does not do its own attention interrupt processing, an attention interrupt results in the message:

ATTENTION INTERRUPT AT NNNNNNNN

and a return to the MTS command mode signified by a pound (#) sign. The number (NNNNNNNN) represents the point in the program at which the interrupt occurs. In this situation, the program is not unloaded from the computer memory so that operation may restarted, if desired, by the command $RESTART.

3. In yet other situations, certain MTS commands ($SIGNOFF

for example) and some programs do their own internal attention interrupt processing. In some cases it is vital that the operation not be suspended until certain critical tasks have been completed. In situations such as these, the BREAK or ATTENTION key is simply disabled. In the $SIGNOFF command, for example, all the bookkeeping for the run has to be updated and attention interrupts must not be allowed to curtail this. In other programs the attention interrupt signal is used to signify that the user no longer wants to continue his current operation and should be rerouted to a more desirable operation. In the HSRI ADAAS system an attention interrupt will result in the message:

+ATTN+

and the user will be routed to a different point in the ADAAS program.

K. MTS COMMAND SUMMARY

	Phone Lines (Area Code 313)
763-0300	Memorex 1270
763-1500	Data Concentrator
763-0420	MTS Operational Status
763-0428	HSRI Systems Analysis
	MTS General
$SIGNON MYID	To identify a user to the system
$SIGNOFF $SIGNOFF SHORT $SIGNOFF $	To notify the system of a user's departure.
$SET PW=PASSWORD	To change the user password.
$RUN FILENAME	To load and initiate execution of a program stored in the file "FILENAME" (e.g., *STATUS, *USERS).

MTS File Handling

$CREATE FILE	To create a permanent or temporary file.
$COPY FILE $COPY FILE1 TO FILE2	To copy from a file to a terminal or another file without line numbers.
$DESTROY FILE	To destroy a permanent or temporary file.
$EMPTY FILE	To empty the contents of a file without changing its size or destroying it.
$FILESTATUS	To obtain information on the files stored on the user's ID.
$GET FILE	To establish a file as the active file.
$LIST FILE	To list a file on the terminal with line numbers.
$NUMBER	To start automatic numbering of input lines being read.
$RELEASE	To release the active file or device.
$UNNUMBER	To turn off automatic numbering.

Device Control

%DON'T	To prevent MTS from hanging up at signoff.
%LEN=NUM	To change the length of the output line at NUM characters.
%RMAR=NUM	To change the right margin to NUM characters.

SECTION 3

THE STATISTICAL RESEARCH SYSTEM (SRS)

The Statistical Research System (SRS) is a package of data handling and processing programs to permit the analysis of large quantities of information in a timely fashion. SRS is resident on the University of Michigan IBM 360/67 computer and may be operated in either batch or conversational mode under MTS as stand-alone-programs in addition to their use in ADAAS.

The system is a group of application programs and a subroutine library. Each program is accessed by a unique reference number. The SRS was designed to process magnetic tape or direct access files and consequently permits the analysis of large data sets. Control operations necessary to handle the magnetic tape (tape rewinding, positioning, etc.) are performed by the programs and are transparent to the user.

A. SRS PROGRAM DESCRIPTIONS

A complete listing of the SRS analysis programs is given in Table 4. For the purposes of this manual, we will only be concerned with those programs that are followed by an asterisk. Users are encouraged to utilize the full capabilities of the system when they feel they have the experience to do so; more information can be obtained by calling HSRI.

Five programs have been chosen for ADAAS to satisfy most user needs. The ADAAS programs are listed in Table 5. A brief description of these programs and the results they produce is presented below.

Data Set Listing

This program is used to selectively retrieve and to list a subset of cases and variables from the original file. This program is not an analysis program in the strict sense, and is useful only when the detailed features of a small number of cases must be carefully perused. For instance, if only 10 to 20 cases

TABLE 4

Catalog of Analysis Programs in the Statistical Research System

1. DATA SET LISTING*
2. BARGRAPH*
3. UNIVARIATE FREQUENCIES*
4. ONE-WAY ANALYSIS OF VARIANCE*
5. BIVARIATE FREQUENCIES WITH TWO-DIGIT CODES AND FOUR FILTERS*
6. BIVARIATE FREQUENCIES
7. BIVARIATE FREQUENCIES WITH TWO-DIGIT CODES
8. MISSING DATA CORRELATIONS
9. PARTIAL CORRELATIONS
10. LINEAR REGRESSION
11. MULTIPLE ANALYSIS OF VARIANCE
12. AUTOMATIC INTERACTION DETECTOR
13. MULTIPLE CLASSIFICATION ANALYSIS
14. FACTOR ANALYSIS

*ADAAS Programs

TABLE 5

Catalog of ADAAS Analysis Programs

1. DATA SET LISTING

   Selects and prints a subset of the data file.

2. BARGRAPH

   Prints a bargraph showing the univariate frequency distribution of one variable.

3. UNIVARIATE FREQUENCIES

   Statistics: Provides case counts, sum of weights, ranges, means, standard deviations, skewness and kurtosis.

   Frequencies: Provides one-way frequency distributions (univariates) of individual variables and percentages on the distributions.

4. ONE-WAY ANALYSIS OF VARIANCE

   Produces one-way analysis of variance tables.

5. BIVARIATE FREQUENCIES

   Provides bivariate (two-way) frequency tables for pairs of variables.

having certain desired characteristics can be found in a given file (i.e., accidents involving a school bus, fire, etc.), then any statistical analysis of the information is probably not meaningful. Such situations can be handled by listing a number of variables of interest for these cases. The resultant listings can be examined manually in detail to see if any significant features are discernible. In other cases, it may be desirable to determine the accident identification numbers for a subset of cases in order to obtain information from the original reports themselves: this may be easily done with a Data Set List. Appendix C describes two special purpose programs for listing the alphabetic translations of numeric code values in the CPIR Revision 3 data files.

Bargraph

One of the most meaningful quantities describing the nature of a given variable in an accident file is the distribution function for that variable. The distribution function describes the percentage of cases in the file that are characterized by each possible value of the variable in question. For example, if our variables describes the side of a coin that lands upwards after being thrown in the air (possible values, 1 = heads, 2 = tails), then the distribution function for a large number of tosses would show a value near 50% for both possible variable values. Other forms of the distribution function simply show the case count for each value of the variable and are not normalized to indicate percentages.

In the Bargraph program, the possible values of the variable being analyzed are grouped into unique "class intervals" that define a range of values on the horizontal axis of the bargraph. The vertical axis represents the total number of cases that occur for each interval of the variable and is divided into units that reflect the approximate number of cases within the given classes.

Univariate Frequencies

This program is used to provide a number of summary statistics characterizing the distribution function for a user-supplied list of variables. This summary includes case counts, sum of weights for weighted data, number of missing data cases for one or two missing data codes, range of variable values, means, standard deviations, skewness, and kurtosis. In addition, the program accumulates univariate frequency distributions for each variable and optionally computes percentage distributions.

If statistics are requested, the following quantities are printed using one line for each variable, 15 variables per page:

1. The variable number
2. The variable name
3. The case count for the variable (excluding missing data).
4. The mean
5. The standard deviation
6. The minimum code value (can be negative).
7. The maximum code value (can be negative).
8. The case count for the first missing data code.
9. The case count for the second missing data code (all code values = the missing code value specified).
10. Skewness (3rd moment).
11. Kurtosis (4th moment).
12. Sum of weights (if the data are weighted, the statistics are of the weighted form and the sum of weights is also printed).

If frequencies are requested, the program will accumulate univariate distributions for each variable. The missing data codes are included in the distribution range. These distributions may be weighted. Distribution percentages can optionally be computed and are printed out below the frequency for each code value. For each variable, starting with the lowest

code, up to 15 code values are printed per line with the frequencies (and percentages) underneath the respective code value. Each consecutive code value, with a non-zero frequency, is printed out until the maximum code value is reached.

One-Way Analysis of Variance

Analysis of Variance is a statistical technique to test the equality of more than two population means, thereby determining if the variations that occur are significant (in a statistical sense) or due to the action of chance alone. For example, a tire company may want to test the lifetimes of three or more different types of tires and determine from the average lifetime for each type if there are significant differences in tire durability.

For each value of the control variable specified, the program evaluates:

1. The control variable code value (CODES).
2. The case count for this code value (N).
3. The sum of weights (WEIGHT-SUM).
4. The percentage distribution function for 3 (%).
5. The mean value of the weighted dependent variable (MEAN).
6. The standard deviation of the weighted dependent variable (S.D. (ESTIM)).
7. The sum of weighted case values (SUM OF X) (Equal to N*MEAN).
8. The percentage distribution function for 7 (%).
9. The sum of squared, weighted case values (SUM OF X-SQUARED).

At the bottom of the data output page, a number of quantities are produced relative to the statistical significance of the variations encountered. These quantities are discussed in a more complete treatment of ANOVA presented in Appendix B.

The analysis of variation technique is very useful in data analysis operations. For instance, suppose we would like to

evaluate the average number of fatalities per accident as a function of weather (e.g., clear or cloudy, raining, snowing, etc.). Running ANOVA on an accident file with Weather as the control variable and Number of Fatalities as the independent variable would compute the desired averages (e.g., average number of rainy weather fatalities).

Bivariate Frequencies

In the Bargraph program, the concept of a distribution function was introduced. For a single control variable this quantity shows the frequency with which the possible values are represented in the accident population. This concept can be easily extended to two or more dimensions. In a general case, the distribution function for N variables indicates the relative occurrence of cases that are characterized by each possible combination of values for the N control variables.

If N = 2, the distribution is referred to as bivariate. As an example, we might consider the distribution of accident cases as a function of driver sex and of light condition (day, night, or dusk). A bivariate frequency table would then have the form:

	Male	Female
day	N_{11}	N_{12}
dusk	N_{21}	N_{22}
night	N_{31}	N_{32}

where the N_{ij} corresponds to the case counts for the particular combination of parameters. That is, N_{22} represents the number of accidents involving a female driver at dusk, etc. The bivariate frequency concept is thus a logical extension of the histogram or one-way distribution concept and represents a

more sophisticated breakdown of the factors involved.

The basic program output is thus bivariate frequency tables that represent the case counts obtained for chosen values of the row and column variables. In addition, one-way percentage distributions that are based on either row or column totals can also be obtained, as well as two-way percentage distributions based on grand totals. Missing data codes may be included or excluded from the bivariate tables.

B. SRS FILE STRUCTURE

Due to the large volumes of data involved, most HSRI accident files are stored on magnetic tape. In a few cases where usage is high or volume of records small, the files are stored on disk. Control operations necessary to handle the tape and to read data are performed by the SRS so that users have no need to know the detailed structure of the files. The conceptual structure must be understood for successful use of the accident analysis system, however.

An SRS file can be visualized as a two-dimensional collection of information items where the dimensions are referred to by the names "CASES" and "VARIABLES". Figure 2 graphically shows a simple accident file described by five cases and five variables. This example will be used below to illustrate important file characteristics.

Each entry in the file is called a "CASE". In SRS data files, a CASE may refer to a variety of events, physical objects, or actions. In the most common situation, referred to as an ACCIDENT FILE, each accident is a case entry into the file. More often than not, of course, more than one vehicle is involved in an accident. It is possible to create a VEHICLE FILE by making each vehicle involved in an accident correspond to one case; descriptive information concerning the accident

Sample File

	V1 Driver Sex	V2 Driver Age	V3 Accident Severity	V4 Object Struck R1	R2	R3	V5 Vehicle Make
Case 1	2	18	07	03	12	99	PLY
Case 2	1	17	03	03	05	08	FOR
Case 3	2	52	00	05	99	99	CHE
Case 4	2	33	01	12	99	99	PON
Case 5	1	99	01	06	03	99	VOL

Codebook

V1 - DRIVER SEX

Numeric (categorical)
Number of Responses = 1
Field Width = 1
Missing Data Code 1 = 9
Missing Data Code 2 = 3
Code Values:

1 = Male
2 = Female
9 = Missing Data

V2 - DRIVER AGE

Numeric (interval)
Number of Responses = 1
Field Width = 2
Missing Data Code 1 = 99
Missing Data Code 2=NONE
Code Values:

01 = 1 Year old
.....
97 = 97 Years old
98 = 98 or older
99 = Missing Data

V3 - ACCIDENT SEVERITY

Numeric (ordinal)
Number of Responses = 1
Field Width = 2
Missing Data Code 1 = 98
Missing Data Code 2=NONE
Code Values:

00 None
01 Minor
02 Non-dangerous, Moderate
03 Non-dangerous, Severe
07 Fatal Lesions in 1 Region

V4 - OBJECT STRUCK

Numeric (categorical)
Number of Responses = 3
Field Width = 2
Missing Data Code 1 = 99
Missing Data Code 2=NONE
Code Values:

03 Other automobile
05 Guardrail
06 Bridge (rail)
08 Ditch
12 Pole or Tree
99 Other:

V5 - VEHICLE MAKE

Alphabetic
Number of Responses = 1
Field Width = 3
Missing Data Code 1=NONE
Missing Data Code 2=NONE
Code Values:

CHE Chevrolet
FOR Ford
PLY Plymouth
PON Pontiac
VOL Volkswagen

FIGURE 2 Sample File and Codebook

scene is repeated for each vehicle in a given accident. Again, each vehicle may contain several occupants and each occupant may be considered as a case (the OCCUPANT FILE). Finally, at the highest level of division usually found, each occupant (in each vehicle in each accident) may have several injuries. Using each separate injury as a case leads to the INJURY FILE concept.

Each specific item of information that is coded for every case is called a "VARIABLE". Thus driver age, vehicle make, time of day, or seat belt usage are all variables that might be used to describe cases. The number of variables used to describe a particular case is usually on the order of one to two hundred, but may go as high as 600-700 in the MDAI files where a large amount of detailed information on vehicle damage is available.

Variables are characterized by a number of parameters that describe how the information is stored. These are:

1. Type. That is:
   a. Alphabetic
   b. Numeric (Interval, Ordinal, or Categorical)
2. Field Width
3. Number of Responses
4. Data Code Values
5. Missing Data Codes
6. Frequency

Each of these characteristics will be discussed below in detail.

1. Type

Information for each variable may be entered as a set of alphanumeric characters by using an alphabetic variable, or as a set of numbers by using a numeric variable. Numeric types are most useful for digital computers where well-defined arithmetic operations can be handled easily and quickly. In the

SRS, analysis operations, other than simply listing the data, can be performed only on numeric variables or on alphabetic variables that have been transformed by the RECODE option (see Section 3-D) to temporarily contain numeric values. Three classes of data may be assigned to a numeric variable: interval, ordinal or categorical. The user should be familiar with these distinctions since interpretation of the analysis results that are obtained depends strongly on the data class.

Interval data involve the coding of arithmetic information. Thus in the age variable V2 (Figure 2), there is a direct correspondence between driver age and code value (code value 52 for Variable 2 represents a person whose age is 52 years). By performing arithmetic operations on the code values (summing, averaging) we obtain quantities that are meaningful in a physical sense.

Ordinal data deal with information that can be ordered--as numbers can--but cannot be assigned a specific value that is meaningful. For instance, a typical accident severity variable might have the code values:

00 = None
01 = Minor
02 = Non-Dangerous, Moderate
etc.

as shown in Variable 3 in Figure 2. Clearly, code 2 is more severe than code 1 which in turn is worse than code 0 (in some sense). The code values are therefore ordered in some severity scale. However, we are not saying that non-dangerous injuries are twice as severe as minor injuries. Similarly, if we found that the average accident severity was 1.6 for a certain class of accidents, we would know that a typical accident is somewhat worse than one involving just minor injuries. It is not meaningful to say, however, that the typical accident is 1.6 times as severe as a minor injury event.

Finally, categorical data are information that cannot be

ordered. For instance, an "OBJECT STRUCK" variable (V4 of Figure 2) might contain a correspondence between code values and a listing of objects struck by the vehicle; that is:

03 = Other Automobile
08 = Ditch

etc.

For this set of code values, there is simply no direct numerical relationship among data values. The "average" object struck for a certain class of accidents is similarly meaningless. This type of information is descriptive and cannot be compared numerically.

Because alphabetic variables cannot be used in analysis programs (except via RECODE), they are used mainly for assigning user readable names to data elements. Thus, vehicle make may be coded as Variable 5 in Figure 2 with the suggestive three letter values:

CHE Chevrolet
FOR Ford
PLY Plymouth

etc.

For ordering purposes, the alphanumeric information contained in alphabetic variables is assumed to be in the following collating sequence:

Special Characters → Lower Case Letters → Upper Case Letters → Numbers

Thus the alphanumeric values (+; $a, AA, 59) are sorted in ascending order.

2. Field Width

The number of different data values corresponding to a single variable that can be recorded in a given file are dependent on a quantity known as the Field Width of the variable. In the SRS, data are stored as characters in a compatible format (IBM EBCDIC); field width is then simply the number of

EBCDIC characters needed to store the variable values. It is possible to use all the alphanumeric characters and store nearly 100 code values with a field width of one. In practice, however, the use of special characters is not desirable. HSRI consequently strives to use only numbers 0-9 for numeric variables; with this convention the field width is equal to the number of decimal digits in a number (i.e., a field width of two can store values from zero to 99; etc.).

3. Number of Responses

The majority of accident data variables are denoted as single response. That is, each case entry in the file is allowed only one coded value for a specific variable. For example, a variable documenting lap belt usage might have two possible code values to record whether the belt was worn or not; however, for a given case only one of these values could be entered in the data file.

The SRS provides for the use of variables that are denoted as multiple response. Such variables allow more than one valid coded entry to be recorded for a single case. Consider a variable used to record the object struck by the case vehicle in an accident. If this variable were single response, then only one object could be coded: this might be the first object struck, the object that caused major damage to the vehicle, or any other object that fulfills some criteria related to the accident. The use of a multiple response variable (or MRV), however, would permit the investigator to record a variety of objects struck. For instance, a simple list of struck objects could be entered in any order, or a chronological ordering of struck objects could be used for successive responses of the variable. The use of MRV's in accident files therefore permits additional information to be incorporated into the file without the complicating necessity of using multiple variables. This technique has many advantages in data analysis procedures.

4. Data Code Values

As discussed earlier, each variable in the file describes some aspect of the accident in terms of a set of alphabetic or numeric "CODE VALUES". That is, the given accident scene is transformed (usually non-uniquely) into a set of code values that provides a representation of the physical event. Reconstruction of the accident requires a knowledge of the code values used; i.e.,

1 Male
2 Female
etc.

5. Missing Data Codes

In any field investigation, desired items of information are often unavailable or are not applicable. In other cases, it is desirable to prevent contamination of the output results by WILD CODE VALUES that may enter the data due to miscoding, keypunching, or data processing errors. These wild codes are code values that do not correspond to any defined value for the variable. In the SRS both types of problems are handled by two missing data codes that apply to each variable.

The first missing data code (MD1) is the code value applied to a variable to signify that data for that variable is missing or not available. In most of the analysis programs, this data is excluded from the computations that are performed. For instance, if the mean ages of drivers were being computed, it would be erroneous to include a significant amount of missing data where age is coded as 99.

A second missing data code (MD2) specifies a value greater than any permissible code value for the variable. Thus if month of the year were represented by code values in the range (1-12), a value of 13 would be a good choice for MD2. In the ANOVA program, any code value that is greater than or equal to MD2 is excluded from the calculations.

6. Frequency

The number of times that each specific code value of a variable occurs in a file is referred to as the "FREQUENCY" of that code value. This includes the documentation of missing data codes.

Because the frequency of a given variable documents the occurrence of certain aspects of an accident (without regard to other factors) it contains a great deal of useful information about the accident profile that can be used in analysis operations.

All of the information above that characterizes a variable is given in the CODEBOOK corresponding to each accident file. Figure 3 shows entries from typical HSRI codebooks and graphically illustrates the items of information that are available from these documents.

C. FILTER STATEMENTS

In many cases the accident files contain much more information than is necessary for the analysis task to be performed. The use of data filters permits the inclusion of data that are pertinent or the exclusion of data that are not pertinent. Thus, filters are used on a data file to subset or to retrieve just those cases of interest for analysis.

A filter statement is a phrase that may be up to three lines, each up to 80 characters in length, that begins with the word "INCLUDE" or "EXCLUDE", and that ends with an asterisk (*). In the statement, variables are denoted by the letter "V" and the variable number (i.e., V23). Code values are expressed as unique values (i.e., V45='CHEV'); as unique values separated by commas (i.e., V45='CHEV','FORD'); as a range of consecutive values whose limits are separated by a hyphen (i.e., V23=10-15); or as a combination of these conventions (i.e., V23=01,02,04,12-24). Note that if the field width of the variable is greater than the number of digits in the code

```
------------
Variable  87   INJURED #4 SEVERITY          M.D.Codes:        9,    None
------------   ------------------------    Field Width:  1,    Numeric

  FREQ.    INJURED #4 SEVERITY

     6       1. FATAL
    66       2. "A" (NSC DEFINITION) INJURY
   180       3. "B" (NSC DEFINITION) INJURY
   232       4. "C" (NSC DEFINITION) INJURY
   271       5. PRESENT BUT NOT INJURED
 56777       9. NOT PRESENT IN THIS LOCATION
```

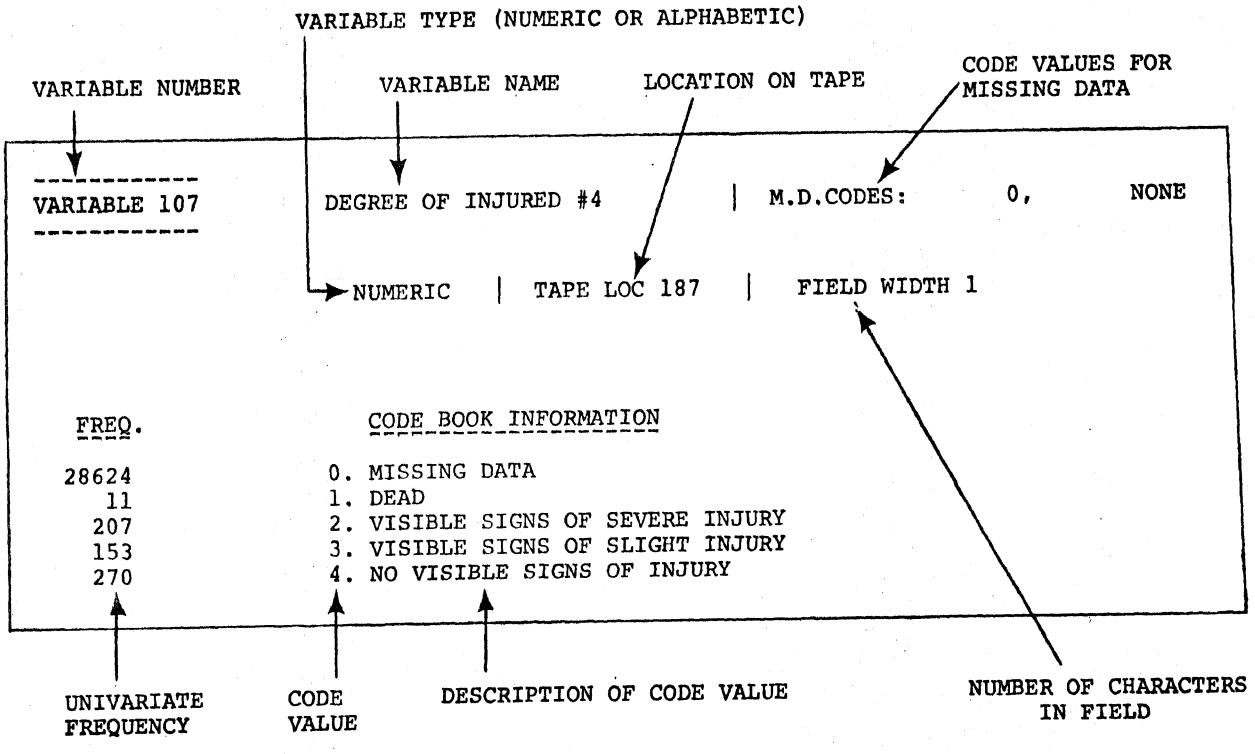

FIGURE 3. Typical HSRI Accident File Codebook Entries

value, the extra digits must be padded with zeros. Note also that alphabetic code values must be enclosed in primes.

The variable statements are combined with "AND" and/or "OR" to provide the desired filter action. The "AND" statement implies both variables listed must have the selected values; "OR" implies either may have the selected value. In addition, the "AND" relationship has precedence over the "OR". Up to 14 "AND"/"OR" connectives may be used in a filter statement. A summary of the filter command statements format for the SRS is given in Table 6.

When using a multiple response variable (MRV) in an SRS global filter, a case will pass the filter (that is, be included or excluded) if any response satisfies the filter condition. This means that some uses of filter statements for single response variables are inappropriate for MRVs so that careful consideration must be given to the exact filter action. Some examples of filter action are shown below for a two response variable (denoted by variable number VN).

GLOBAL FILTER	CASE VALUES FOR VN		RESULT
INCLUDE VN=5,7*	1	8	Excluded
INCLUDE VN=5,7*	1	5	Included
INCLUDE VN=5,7*	7	3	Included
EXCLUDE VN=1-3,5-9*	4	2	Excluded
EXCLUDE VN=1-3,5-9*	8	4	Excluded
EXCLUDE VN=1-3,5-9*	4	4	Included

Note that the discussion above applies exclusively to global filters; that is filters that apply to the entire data set used for analysis. These global filter statements are entered in response to the FILTER(RECODE OR LABEL) request. Certain programs have a provision for local filters. It is important to note that MRVs may not be used in local filters.

TABLE 6
SRS Filter Statement Summary

INCLUDE	Includes only the cases that are specified.
EXCLUDE	Includes all cases but the ones specified.
V	Prefix for variable number (i.e., V23).
C1,C2, ···,	C3,C4-C5,C6-C7, ··· C8-C9 Format for acceptable code values.
AND	Includes or Excludes only those cases for which both conditions are true.
OR	Includes or Excludes only those cases for which either condition is true.
*	Terminates a filter statement.

NOTES

1) AND has precedence over OR

2) The code values specified in the filter statement must have the same field width as the variable they describe (check the codebook).

STANDARD FORM

INCLUDE / EXCLUDE VN1=code value list 1 AND / OR VN2=code value list 2

AND / OR ... AND / OR VNn=code value list n*

where:

N1,N2,···,Nn are n variable number used in the filter

(N ≤ 15)

In order to clarify the filtering concept, a number of examples are presented below:

EXAMPLES:

EXCLUDE V16=5 AND V14=1, 7-9*

This excludes those cases for which both variable 16 is coded 5 and variable 14 is coded 1,7,8, or 9.

INCLUDE V66=05,22 AND V52=004*

This includes only those cases for which variable 66 (a two-digit variable) has a code of 5 or 22 and variable 52 (a three-digit variable) has a code of 4.

INCLUDE V16=5 OR V14=9*

This includes only those cases for which either variable 16 is coded 5 or variable 14 is coded 9 or both.

INCLUDE V16=5 AND V14=9 or V16=6*

This includes only those cases for which either both V16 is coded 5 and V14 is coded 9, or V16 is coded 6.

INCLUDE V16=5 AND V14=9 AND V66=05 OR V16=5 AND V14=9 AND V52=004*

If the user desires those cases where V16=5 and V14=9 and either V66=05 or V52=004, enter the above filter. Remember that AND takes precedence over OR so "INCLUDE V16=5 AND V14=9 AND V66=05 OR V52=004*" would not select the cases desired, e.g., all cases where V52=004 would be included.

INCLUDE V54='RED ','BLUE' AND V129='FORD'*

This includes only those cases where V54 is coded 'RED ' or 'BLUE' and V129 is coded 'FORD'.

EXCLUDE V78='CHEU' - 'CHEZ'*

This excludes all cases where V78 is coded 'CHEU', 'CHEV', 'CHEW', 'CHEX', 'CHEY', or 'CHEZ'.

D. RECODE STATEMENTS

Many analysis tasks are facilitated by the recoding of variable values: that is, the substitution of new code values for the permanent ones that exist in the accident file without permanently creating a new variable. For instance, a variable that specifies driver age by years might be recoded by the correspondence:

00-09 → 01
10-19 → 02
etc.

In other cases, recoding provides a means of circumventing certain program restrictions. For example, the bivariate program only prints up to 12 consecutive code values for the column variable. Portions of a variable with a field width of two can be used in a bivariate by recoding the desired portion into the range of 0 to 9. A final very important use of the recode option is the conversion of alphabetic code values to numeric values in order to permit the use of analysis programs on otherwise unusable variables.

Note that the recoding is temporary. The analyst does not have to permanently alter the data file by computing and storing a new variable. This option permits a dynamic recoding, i.e., good only during program operations. The modifications are lost when execution is terminated.

Because Data Set List is not an analysis program, the RECODE option is not valid. The RECODE statement may be used with all other ADAAS analysis programs.

Each recode statement must begin with the word "RECODE" and end with an asterisk (*). The recode statement has the generic form

```
RECODE VN (ORIGINAL VALUE LIST 1) = NEW VALUE 1 (ORIGINAL
VALUE LIST 2) = NEW VALUE 2, ..., (ORIGINAL VALUE LIST
n-1) = NEW VALUE n-1, (ELSE) = NEW VALUE n*
```

The term "VN" is the standard designation for variable number N with the prefix letter "V". Each original value list (enclosed in parenthesis) indicates the original accident file code values that are to be given the new value indicated. This list has the same form as the acceptable code values in a filter statement. Thus, commas separate unique values, and a hyphen indicates ranges of values. The "ELSE" operand in the initial value list assigns any unspecified original values to the specific new value and must occur at the end of a statement if it is used. The "ELSE" statement prevents new recoded values from overlapping permanent original values. "ELSE" is helpful in recoding wild or stray original values to an "Unknown" code value, and must be used with alphabetic variables.

The temporary recode option is valid only for the recoding of numerics to numerics and alphabetics to numerics.

Each recode statement specifies the code translation of a single variable, but up to five statements may be used in one program operation; however, there is an overall restriction that no more than ten 80 character lines be used for all recodes. The statements may be entered in any order, but the actual operation will be performed sequentially in the order indicated by their entry. Finally, the field widths of the initial and final code value values must correspond to the field width of the variable as specified in the file codebook (see, for example, Figure 3).

In a strict sense, MRVs cannot be recoded. If RECODE is used, only the first response is recoded; the remaining responses are unaltered by the operation. Consequently, RECODE may be utilized for certain particular operations if this restriction is understood.

A summary of the RECODE statement parameters is presented in Table 7 and a number of practical examples are given below:

TABLE 7
SRS Recode Statement Summary

RECODE	First term in a recode statement.
V	Prefix for variable number (i.e., V23).
C1,C2,C4-C6,...	Form of initial code value list.
(Initial code value list) = final code value	Code value recode specification.
ELSE	Represents all codes not specified explicitly by a recode specification.
,	Separates individual recode specifications.
*	Terminates a recode statement.

NOTES

1) ELSE, if used, must be the last recode specification in a statement.

2) The code values specified as initial or as final values in the recode statement must have the same field width as the variable they describe.

STANDARD FORM

RECODE VN (Original value list 1)=New value 1, (Original value 2)=New value 2,•••,(Original value n)=New value n, (ELSE)=New value*

where:

N is the variable number

n is the number of the last set of values to be recoded

••• represents the intervening records

RECODE EXAMPLES:

RECODE V87 (00-09) = 01, (10-19) = 02, (20-29) = 03, (30-39) = 04, (40-49) = 05, (ELSE) = 06*

This example brackets the values into groups of ten and lumps all values of 50 or greater into a single value.

RECODE V13 ('WHITE') = 00001, ('YELLO') = 00002, ('GOLD') = 00003, (ELSE) = 00009*

This recodes alphabetic color information into numeric values. Note that field width of new values equals original width.

RECODE V33 (0,1,8) = 9*

This converts all 0, 1, or 8 code values to 9 and leaves the remainder unchanged. No "ELSE" is used.

RECODE V17 (00-04) = 01, (05-09) = 02, (10-14) = 03, (15-19) = 04, (20-24) = 05*

Recodes code values below 25 into 5 unit increments.

E. LABEL STATEMENT

Provisions are made on SRS to label the output information. A label of up to 80 characters may be used for this purpose. Any combination of letters and numbers may be used, but the label must not begin with "INCLUDE", "EXCLUDE", or "RECODE". The label may be left blank by entering an end-of-line code (CNTR-S or RETURN) when a label is requested.

The label is very useful where several persons share one computer user number. It is recommended that labels contain a name or initials, date, and subject title for a program run.

F. PARAMETER STATEMENTS

ADAAS programs require at least one parameter card in order to run the program. The parameter card is made up of a set of keyword expressions and modifiers that usually specify what

variables are to be used and what program options are desired.

A keyword expression is made up of a keyword and an equal sign followed by some subject. For example,

```
VAR=10-24,26
```

"VAR" is the keyword (meaning the variable list is) and "10-24,26" is the subject of the keyword (meaning here that Variables 10 through 24 and Variable 26 are to be used in the program execution).

A modifier is a single word usually indicating a program option. For example, "SPACE" in the data set list program will double space the output list.

The parameter card should be typed with at least one blank separating each parameter (i.e., keyword expression or modifier) from the next, and there should be no imbedded blanks within a parameter expression. If there is not enough room to type the needed parameters on one line or card, a dash ("-") can be typed at the end of the line and the remaining parameters can be typed on the next input line. Parameters may be typed in any order.

After a set of parameters is read, they are scanned for spelling errors and invalid characters. If an error is found, the improper parameter is printed. If in batch mode, the parameter card will be ignored or the program will terminate execution. If in conversational mode, the following message will be printed:

```
ENTER REPLACEMENT PARAMETER,
RETURN TO IGNORE, OR "CANCEL".
```

the program user then has the choice of retyping the parameter in error, hitting the "return" button, or typing "cancel". The second choice causes the program to ignore the improper parameter. The final choice will delete the set of parameters just read and prompt the user for a new set.

If "DESC" is typed, a description of the keyword and

modifiers for the program being run will be listed. This listing may be terminated at any point without terminating the program by pressing the "attention" or "break" button.

A typical parameter card for data set list might be as follows:

```
SPACE VAR=1-3,100-102,145 STOP=20
```

This would list Variables 1, 2, 3, 100, 101, 102 and 145 for the first twenty cases that pass the global filter. In addition, the output list would be double-spaced.

SECTION 4

HSRI ACCIDENT FILES

HSRI maintains an extensive library of data that is very pertinent to the investigation of highway accidents. Most of the data are accident data describing hundreds of factors in hundreds of thousands of accidents. Other exposure data are also available however, to describe the background population from which the accident population is derived. There are currently over one hundred files directly accessible to users of ADAAS.

HSRI obtains most of its accident data from cooperating police agencies or from special accident investigation teams sponsored by the National Highway Traffic Safety Administration (NHTSA) or by the Motor Vehicle Manufacturers Association (MVMA). These data differ strongly in the degree to which accident factors are investigated.

Data from police agencies are derived from the local police accident report and generally contain all the data recorded on the applicable accident report. Since there is little uniformity in investigating and coding requirements among the local police jurisdictions, the information obtained from various sources must be compared with great care. Police data are referred to as Level I data: HSRI regularly obtains data of this type from the following sources:

1. Denver County, Colorado (DENVER)
2. Dade County, Florida (MIAMI)
3. King County, Washington (SEATTLE)
4. Oakland County, Michigan
5. Washtenaw County, Michigan
6. An 8 County Area around Buffalo, New York
7. The state of Texas

Each of these jurisdictions (with the exception of Texas) results in approximately 30,000 cases each year for which 200 pieces of

information are recorded. Although Texas has a much larger number of accidents overall, the total accident set is divided into several subsets of manageable size.

In some cases the police investigations are supplemented by the addition of accessory information derived by special investigators. This information may be the addition of several measured factors, or may involve the more complete investigation of vehicle damage. These data are referred to as Level II and are supplied to HSRI by MVMA-sponsored investigation teams.

Finally, there are accident data designated Level III that result from an in-depth investigation of the accident by a number of highly trained accident investigation teams. These include the NHTSA-sponsored Multidisciplinary Accident Investigation (MDAI) teams, MVMA-sponsored teams, and teams sponsored by the Canadian Department of Transportation. All data derived from these sources are recorded on the General Motors Collision Performance and Injury Report Revision 3 data form, plus certain supplementary forms.

A limited number of files are also available to determine the population characteristics of a given year. A file giving the profile of registered drivers in Washtenaw County, Michigan is available as well as the results of an exposure survey conducted by HSRI for NHTSA. Figure 4 lists the current HSRI data banks as of December 31, 1973. A current catalog of available files can be obtained via the $COPY HSRI:FILES command.

HIGHWAY SAFETY RESEARCH INSTITUTE
LIST OF CURRENT FILES
November 30, 1973

FILE NAME	FILE TYPE	SPAD ACCESS	DATA BASE KEYWORD	NUMBER OF CASES	NUMBER OF VARIABLES
Bureau Motor Carrier Safety					
1966 (1/2 Year)	A	No	BMCS-66	24,405	42
1967	A	Nc	BMCS-67	42,604	42
1968	A	No	BMCS-68	46,320	42
1969	A	No	BMCS-69	50,609	42
1966-1969	A	Nc	BMCS	163,938	42
CPIR Revision 2					
Vehicle	V	Yes	CPIR2VEH	716	320
Occupant	O	Yes	CPIR2OCC	1,162	507
CPIR Revision 3					
Vehicle	V	Yes	CPIR3VEH	4,201	576
Occupant	O	Yes	CPIR3OCC	6,885	636
Injury	I	Yes	CPIR3INJ	23,048	647
Veh. Cond. & Maint. Report	V	Nc	VCMR	401	298
Dade Co., Florida					
1969 (1/2 Year)	A	Yes	DADE-69	31,056	83
1970	A	Yes	DADE-70	61,767	83
1971	A	Yes	DADE-71	64,046	84
1972	A	Yes	DADE-72	64,190	84
Denver Co., Colorado					
1969	A	Yes	DENV-69	25,581	234
1970	A	Yes	DENV-70	29,432	217
1971	A	Yes	DENV-71	29,585	217
1972	A	Yes	DENV-72	33,166	217
Indiana Turnpike	A	Nc	INDTNPK	5,744	145
King Co., Washington					
1969 (Seattle Metro Area)	A	Yes	KING-69	28,572	194
1970	A	Yes	KING-70	35,181	236
1971	A	Yes	KING-71	34,720	236
1972	A	Yes	KING-72	35,355	235
Michigan Fatal					
Accident					
1964	A	No	MF64ACC	1,808	24
1965	A	Nc	MF65ACC	1,823	24
1966	A	Nc	MF66ACC	1,940	24
1967	A	No	MF67ACC	1,754	24
1968	A	Nc	MF68ACC	1,987	24
1969	A	No	MF69ACC	2,154	24
1970	A	Nc	MF70ACC	1,863	24
1964-1970	A	No	MFACC1	13,329	24
1971	A	Nc	MF71ACC	1,889	46
1972	A	No	MF72ACC	1,997	46
Vehicle					
1964	V	No	MF64VEH	2,715	43
1965	V	Nc	MF65VEH	2,749	43

FIGURE 4 List of Current Files

FILE NAME	FILE TYPE	SPAD ACCESS	DATA BASE KEYWORD	NUMBER OF CASES	NUMBER OF VARIABLES
Michigan Fatal					
Vehicle					
1966	V	No	MF66VEH	2,946	43
1967	V	Nc	MF67VEH	2,606	43
1968	V	No	MF68VEH	3,057	43
1969	V	No	MF69VEH	3,265	43
1970	V	Nc	MF70VEH	2,815	43
1964-1970	V	No	MFVEH1	20,153	43
1971	V	Nc	MF71VEH	3,287	120
1972	V	No	MF72VEH	3,453	121
Mini-Car	O	Nc	MINICAR	372	118
New York Level I					
1970	A	Yes	CAL1-70	39,992	159
New York Level II					
Accident					
1970(1/4)-1971	A	Yes	NY71ACC	9,081	32
1972	A	Yes	NY72ACC	8,048	32
1973 (1/2 Year)	A	Yes	NY73ACC	3,654	32
Vehicle					
1970(1/4)-1971	V	Yes	NY71VEH	17,533	66
1972	V	Yes	NY72VEH	15,695	66
1973 (1/2 Year)	V	Yes	NY73VEH	7,012	66
Occupant					
1970(1/4)-1971	O	Yes	NY71OCC	24,914	81
1972	O	Yes	NY72OCC	21,817	81
1973 (1/2 Year)	O	Yes	NY73OCC	9,377	81
Oakland Co., Michigan					
1968	A	Yes	OAK-68	25,387	120
1969	A	Yes	OAK-69	29,265	213
1970	A	Yes	OAK-70	29,650	190
1971	A	Yes	OAK-71	29,362	233
1972	A	Yes	OAK-72	34,262	189
Ohio Turnpike					
Accident	A	No	OTNPKACC	6,189	87
Vehicle	V	Nc	OTNPKVEH	8,663	49
Pennslyvania Turnpike	A	No	PENNTNPK	11,492	124
Truck, Bus, Motorcycle, and Pedestrian	V	No	TBMP	212	62
Texas					
Bexar County					
1969 Accident	A	Yes	BEX69ACC	26,673	56
1969 Vehicle	V	Yes	BEX69VEH	45,859	139
1970 Accident	A	Yes	BEX70ACC	27,458	56
1970 Vehicle	V	Yes	BEX70VEH	47,284	139
1971 Accident	A	Yes	BEX71ACC	27,254	56
1971 Vehicle	V	Yes	BEX71VEH	48,359	179
1972 Accident	A	Yes	BEX72ACC	32,329	56
1972 Vehicle	V	Yes	BEX72VEH	57,532	179

FIGURE 4 (Continued)

FILE NAME	FILE TYPE	SPAD ACCESS	DATA BASE KEYWORD	NUMBER OF CASES	NUMBER OF VARIABLES
Texas					
5% State Sample					
1969 Accident	A	No	TXS69ACC	18,837	56
1969 Vehicle	V	No	TXS69VEH	32,224	139
1970 Accident	A	No	TXS70ACC	19,392	56
1970 Vehicle	V	No	TXS70VEH	33,204	139
1971 Accident	A	No	TXS71ACC	19,088	56
1971 Vehicle	V	No	TXS71VEH	33,140	179
1972 Accident	A	No	TXS72ACC	21,000	56
1972 Vehicle	V	No	TXS72VEH	36,505	179
Fatal					
1969 Accident	A	No	TXF69ACC	2,913	56
1969 Vehicle	V	No	TXF69VEH	4,257	139
1970 Accident	A	No	TXF70ACC	2,965	56
1970 Vehicle	V	No	TXF70VEH	4,280	139
1971 Accident	A	No	TXF71ACC	2,993	56
1971 Vehicle	V	No	TXF71VEH	4,896	179
1972 Accident	A	No	TXF72ACC	3,099	56
1972 Vehicle	V	No	TXF72VEH	5,133	179
Truck					
1969 Accident	A	No	TXT69ACC	11,590	56
1969 Vehicle	V	No	TXT69VEH	20,641	139
1970 Accident	A	No	TXT70ACC	10,680	56
1970 Vehicle	V	No	TXT70VEH	19,088	139
1971 Accident	A	No	TXT71ACC	8,172	56
1971 Vehicle	V	No	TXT71VEH	14,467	179
1972 Accident	A	No	TXT72ACC	10,835	56
1972 Vehicle	V	No	TXT72VEH	19,530	179
Washtenaw Co., Michigan					
1969 - 1973 (1/2 Year)	A	Yes	WASH	34,985	185
1973 (1/2 Year)	A	No	WASH-73	4,408	185
Washtenaw Driver Record	D	Yes	WASHDRIV	17,989	48

Legend of File Types
- A Accident
- V Vehicle
- O Occupant
- I Injury
- D Driver Registration
- R Vehicle Registration

FIGURE 4 (Continued)

SECTION 5

OPERATING THE ADAAS ACCESS SYSTEM

Access to the HSRI accident files and SRS analysis programs is provided by the ADAAS (Automated Data Access and Analysis System) data management program. Keyword access to most system files and a selected subset of analysis programs supplies the inexperienced user with a conversational approach to data analysis that bypasses the need to know logical I/O units, data definition statements, and other computer requirements. Experienced users will also find the system useful since the program is self-describing in many aspects and the information that must be remembered to operate the system is greatly minimized.

The program was primarily designed for terminal operation and consequently makes extensive use of descriptions, interactive correction of input data, attention interrupt processing, and other features that make terminal operation convenient. Batch mode operation is also provided for, however, and has many advantages when the computer processing load is heavy.

When using ADAAS, all desired analysis operations (at present) are performed by the Statistical Research System (SRS) program package. This interface with SRS is transparent to the user during operation. Features of ADAAS described herein, however, do not generally apply to the SRS programs so that certain operations, such as attention interrupt processing, will only function in what appears to the user to be selected parts of ADAAS.

Developmental work is currently underway to modify the SRS analysis programs so as to provide a more uniform operating convention. At present, it must be remembered that portions of the analysis commands following the statement "DDEF STATEMENT=" do not correspond to the conventions described below. Operation instructions for this portion of program operation are given in Section 6.

A. OPERATION

The program is initiated by the MTS command:

$RUN HSRI:ADAAS

In order to successfully operate the program, this run command must be issued from a user ID that is authorized to access the system. Operation is terminated by entering "FINISH" in response to the prompt "COMMAND?".

B. OPERATING FORMAT

The user is primarily in contact with a supervisor that accepts and decodes commands, then performs the operations requested by these commands.

Commands are requested by the program prompt "COMMAND?". A number of possible commands may be entered in response to this request. These commands are listed below by category.

Control Commands

FINISH	To terminate execution
STOP	To provide a restartable return to MTS.

Information Commands

DESCRIBE	To describe commands.
LIST	To list commands.
NEWS	To provide current information on additions or changes to the accident file system.

Data Access Commands

DATA	To access data files.
RELEASE	To release data files.
TAPE	To terminate execution without releasing a magnetic tape file.

Data Analysis Commands

SETUP	To create control (or setup) files for analysis operations.
ANALYZE	To perform analysis operations.
PRINT	To print analysis results on a batch printer.

Each command is discussed separately in Section 5-F. As indicated by the underlining, the first two letters are sufficient to define the command to the supervisor.

C. ATTENTION INTERRUPT PROCESSING

Extensive attention interrupt processing is performed by ADAAS. This means that in most phases of the program, attention interrupts, or breaks, are intercepted by the program and the user is rerouted to a different (and hopefully more desirable) portion of the program. As an example, if a user asks for "DESC" (i.e., describe) in response to one of the program prompts and has received the information he desired, an attention interrupt will return to the original prompt request -- just as if the user had waited for the description operation to be completed.

The actual return point for all the possible attention interrupts is useless to the user since there is in general no way of telling what operation is currently being performed. Every attempt has been made, however, to return program control to the most logically useful point. Table 8 shows some return points for commonly encountered break points.

TABLE 8
Attention Interrupt Break & Return Points

BREAK	RETURN
Command?	MTS (Restartable Return)
Describe Operation	Program Prompt
Command Processing	Command?

D. SELF DESCRIBING FEATURES

A description of the information desired by the program in response to any request may be obtained by entering "DESC" in place of the necessary response.

E. BATCH OPERATION

Although the program is designed for terminal use, batch mode operation is provided for and has many advantages when the computer load is heavy. Program operation in batch and terminal mode is similar with the difference that execution is terminated when errors are encountered in batch. Program prompts and messages are otherwise identical in both modes.

F. ADAAS CONTROL COMMAND DESCRIPTIONS

COMMAND: ANALYZE

Purpose: To perform analysis operations on any data file to which access has been permitted by the DATA command. The actual analysis operations are performed by Statistical Research System programs; the desired program is selected by a keyword. The analysis output information generated by the SRS program may be recorded on any file or device selected by the user.

Prompt: ANALYSIS PROGRAM?

Defaults: None

Valid Responses:
1) "DESC" to describe program keywords.
2) A four letter analysis program keyword.

Description: The desired analysis program is selected by entering a four letter keyword. Use of "DESC" provides an on-line listing of the valid keywords. These keywords are described below:

KEYWORD	SRS NUMBER	PROGRAM
LIST	PC210	DATA SET LIST
HIST	PG680	HISTOGRAM
UNIV	PG610	UNIVARIATE FREQUENCIES
ANOV	PG640	ONE WAY ANALYSIS OF VARIANCE
BIVA	PG650	BIVARIATE FREQUENCIES

The "SRS NUMBER" designation refers to the computer name of the particular program.

Prompt: DATA STORAGE FILE OR DEVICE?

Defaults:
1) TBBL in terminal mode.
2) *SINK* in batch mode.

Valid Responses:
1) Any valid name for a permanent or temporary file.
2) Any valid pseudo-device name (i.e., *PRINT*, *DUMMY*, etc.).
3) "DESC" for a description

Description: Output information generated by the analysis program is assigned a destination through this command. Data may be stored in permanent or temporary files or routed to an output device such as *SINK* or *PRINT*.

When a file name is specified, a nine page file is created if the name does not correspond to an existing file. If the name does correspond to an existing file, the file is emptied in batch mode, or the user is asked if the file can be emptied in terminal mode.

Error Comments:

1) THERE IS NO ACTIVE FILE TO ANALYZE!

Access to a file has not been generated through the DATA command.

2) INCORRECT PROGRAM KEYWORD. USE "DESC" FOR CORRECT KEYWORDS.

3) FILE "FILENAME" CANNOT BE CREATED.

The specified storage file cannot be created because of an incorrect name, insufficient file space, or other technical reasons.

COMMAND: DATA

Purpose: To provide the program with access to any of the individual data files that are maintained as an integral part of the system. If the desired data files are stored on magnetic tape, then the appropriate tape is mounted on a drive.

Once access to a file has been generated by the DATA command, it can only be changed through a second application of the DATA command to access another file or by use of the

RELEASE command. File access information is maintained even though execution of the program is terminated and rerun at a later time during the same signon period. It is important that tapes are not manually dismounted if they have been mounted via the DATA command. Such action will lead to confusion of the program.

Prompt: DATA KEYWORD?

Default: None

Valid Responses:
1) "DESC" for a description.
2) A single letter requesting a list of Data Base Keywords.
3) A valid Data Base Keyword.

Description: Each data file is accessed by entering the appropriate data base keyword. Finding the proper keyword is aided by entering a single letter to obtain a listing of keywords for a group of files. The appropriate letter in turn may be found by using the describe routine "DESC".

If the use of the DATA command has resulted in a tape being mounted, then a second call on DATA to access a file maintained on disk storage or on a different tape will result in a request (in terminal mode) to confirm the new data request. If the response is affirmative, the current tape will be dismounted.

Note that only one DATA command specification is necessary for analysis of a given data file. That is, any number of analysis operations can be performed on the same file (through the ANALYZE command) by a single DATA/DATA or DATA/RELEASE sequence.

Error Comments: 1) INCORRECT DATA KEYWORD. TRY AGAIN.

Your data base keyword is incorrect. Use "DESC" to obtain a list of valid keywords.

2) ACCESS TO FILE " " CANNOT BE ESTABLISHED.

The desired access operation cannot be performed because all tape drives are busy or because the desired tape is in use.

COMMAND: DESC

Purpose: To provide a brief description of program operation on the function of the valid system commands.

COMMAND: FINISH

Purpose: To terminate operation of ADAAS with no capability to restart.

Error Comments: 1) A MAGNETIC TAPE IS STILL MOUNTED.

The data file should be released via RELEASE before program operation is terminated.

COMMAND: LIST

Purpose: To provide a list of the currently effective control commands.

COMMAND: NEWS

Purpose: To list recent changes or addition to the accident file data base in reverse chronological order.

COMMAND: PRINT

Purpose: To copy or list files or pseudo-devices to one of the available batch print stations. Four of these stations are currently connected to MTS: their location is discussed below.

The user can specify the desired print station, the number of copies desired, and a list of files or devices to be copied or listed to the printer. Any number of files may be transferred at one application of the PRINT command.

Prompt: PRINT ROUTE?

Default: CNTR

Valid Responses:

1) "DESC"
2) A four letter route code

Description:

The desired print station is selected by entering the appropriate four letter route code. Currently valid codes are:

CNTR Computer Center, Ann Arbor, Michigan
DBRN University of Michigan Dearborn Campus, Dearborn, Michigan.
HSRI Highway Safety Research Institute, Ann Arbor, Michigan.
NHSB Nassif Building, Washington, D.C.
NUBS Central Campus Station, Ann Arbor, Michigan.
SWRI SWRI, San Antonio, Texas

Prompt:

NUMBER OF COPIES?

Default:

1

Valid Responses:

1) "DESC"
2) A number in the range 1-9.

Description:

The desired number of print copies (1-9) should be entered.

Prompt:

PRINT LIST?

Valid Responses:

1) "DESC"
2) C FILENAME
3) L FILENAME
4) $ENDFILE

Description:

A list of files to be listed or copied to the printer should be supplied with a $ENDFILE to terminate the list. The list format is:

C FILENAME to $COPY FILENAME

L FILENAME to $LIST FILENAME

(NOTE: The "PRINT LIST" prompt will be issued following each copy or list command entered by the user.)

COMMAND RELEASE

Purpose: The use of this command releases the active status of any data file currently accessed by the program.

This command need only be used when no further use of the existing data file is anticipated and access to another file is not required.

COMMAND SETUP

This command is not currently implemented.

COMMAND STOP

Description: This command returns the user to MTS control. While in MTS, the user may perform any defined operation ($LIST, $COPY, etc.). Providing the commands $RUN or $LOAD are not issued while in MTS, control may be returned to ADAAS by issuing the command $RESTART.

If a $RUN command is issued, MTS automatically unloads ADAAS thereby preventing a restart. The user must be careful not to use commands such as $RUN *CATALOG if a restart is desired.

SECTION 6

OPERATING THE ADAAS ANALYSIS PROGRAMS

Operation of any of the five analysis programs discussed in Section 3 simply involves the entry of a number of different data statements whose formats are also discussed in Section 3. The analysis programs are accessed by issuing the ANALYZE command to ADAAS, and by designating the program desired by the proper Analysis Program Keyword.

A. OPERATING FORMAT

With the exception of the Data Set List program, the analysis programs all have similar input formats that are identical whether the system is run in conversation or batch mode. Data Set List differs only in that there will be no prompt for a recode statement, since recode is not a permissible operation in this program.

Following the DDEF request that is automatically answered by ADAAS, the program operating format for Data Set List is as follows:

1) FILTER (OR LABEL)=
2) LABEL=
3) ENTER PARAMETERS:

Filter, label, and parameter strings must be constructed in accordance with the definition of statement syntax in Section 3. If a label is entered in response to the filter request, there will be no prompt for a label.

Following the DDEF prompt (again, automatically answered by ADAAS), the operating format for the remaining ADAAS programs is as follows:

1) FILTER (RECODE OR LABEL)=
2) RECODE (OR LABEL)=
3) LABEL=
4) ENTER PARAMETERS:

Filter, recode, label, and parameter strings must be constructed in accordance with the definition of statement syntax in Section 3. The filter, recode, and label entries are optional, but with the provision that as a minimum, some label must be entered, and that the statements must be entered in the order indicated.

B. DATA SET LIST CONTROL PARAMETERS

One line of control parameters is required for the Data Set List program. The applicable keywords and modifiers are described below:

Modifiers

ALLV — All variables in the specified file are listed in a compact format, with no spaces between variable values. Each record has a case sequence number and a column guide heading.

DEFAULT: Only a subset of variables is listed. A variable list must be supplied.

DESC — A description of the program modifiers and keywords is printed. If this modifier is present, all other parameters entered on the same line are ignored.

DEFAULT: No description is printed.

NODICT — The dictionary records for the variables listed are not printed in the bulkfile output.

DEFAULT: The dictionary records are printed in the bulkfile.

SPACE — The output listing is double-spaced.

DEFAULT: The output listing is single-spaced.

Keywords

SKIP=N — Every N'th record that satisfies the global filter criteria is listed.

DEFAULT: Every record that satisfies the global filter criteria is listed.

STOP=N	Terminate processing after N records have been listed. DEFAULT: List all records.
VAR=1,2,4-8,12	Include those variables specified in the output list. DEFAULT: None.

Several examples of the use of these parameters are given below:

1) ENTER PARAMETERS:
?ALLV NODICT STOP=10

This parameter selection will produce a list of all variables for the first ten cases that satisfy the global filter. No dictionary will be printed.

2) ENTER PARAMETERS:
?STOP=0 VAR=1,3,5-8,15

Using STOP=0, no data records are listed, so this parameter choice is useful for listing the dictionary values of selected variables.

3) ENTER PARAMETERS:
?SKIP=5 VAR=1-6,18,38-42,176-190 SPACE

This set of parameters will produce a double-spaced listing of the variables selected for every fifth case which satisfies the global filter (i.e., cases 5, 10, 15, 20, etc.).

C. BARGRAPH CONTROL PARAMETERS

The bargraph program will produce up to 15 bargraphs per program execution. Parameters for each bargraph should be entered on a single line with up to 15 lines entered in sequence. The applicable keywords and modifiers are described below:

Modifiers

DESC	A description of the program modifiers and keywords is printed. If this modifier is entered, all other parameters entered on the same line are ignored. DEFAULT: No description is printed.

DONE

Indicates that all desired bargraphs have been specified. May be entered as part of the last bargraph specification, or on a line by itself.

DEFAULT: Additional bargraph entries will be specified.

MD

Include missing data codes in the bargraph.

DEFAULT: This is the default option.

NOMD

Exclude missing data codes from the bargraphs.

DEFAULT: Include missing data.

SAME

All parameters not explicitly specified for the current bargraph are set equal to those used in the previous one.

DEFAULT: Parameters must be specified for each bargraph.

Keywords

HV=4:1-6

Bargraph variable number and range of code values, which must be a range of two or greater.

DEFAULT: None

WV=675

Weight variable number.

DEFAULT: No weighting used.

FV1=212:3-15

First local filter and range. May be multiple-response. Only cases where the specified variable has a value in the indicated range will be included in the bargraph.

DEFAULT: All cases used.

FV2=147:3-4

Second local filter and range (see FV1)

DEFAULT: All cases used.

TN=347

Optional table number.

DEFAULT: First bargraph is numbered 1. In other cases, each bargraph is numbered sequentially from the preceding one.

ID=FATALS

Bargraph identification label. An eight-character label may be supplied for each bargraph.

DEFAULT: An ID consisting of the month-day-year is used. Once an ID has been specified, it will be used on all following bargraphs.

Several examples of the use of these parameters are shown below:

1) ENTER PARAMETERS:
?HV=4:1-8 ID=CAR1 TN=5479 DONE

A bargraph of code values 1-8 for Variable 4 will be produced. The table number will be 5479 and an identification string "CAR1" will be printed on the output.

2) ENTER PARAMETERS:
?HV=276:1-37 FV1=5:2 FV2=86:3-7 NOMD
?SAME MD WV=97 DONE

Two bargraphs of Variable 276 filtered by Variables 5 and 86 will be produced. In the first, missing data will be excluded. In the second, missing data will be included and the distribution will be weighted by the code values of Variable 97.

D. UNIVARIATE FREQUENCIES CONTROL PARAMETERS

One line of control parameters is required for the univariate frequencies program. The applicable modifiers and keywords are described below:

Modifiers

BOTH	The descriptive statistics and univariate frequencies are computed for all variables in the variable list.
DESC	A description of the program modifiers and keywords is printed. If this modifier is present, all other parameters entered on the same line are ignored. DEFAULT: No description is printed.
FREQ	Univariate frequencies are calculated for all variables in the variable list.
FREQ%	Univariate frequencies and percentages are calculated for all variables in the variable list.
SKEW	All descriptive statistics including the 3rd and 4th moments are computed for all variables in the variable list.

STAT	All descriptive statistics excluding the 3rd and 4th moments are computed for all variables in the variable list.

Keywords

VAR=1,2,4-8,12	Process those variables specified according to the modifiers listed above. DEFAULT: None.
WRITE=FILENAME	Frequencies are to be written on output file, FILENAME, to be used as input for HSRI:PD330. DEFAULT: Frequencies are not saved.
WVF=123	Frequencies are weighted by specified variable. DEFAULT: Frequencies are not weighted.
WVS=245	Statistics are weighted by specified variable. DEFAULT: Statistics are not weighted.

Several examples of the use of these parameters are shown below:

1) ENTER PARAMETERS:
?FREQ% VAR=12,34-37

Univariate frequencies and percentages will be computed for variables 12, and 34 through 37.

2) ENTER PARAMETERS:
?BOTH VAR=1-10 SKEW WVF=12

Descriptive statistics, including the 3rd and 4th moments, will be computed for Variables 1 through 10. In addition, univariate frequencies weighted by the code values of Variable 12 will be computed.

E. ONE-WAY ANALYSIS OF VARIANCE (ANOVA) CONTROL PARAMETERS

The ANOVA program will produce up to 100 tables per program execution. The applicable keywords and modifiers are described below:

Modifiers

DESC

A description of the program modifiers and keywords are printed. If this modifier is present, all other parameters entered on the same line are ignored.

DEFAULT: No description is printed.

DONE

Indicates that all desired ANOVA tables have been specified. This modifier may be entered on the last table specification line, or on a line by itself.

DEFAULT: Additional tables will be specified.

MISS

The missing data values for the dependent variable are not checked.

DEFAULT: Missing data values of the dependent variable are checked, and cases where the dependent variable has such a value will not be included in the table.

NOMISS

Checking of the missing data values for the dependent variable will be performed. (see MISS).

DEFAULT: This is the default option.

SAME

All parameters not explicitly specified on the current parameter input line will be set equal to those used in the previous line.

DEFAULT: All parameters desired must be specified.

TERM

For conversational mode only. The eta statistic, and the f-ratio with its degrees of freedom and significance level will be printed on line with the bulkfile table and line numbers.

DEFAULT: Only the bulkfile table and line numbers will be printed on line.

2DIGIT

The program will allow for control variable codes with a range of 0 to 99.

DEFAULT: Control variable codes with a range of 0 to 11 are assumed.

Keywords

CV=1-4,10-12

Control variable number list. The program will generate a separate table for each control variable listed.

DEFAULT: None

DV=25,27,29	Dependent variable number list. The program will generate a separate table for each dependent variable listed. DEFAULT: None.
WV=56	Weight variable number. DEFAULT: No weighting used.
FV1=35:1-24	First local filter and range. Only those cases where the specified variable has a value in the indicated range will be included in the ANOVA table. If the filter variable is multiple-response, only the first response value is checked. DEFAULT: All cases used.
FV2=56:1-3	Second local filter and range (see FV1).
TN=50	Optional table number. DEFAULT: First ANOVA table is numbered 1. In other cases, tables are numbered sequentially from preceding ones.
ID=DRINKING	Table identification label. The "ID" is an eight-character label with no embedded blanks. It may be supplied for each ANOVA table. DEFAULT: An "ID" consisting of the month-day-year is used. Once an "ID" has been specified, it will be used on all following tables.

Examples of the use of the above parameters follow:

1) ENTER PARAMETERS:
?CV=10,11 DV=1,2 TN=10 ID=VEH-1 DONE

Four ANOVA tables, numbered beginning with 10 and each labelled "VEH-1" will be produced. For Table 10, the control variable number will be 10 and the dependent table number will be 1; for Table 11, the control variable number will be 10 and the dependent variable will be 2; for Table 12, the control variable number will be 11 and the dependent variable number will be 1; and, for Table 13, the control variable number will be 11 and the dependent variable number will be 2.

2) ENTER PARAMETERS:
?CV=10 DV=5 MISS 2DIGIT
?SAME DV=7 FV1=34:1-3
?DONE

Two ANOVA tables will be produced. Both tables allow

control variables with a range of 0 to 99, and both will bypass checking missing data values of the dependent variable. The first table has Variable 5 as the dependent variable and Variable 10 as the control variable. The second table has Variable 7 as the dependent variable and Variable 10 as the control variable, and chooses those cases which satisfy the filter Variable 14.

F. BIVARIATE CONTROL PARAMETERS

The bivariate tables program will produce up to 50 tables per program execution. The applicable modifiers and keywords are described below:

Modifiers

DESC	A description of the program modifiers and keywords is printed. If this modifier is present, all other parameters entered on the same line are ignored. DEFAULT: No description is printed.
DONE	Indicates that all desired bivariate frequency tables have been specified. This modifier may be entered on the last table specification line, or on a line by itself. DEFAULT: Additional tables will be specified.
SAME	All parameters not explicitly specified on the current parameter input line will be set equal to those used in the previous line. DEFAULT: All parameters desired must be specified.
FREQ	Print frequencies on bivariate tables. DEFAULT: This is the default option.
NOFREQ	Suppress printing of frequencies on the bivariate tables. DEFAULT: Frequencies are printed.
COL%	Calculate percentages based on the column totals and print percentages on bivariate tables. DEFAULT: Column percentages are not calculated.

ROW%

Calculate percentages based on the row totals

DEFAULT: Row percentages are not calculated.

TOT%

Calculate total percentages based on the total case count for the table.

DEFAULT: Total percentages are not calculated.

ALL%

Calculate column, row, and total percentages.

DEFAULT: No percentages are calculated.

NO%

Do not calculate any percentages.

DEFAULT: This is the default option.

MD

Include missing data codes in the bivariate tables.

DEFAULT: This is the default option.

NOMD

Exclude missing data codes in the bivariate tables.

DEFAULT: Include missing data codes.

Keywords

CV=12,34-36,72,75

Column variable number list. The program will generate a separate table for each column variable listed.

DEFAULT: None.

RV=1-3,102

Row variable number list. The program will generate a separate table for each column variable listed.

DEFAULT: None.

CVR=25:10-20

Column variable with a range. Only those cases where the specified column variable has a value in the indicated range will be included in the bivariate table. The column numbers printed on the table will begin with the lowest number of the range. If no range is specified, no filtering will be performed, and the column number printed on the table will begin with zero. If "CV" and "CVR" keyword expressions are entered on the same parameter input line, the "CV" keyword expression will be ignored.

DEFAULT: None.

RVR=10:100-199	Row variable number with a range. Similarly to "CVR", only those cases where the specified row variable has a value in the indicated range will be included in the bivariate table. The row variable numbers printed on the table will begin with the lowest value of the range. If no range is specified, no filtering will be performed, and the row numbers printed on the table will begin with zero. If "RV" and "RVR" keyword expressions are entered on the same parameter input line, the "RV" keyword expression will be ignored. DEFAULT: None.
WV=102:1-3	Weight variable number. A range may optionally be specified with a weight variable, and cases will be filtered similarly as in a "CVR" keyword expression. If no range is specified, only those cases where the weight variable has a non-missing value will be included in the bivariate table. DEFAULT: No weighting used.
FV1=23:0-5	First local filter and range. Only those cases where the specified variable has a value in the indicated range will be included in the bivariate table. The filter variable may be a multiple response variable. DEFAULT: All cases used.
FV2=35:1-15	Second local filter and range. (see FV1).
FV3=164:5-9	Third local filter and range. (see FV1).
FV4=123:25-125	Fourth local filter and range. (see FV1).
TN=100	Optional table number. DEFAULT: First bivariate table is numbered 1. In other cases, tables are numbered sequentially from preceding ones.
ID=ROLLOVER	Table identification label. The "ID" is an eight-character label with no embedded blanks. It may be supplied for each bivariate table. DEFAULT: An "ID" consisting of the month-day-year is used. Once an "ID" has been specified, it will be used on all following tables.

SAVE=FILENAME — Bivariate tables are saved on output file, FILENAME, to be used as input for special purpose routines.

DEFAULT: Tables are not saved.

Examples follow:

1) ENTER PARAMETERS:
?CVR=45:10-20 RV=102,105 FV1=12:1-5 TN=100 ID=VEH2 DONE

Two bivariate tables filtered by Variables 12 and 45 will be produced. The first table will use as a column variable number 45 and a row variable number 102, and the second table will use the same column variable number and row variable number 105. The tables will be numbered 100 and 101 respectively and an idenfication string "VEH-2" will be printed on both tables in the bulkfile output.

2) ENTER PARAMETERS:
?CV=51 RV=164 NOMD FV1=14:1-8
?SAME NOFREQ ALL%
?SAME FREQ NO% MD FV1=NONE
?DONE

Three bivariate tables will be produced, each with Variable 51 for the column and Variable 104 for the row. The first two tables will exclude missing data and will be filtered by Variable 14. The first table will contain frequencies only and the second table will contain all the percentages of the first table. The third table will print bivariate frequencies including missing data with no percentages and no filtering by Variable 14.

APPENDIX A

PHYSICAL CHARACTERISTICS OF COMMON TERMINAL DEVICES

A. TELETYPES

The Model 33 and 35 Teletypes have a transmission rate of 10 characters per second and have a carriage width of 72 positions. These models do not provide lower-case characters nor several of the special symbols used in programming. Some models have paper tape reading and punching equipment built into them.

The Model 37 Teletype transmits at a rate of 15 characters per second and has a carriage width of 75 positions. Both upper and lower-case alphabetics and many special symbols can be used with the Model 37. Because the speed of the Model 37 is faster than other teletypes, this model can only be connected to MTS through the Data Concentrator (i.e., phone line 763-1500).

Most keys and controls are common to all Teletypes. The position and shape vary considerably, but the functions are identical.

Teletypes are capable of both half-duplex and full-duplex mode of operation. In half-duplex mode, the keyboard is connected to the printer so that whenever a key is pressed it is immediately printed, as well as sent to the computer. In full-duplex mode, the keyboard and printer are not connected. A character is printed only when sent from the computer. In this mode of operation, the computer will usually "echo" each keystroke so the user can see what he is typing. The control for HDX-FDX operations may be located in two places. On some Model 35 Teletypes (the ASR models), there is a switch to the left of the keyboard. Twisting it clockwise will put it into half-duplex mode. On the Model 33 and some Model 35 Teletypes (the ASR models), there is a toggle switch located above the telephone dial. Switching it to the left will put it into the half-duplex position.

Under the telephone dial, six buttons are found which are

primarily concerned with the connection of the Teletype with the telephone lines and MTS. The ORIG, or "originate" button is located at the extreme left. This button must be pressed to connect the Teletype with the telephone line.

The CLR or "clear" button is generally the second from the left. This button disconnects the telephone line and severs communication with MTS. This is pressed if, for example, the user gets a busy signal in response to his call, or if he gets no answer at all. Using the CLR button is not the recommended way to sign off.

The button labelled LCL or "local" is used to operate the Teletype as a regular typewriter. This button enables the keyboard to work but makes no connection to the telephone lines and MTS.

The ordinary character keys on the keyboard act much like typewriter keys except that as their symbol is printed (if operating in half-duplex mode), the code for that symbol is sent to the computer. The SHIFT key selects the upper character on dual character keys. Note that some keys do not have an upper-shift function and may not be pressed in combination with the SHIFT.

A special key labelled CTRL for "control" is similar to the SHIFT key in that it selects an alternate function for a key. The code sent to the computer by a key pressed in combination with the CTRL key usually does not represent a printable character and thus no symbol is printed. Some CTRL combinations are used for editing purposes by the device support routines (e.g., backspace and "end-of-line"). These combinations have been designated in this guide by "CONTROL-x" where x is the character key operated in conjunction with the CTRL key. Note that the labels on the upper part of the alphabetic keys generally refer to the control function, not to the shift function. Many Teletypes have color-coded labels: white labels for shift functions and red labels for control functions.

The LINE FEED key causes the paper to move up one line

without changing the lateral position of the typing element. It also sends a LINE FEED character to the computer. This key is not used for any special purpose.

The RETURN key causes the typing element to return to the beginning of the line without spacing the paper. It sends a RETURN character to the computer. This signifies "end-of-line".

The RUBOUT key has meaning only when the Teletype is connected through the Data Concentrator. It signals a line-delete (and a "#" is echoed as a response).

The REPT key, when pressed in conjunction with any character key, causes that character to be repeated until the key is released. This is useful for spacing forward or for multiple "backspaces".

The BREAK key causes an attention signal to be sent to the computer to interrupt the current operation. This key is located to the left of the keyboard on a Model 35 Teletype, and at the right end of the Model 33. Any input or output in progress will be terminated. If the Model 35 is being used, the BRK RLS key, located at the left of the keyboard, must be operated after the BREAK key has been pressed if it lights up.

The ESC key is not found on all Teletypes. It is used only with the Data Concentrator to stop printing. If the Teletype is not equipped with the ESC key, the same function is provided by the combination of "CONTROL-SHIFT-K".

Two other keys are available on the Model 35. They are LOC LF and LOC CR. These allow the user to space the paper up and to perform a carriage return, respectively, without sending signals on to the communications line.

B. THE IBM 2741

The IBM 2741 is similar to the IBM Selectric typewriter. The table on which the IBM 2741 stands contains the control electronics for the typewriter and communications line. A mode switch

on the left side of the table has two positions: LCL and COM, for "local" and "communicate". In COM mode, the terminal is connected to the communications line, while in LCL mode, it may be used only as an ordinary typewriter. Power to the terminal is controlled by an ON-OFF rocker switch on the right side of the keyboard. This switch should be turned OFF when the terminal is not in use.

Most of the normal typewriter controls exist on an IBM 2741. There is a lever on the right rear of the typewriter cover which lessens the pressure on the paper and allows paper adjustment. On a pin-feed IBM 2741, this lever must be set so there is no pressure on the paper or it will not feed properly. A lever on the left adjusts the typing-head pressure to compensate for varying thicknesses of paper. The typing elements may be changed to provide a variety of character sets.

Left and right margin stops limit the travel of the typing element. The MAR REL key will temporarily release these stops. The margins may be used in conjunction with the margin editing facility of the device support routines to position the printing on the paper. A small pointer rides in a slot between the margin stops and indicates the current printing element position.

Physical tab stops may be set using the CLR-SET rocker switch on the left side of the keyboard. These may be used in conjunction with the input tabulation editing facility of the device support routines.

The IBM 2741 keyboard is very similar to that of an ordinary electric typewriter. When a character key is pressed, the symbol is printed on the paper and at the same time the internal code for that character is transmitted to the computer. The shift key allows upper and lower-case alphabetic characters to be produced and selects the upper or lower symbol on dual-character keys.

The TAB key causes the typing element to move to the next tab stop while sending the tab character code to the system. The backspace key moves the typing element back one position while sending a backspace character code to the system. Note that

several keys, among them the space bar and the backspace key, may have a "typamatic" mode. By pressing the key and holding it fully down, the user may have its function repeated until the key is released.

The RETURN key causes the typing element to return to the left margin and spaces the paper up one line. It also sends a return character to the system and locks the keyboard. The RETURN code informs the computer that the user has completed a line. The keyboard remains locked until the system again requests input. While locked, all keys except ATTN are inoperative.

The ATTN key is located on the upper right keyboard. When this key is pressed, an attention interrupt is signalled to MTS.

C. THE GE TERMI NET 300*

The GE TermiNet 300 is a high-speed (30 characters/second) impact-printing terminal with a carriage width of 118 characters that is utilized by some users of the ADAAS system. Although not generally supported by MTS, it may be operated like a low-speed teletype at 10 characters per second by utilizing the teletype phone line (763-0300). Using a special signon procedure that is described below, this terminal may be operated at 30 characters/second by using the inherent flexibility of the Data Concentrator (763-1500).

The keyboard is similar to that of a standard typewriter with the addition of a number of extra control keys and switches. A detailed description may be found in the GE publication GEH-2184A entitled "Operations Manual for the TermiNet 300 Prints."

To use the TermiNet 300 at 30 CPS, instead of the normal 10 CPS, do the following:

1. Turn on terminal, set duplex switch to "HALF" on back of coupler, push "ON LINE", set rate to "30", all else as normally used.

---

*This terminal is not optimally supported by MTS but is included here because it is employed by ADAAS users at NHTSA in Washington, D.C.

2. Dial (313) 763-1500 for the UM Data Concentrator, wait for tone.

3. Place phone in coupler, Press "INTERRUPT" until READY light stays on.

4. Hit "RETURN" key. If "LF" and "SPECIAL" are in the computer response you are on!

5. Enter "CTL-A CONTROL OFF=DUPLEX" to stop the double printing of your input.

6. To utilize the full 118 character page width, at 30 CPS, enter the following two device commands:

   "CTL-A CTL-A RM=118"

   "CTL-A CTL-A LEN=255"

7. Proceed with terminal operations.

APPENDIX B

ONE-WAY ANALYSIS OF VARIANCE

A. INTRODUCTION

The method of one-way analysis of variance is used to test the equality of two or more population means (r,r>2), when the data are composed of r independent random samples. Thus, we might state our hypotheses as:

H_0(null hypothesis): $\mu_1=\mu_{2=} \cdots = \mu_r$

H_1(alternative hypothesis): $\mu_i \neq \mu_j$ at least two means are not equal where μ_j; j=1,2,....,r is the mean of the jth population, etc. for μ_i.

In many situations it may be of interest to compare a number of means. For example; a company testing three brands of tires may want to know if the average life of each brand is the same, by measuring tread depth after some specified hours of testing.

B. STATISTIC METHOD:

Under the assumptions: (a) the r columns represent r random samples from r populations; (b) each of the r populations is normally distributed; (c) each of the r populations has the same variance:

The observations can be arranged as shown.

The first column represents a random sample of size n, from the population that has received condition 1; the second column represents a random sample of size n_2 from the population that has received condition 2, etc.

	Conditions				
	1	2	3--------r		
Observations	x_{11}	x_{12}	x_{13}	x_{1r}	
	x_{21}	x_{22}	x_{23}	x_{2r}	
	$\vdots$	$\vdots$	$\vdots$	$\vdots$	
	$x_{n_1 1}$	$x_{n_2 2}$	$x_{n_3 3}$	$x_{n_n r}$	
Totals	$T._1$	$T._2$	$T._3$	$T._r$	$T..$
Means	$x._1$	$x._2$	$x._3$	$\bar{x}._r$	$\bar{x}..$
Variances	S_1^2	S_2^2	S_3^3	Sr^2	S^2

The total of the jth column is represented by $T._j$.

$$T._j = \sum_{i=1}^{n_j} X_{ij}$$

where X_{ij} is the ith observation under condition j.

The mean of the jth column is expressed by $\bar{X}._j$ where

$$\bar{X}._j = \frac{T._j}{n_j}$$

The variance of the jth column S_j^2 is defined as:

$$S_j^2 = \frac{\sum_{i=1}^{n_j} (X_{ij} - \bar{X}._j)^2}{n_j - 1}$$

The total of all observations T.. is evaluated by:

$$T.. = \sum_{j=1}^{r} T._j$$

The mean of all observations $\bar{X}..$ can be computed by:

$$\bar{X}.. = \frac{T..}{N}$$

where N the total number of observations $= \sum_{j=1}^{r} n_j$

The variance of all N observations regarded as a single sample is defined as S^2 where:

$$S^2 = \frac{\sum_{j=1}^{r} \sum_{i=1}^{n_j} (X_{ij} - \bar{X}..)^2}{N-1}$$

One-way Analysis of Variance Table

Source of Variation	SS	d.f.	MS	EMS	F
Among Groups	SS_A	r-1	$SS_A/(r-1)=MS_A$	$\sigma^2+\sum^{r}\frac{n_j(\mu_j-\mu)^2}{r-1}$	MS_A/MS_W
Within Groups	SS_W	N-r	$SS_W/(N-r)=MS_W$	σ^2	
Total	SS_T	N-1			

The total sum of squares is represented by SS_T.

$$SS_T = \sum_{j=1}^{r} \sum_{i=1}^{n_j} X_{ij}^2 - \frac{T_{..}^2}{N}$$

The within sample or within groups sum of squares is represented by SS_W.

$$SS_W = \sum_{j=1}^{r} \sum_{i=1}^{n_j} X_{ij}^2 - \sum_{j=1}^{r} \frac{T_j^2}{n_j} = SS_T - SS_A$$

The among samples or among groups sum of squares is referenced by SS_A.

$$SS_A = \sum_{j=1}^{r} \frac{T_{.j}^2}{n_j} - \frac{T_{..}^2}{N}$$

The expected mean square (abbreviated EMS) is defined as:

$$= \alpha^2 + \sum_{j=1}^{r} \frac{n_j(\mu_j - \mu)^2}{r-1}$$

The "ETA" statistic is known as the correlation ratio. The correlation between X and Y can be defined as:

$$\rho_{xy} = \frac{Cov(X,Y)}{\sqrt{(VarX)(VarY)}} = \frac{E[XY] - \mu_x \mu_y}{\sigma_x \sigma_y}$$

ETA, the "correlation ratio" is a measure of the total variance of the dependent variable accounted for (both linear and

non-linear variance) by the difference between groups in the independent variable. ETA gives a rough indication of the correlation between the independent and dependent variables.

A necessary assumption for valid analysis of variance is the homogeneity or equality of the group variances. If one cannot meet this assumption, any significant or non-significant difference between groups is suspect because of the possible invalidity of the analysis of variance. In order to test this assumption, one can perform the F-ratio test by simply dividing the variance of the largest group by the variance of the smallest group ($F = \frac{V_{large}}{V_{small}}$) and examining a table of F distributions with $df_{large}=N-1$ (where N is the number of observations in the large group) and $df_{small}=L-1$ (where L is the number of observations in the small group). Tables of F-ratios are available in most statistical texts (C. F. Winer). In general the ANOVA is known to be robust with regard to violations of homoscedasticity (equal variances, normally distributed) and thus can tolerate mild disturbances of these parameters.

C. STATISTICS - REFERENCES

1. The one-way analysis of variance
   William C. Guenther: ANALYSIS OF VARIANCE, 1964 by Prentice-Hall, Inc., Englewood Cliffs, New Jersey. pp. 31-43.

2. CORRELATION COEFFICIENT
   Harold J. Larson: INTRODUCTION TO PROBABILITY THEORY AND STATISTICAL INFERENCE, 1969 by John Wiley & Sons, Inc., pp. 156, 163

3. Analysis of Variance (General)
   B. J. Winer: STATISTICAL PRINCIPLES IN EXPERIMENTAL DESIGN, 2nd Edition, 1971 McGraw-Hill.

APPENDIX C

LISTING ALPHANUMERIC TRANSLATIONS OF NUMERIC VARIABLES

The user has access to two special purpose programs for listing cases that provide for the translation of numeric code values into their corresponding alphanumeric definitions. The alphabetic data set list (e.g., $SOURCE HSRI:SPAD(N); N=307, 317, or 327) program permits the user to select which variables are to be translated in the standard tabular listing. The CPIR case writing program ($SOURCE HSRI:CASEWRITER) prints a one-page case summary according to a predetermined format.

A. ALPHABETIC DATA SET LIST

This is a new version of the data set list program that will translate numeric code values into their alphanumeric equivalent definitions. The program is operated identically to the SRS data set list program except that the user is permitted to use a "T" (for translate) for "V" on any variable in the Variable List that is to be translated. Variable numbers may be repeated and specified in any order desired for printing. The "V" and "T" can be freely intermixed. Thus, for example, the user can list both the standard (numeric) code value right beside the translated code value by specifying V10, T10. Requesting the translation of a variable for which no specific translation table has been provided will simply result in its printing of the standard (numeric) code value. All responses of MRV's (multiple response variables) are translated and printed.

If the code value definitions for Variable 10 were:

0 Unknown
1 Urban
2 Rural

the resulting alphabetic data set list printout would appear as follows:

Standard/Numeric	Translated
0	UKN
1	URBAN
1	URBAN
2	RURAL
1	URBAN

Note that it is fairly easy to fill an entire printout line with just a few variables, so caution should be exercised in limiting both the number of cases and the number of variables to be printed. See Figure 5 for sample execution and Figure 6 for sample printout. The Variable List for the example was:

V16,V20,T20,T36,T39*

The alphabetic data set list program is programmed to run on the CPIR Revision 3 Vehicle, Occupant and Injury SPAD data files (300, 310, 320) with the SPAD analysis number 7. Two SOURCE HSRI:SPAD commands are used to mount the desired tape and list the desired cases as follows:

1. For the CPIR Revision 3 Vehicle File - enter:

   $SOURCE HSRI:SPAD(300) - tape mount
   $SOURCE HSRI:SPAD(309) - alphalist program

2. For the CPIR Revision 3 Occupant File - enter:

   $SOURCE HSRI:SPAD(310) - tape mount
   $SOURCE HSRI:SPAD(317) - alphalist program

3. For the CPIR Revision 3 Injury File - enter:

   $SOURCE HSRI:SPAD(320) - tape mount
   $SOURCE HSRI:SPAD(327) - alphalist program

Once the tape is mounted, the program has been requested (e.g., SPAD(317), and program execution has begun, the following prompts for user input will be received. Respond to each in turn with a (1) SRS Filter Statement, (2) SRS Label Statement

```
sou hsri:spad(300)
#$RUN HSRI:TAPE PAR=1000
#EXECUTION BEGINS

 TAPE MOUNTED


#EXECUTION TERMINATED
#$COM    ***$COPY HSRI:*EMS***
#sou hsri:spad(307)
#$RUN SDGP:ALPHALIST 5=*SOURCE**MSOURCE* 8=TBBL 12=SDGP:DATAMA
#EXECUTION BEGINS
  DDEF STATEMENT=

 UNIT  1 READING TAPE *T*              , CALIN   , FILE IS VEHDIC

 UNIT  2 READING TAPE *T*              , CALIN   , FILE IS VEHDAT
 FILTER OR LABEL
 include v20=5*
 LABEL =
 an example of coir cases involving 'sleet'

 ENTER)PARAMETERS (OR DESC):


 THE VARIABLE LIST IS:
 v16,v20,t20,t36,t59*

 VARIABLE NUMBERS:
 START=   1
 END  = 576

 LOGICAL RECORD LENGTHS:
 OUTPUT=  15
 INPUT = 836




 ***** PROCESSING COMPLETE
 NUMBER OF CASES=     13
#EXECUTION TERMINATED
#sou hsri:spad
#$RELEASE *T*
#T204 RELEASED.
#sou hsri:spad(2)
#*SET ECHO=OFF
#*PRINT* ASSIGNED RECEIPT NUMBER 626145
#*PRINT* 626145 RELEASED, 6 PAGES
#sig *
#OFF AT 13:50.53    01-20-74
#  $3.70
#$217.94
```

FIGURE 5 Alphalist Program Execution Example

```
TIME: 13:49.13
DATE: JAN 29, 1974

DDEF IS:  PN=*T*,VO=CALIN,C=VEHDIC,D=VEHDAT,END

  FILTER CARD IS:
   INCLUDE V29=5*



LABEL:
AN EXAMPLE OF CPIR CASES INVOLVING 'SLEET'

VARIABLE LIST:
V16,V20,T20,T36,T59*

OPTIONS SPECIFIED:
IFALL  =     1
LSTNTH =     1
ISPACE =     0
NODICT =     0

VARIABLE NUMBERS:
START=   1
END  = 576
```

```
 LISTING OF DICTIONARY IN SAME ORDER AS RECORDS
VAR.#      VARIABLE NAME      LOC.  WID DEC RESP CTYPVTYP MDCODE1 MDCODE2
  16 TEAM CASE NUMBER          31   11   0    1    1    0
  20 URBAN/RURAL AREA          64    1   0    1    0    0  0000000
  20 URBAN/RURAL AREA          64    1   0    1    0    0  0000000
  36 TIME OF DAY               81    1   0    1    0    0  0000000
  59 COLLISION-VEH. TO VEH.   106    1   0    1    0    0  0000000

LOGICAL RECORD LENGTHS:
OUTPUT=  15
INPUT = 836
```

```
                              AN EXAMPLE OF CPIR CASES INVOLVING 'SLEET'
VAR.#= 16              20   20       36       59
     1  CAL 71 52A     2   RURAL    DAY      HEAD-ON
     2  CAL 71 107A    2   RURAL    NIGHT    HEAD-ON
     3  CA 71 109A     2   RURAL    NIGHT    VEH-OBJ
     4  CAL 71 118A    2   RURAL    NIGHT    VEH-OBJ
     5  CAL 71 119A    2   RURAL    NIGHT    VEH-OBJ
     6  CAL 71 240A    2   RURAL    NIGHT    VEH-OBJ
     7  CAL 72 28A     2   RURAL    NIGHT    HEAD-ON
     8  CAL 72 82A     1   URBAN    NIGHT    VEH-OBJ
     9  CAL 72 96 A    2   RURAL    DAY      TYPE L
    10  CAL 72 323A    2   RURAL    DAY      VEH-OBJ
    11  CAL 72 345A    1   URBAN    DAY      TYPE L
    12  CAL 72 361A    1   URBAN    NIGHT    VEH-OBJ
    13  OK 183         1   URBAN    NIGHT    TYPE T
***** PROCESSING COMPLETE
NUMBER OF CASES=     13
```

FIGURE 6
Alphalist Program Sample Output

(or "Return"), (3) "Return" for parameter defaults (or user prescribed parameters), and (4) SRS Variable List, using "T" instead of "V" for those variables to be translated into alphanumeric equivalents. The prompts and responses are displayed below:

Program Prompts/Responses:

1. FILTER OR LABEL=
   Filter

2. LABEL=
   Label

3. ENTER PARAMETERS (OR DESC):

For default values (1,1,0,0) enter an empty line (hit Return key). Otherwise, enter the program control parameters IFALL, LSTNTH, ISPACE, and NODICT as described below:

IFALL:	1	If only a subset of variables are to be listed; a variable list card must be supplied.
	0	If all variables in the data set are to be listed.
LSTNTH:	1	If every record (which satisfies any filter conditions) is to be listed.
	N	Records between successive increments of "N" are not listed (Filter conditions have higher priority than "N").
ISPACE:	0	Single space data for each case.
	1	Double space data for each case.
NODICT:	0	If dictionary is to be listed.
	1	If dictionary is not to be listed.

Enter parameters in the order IFALL,LSTNTH,ISPACE,NODICT. Follow each value by a comma (for example: 1,5,1,0).

4. THE VARIABLE LIST IS:

Enter the Variable List Statement showing the variables to be listed in the order in which they are to appear in the output.

Use "T" instead of "V" to prefix those variable numbers to be translated to alphanumerics.

B. CPIR CASE SUMMARY WRITER

When reviewing a limited number of CPIR cases pertaining to some selected area of inquiry, it is convenient to have a one-page case summary that will provide all the pertinent features describing the accident in an easily readable format. This function is provided by the CASEWRITER program which accesses the latest CPIR Revision 3 occupant file and produces formatted case summaries in English for the subset of cases specified by the user.

The program is accessed by the MTS command:

$SOURCE HSRI:CASEWRITER

The operation of the program initiated by this command is shown in Figure 7. Note that the program is largely self-explanatory and descriptions of each ensuing operation are given.

Following the initial command to start the program, only two additional control inputs are required. The cases selected for case summary production are selected by an appropriate SRS filter statement. The command "$ENDFILE" should then be issued in response to the "LABEL=" request.

When program operation is complete, the case summaries are stored in a temporary file "-TBBL". A carriage width of 132 characters is necessary to reproduce the entire table, but much of the available information is printable on a teletype.

Examples of the case summaries produced by the sample run described above are shown in Figures 8 and 9 .

```
#$source hsri:casewriter                    Command to initiate
#$SET ECHO=OFF                              Program operation

TRAL CASEWRITER
HIGHWAY SAFETY RESEARCH INSTITUTE
VERSION 12-18-75
CLOCK  10:41.1?  DATE  01-18-76

***** NOTE *****
A MAGNETIC TAPE WILL BE MOUNTED NEXT.
PLEASE WAIT FOR MOUNT CONFIRMATION.

IF THE MOUNT IS UNSUCCESSFUL, ATTENTION BREAK
AND THEN $SOURCE HSRI:CASEWRITER(80).
**********

TAPE MOUNTED


***** NOTE *****
IN THE NEXT PROGRAM, THE FOLLOWING RESPONSES
SHOULD BE ENTERED AT THE PLACE INDICATED.

 1) ENTER FILTER AS REQUESTED IN SRS FORMAT.
    IF THERE IS NO FILTER, ENTER "$ENDFILE".

 2) IF A FILTER IS USED, ENTER "SENDFILE"
    AFTER THE LABEL REQUEST.
**********



 DDEF STATEMENT=

UNIT  1 READING TAPE *T*

UNIT  2 READING TAPE *T*
FILTER OR LABEL
include v5=01 and v3=00100*                 SRS Filter Statement
LABEL =                                     and End-of-File indica-
$endfile                                    tion.

DESCRIPTION OF PARAMETERS?
PARAMETER CARD
....5...10...15...20...25...30

OUTPUT DATA SET IS 01 CARDS

OUTPUT WILL HAVE 80 CHARACTERS

NO CARD WEIGHTING SPECIFIED


NUMBER OF CARDS/RECORD: 4


 STOP     0
T908 RELEASED.

***** NOTE *****
EACH RECORD FOUND BY THE RETRIEVAL PROGRAM
CORRESPONDS TO A SINGLE OCCUPANT. SEVERAL
RECORDS MAY BE USED IN A SINGLE CASE SUMMARY.


***NUMBER OF OUTPUT RECORDS=     10
**********

A TOTAL OF    2 CASE SUMMARIES HAVE BEEN WRITTEN.

***** NOTE *****
 PROGRAM OPERATION IS NOW COMPLETED.
  THE CASEWRITER SUMMARIES ARE STORED
  IN TEMPORARY FILE "-T08L".
**********
```

FIGURE 7 Casewriter Execution Example

ACCIDENT INVESTIGATION SUMMARY--UNIVERSITY OF MICHIGAN HIGHWAY SAFETY RESEARCH INSTITUTE

CASE NUMBER AA 100 ACCIDENT DATE 7-17-71 AA 100

	DATA	NOTES
CASE VEHICLE	A 1969 PONTIAC FULL SIZE STATION WAGON, 9 OCCUPANTS	
OBJECT STRUCK		
OTHER VEHICLE	A 1968 PONTIAC INTERMEDIATE CONVERTIBLE,	
COLLISION TYPE	HEAD-ON	
DAMAGE INDEX	PRIMARY 11-FDEW-6	
OCCUPANT I. D.	DRIVER /23YRS, 140LBS, MALE, C-FRONT /MALE, R-FRONT /MALE, L-REAR /MALE, MALE, MALE,	
INJURY SEVERITY	DRIVER /FATAL- 2 REGIONS, C-FRONT /FATAL- 2 REGIONS, R-FRONT /FATAL- 2 REGIONS, L-REAR /FATAL- 2 REGIONS, FATAL- 2 REGIONS, FATAL- 2 REGIONS,	
RESTRAINTS USED	DRIVER /NONE C-FRONT /NONE R-FRONT /NONE L-REAR /NONE NONE NONE	
EST. IMPACT SPEED	CASE VEHICLE-60 MPH, OTHER VEHICLE-80 MPH,	
ST'G. COL. COLLAPSE: E. A. DEVICE CAPSULES TELESCOPING UNIT		
SEAT PERFORMANCE: LATCH TRACK OTHER	NO DAMAGE	
GENERAL COMMENTS	63 IN. CRUSH TO FRONT FUEL LEAKAGE PRESENT ALCOHOL OR OTHER INTOXICANTS INVOLVED OCCUPANT EJECTION: L-REAR /	

FIGURE 8
Example of Case Summaries Produced by Casewriter Program

ACCIDENT INVESTIGATION SUMMARY--UNIVERSITY OF MICHIGAN HIGHWAY SAFETY RESEARCH INSTITUTE

CASE NUMBER AA 100 ACCIDENT DATE 7-17-71 AA 100

	DATA	NOTES
CASE VEHICLE	A 1968 PONTIAC INTERMEDIATE CONVERTIBLE, 1 OCCUPANTS	
OBJECT STRUCK		
OTHER VEHICLE	A 1969 PONTIAC FULL SIZE STATION WAGON,	
COLLISION TYPE	HEAD-ON	
DAMAGE INDEX	PRIMARY 11-FDEW-8	
OCCUPANT I. D.	DRIVER /25YRS, 180LBS, MALE,	
INJURY SEVERITY	DRIVER /FATAL- 2 REGIONS,	
RESTRAINTS USED	DRIVER /NONE	
EST. IMPACT SPEED	CASE VEHICLE-80 MPH, OTHER VEHICLE-60 MPH,	
ST'G. COL. COLLAPSE: E. A. DEVICE CAPSULES TELESCOPING UNIT	8.7 IN. COMPRESSION 4.0 IN. SEPARATION	
SEAT PERFORMANCE: LATCH TRACK OTHER	L-FRONT LOCK FAILED, R-FRONT LOCK FAILED,	
GENERAL COMMENTS	82 IN. CRUSH TO FRONT FUEL LEAKAGE PRESENT ALCOHOL OR OTHER INTOXICANTS INVOLVED OCCUPANT EJECTION: DRIVER /	

FIGURE 9
Additional Example of Case Summaries Produced by Casewriter Program