TECHNICAL REPORT STANDARD TITLE PAGE

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.
FHWA/TX-88/456-1F, Vol.II		

4. Title and Subtitle	5. Report Date
Texas Flexible Pavement Data Base	August 1988
	6. Performing Organization Code
7. Author(s)	8. Performing Organization Report No.
Rebecca Yette, Trevor Pereira, Victor Wong	Research Report 456-1F,Vol. II
9. Performing Organization Name and Address	10. Work Unit No.
Texas Transportation Institute The Texas A&M University System College Station, Texas 77843-3135	11. Contract or Grant No. Study No. 2-8-86-456
	13. Type of Report and Period Covered Final- September 1985 - August 1988
12. Sponsoring Agency Name and Address	14. Sponsoring Agency Code
Texas State Department of Highways and Public Transportation; Transportation Planning Division P.O. Box 5051 Austin, Texas 78763	

15. Supplementary Notes

Research performed in cooperation with FHWA. Research Study Title: Texas Flexible Pavement Data Base.

16. Abstract

This volume of the final report is the programmers manual for the Texas Flexible Pavement Database System. It describes the system from the programmer's point of view and includes a system overview, general program narratives, flow diagrams, program specifications, file layouts, report layouts, sample reports, sample screens and program listings.

17. Key Words	18. Distribution Statement
Microcomputer; Flexible Pavement; Database; Pavement Performance	No restriction. This document is available to the public through the National Technical Information Service 5285 Port Royal Road Springfield, Virginia 22161

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of Pages	22. Price
Unclassified	Unclassified	683	

Form DOT F 1700.7 (8-69)

TEXAS FLEXIBLE PAVEMENT DATABASE

VOLUME II. PROGRAMMER'S MANUAL

By

Rebecca Yette, Trevor Pereira and Victor Wong

Research Report
456-1F Volume II

on

Research Study Number 2-8-86-456
Texas Flexible Pavement Database

Sponsored By
Texas State Department of Highways & Public Transportation

In Cooperation with
Federal Highway Administration

August 1988

Texas Transportation Institute
Texas A&M University System
College Station, Texas

METRIC (SI*) CONVERSION FACTORS

APPROXIMATE CONVERSIONS TO SI UNITS

Symbol	When You Know	Multiply By	To Find	Symbol
LENGTH				
in	inches	2.54	centimetres	cm
ft	feet	0.3048	metres	m
yd	yards	0.914	metres	m
mi	miles	1.61	kilometres	km
AREA				
in^2	square inches	645.2	centimetres squared	cm^2
ft^2	square feet	0.0929	metres squared	m^2
yd^2	square yards	0.836	metres squared	m^2
mi^2	square miles	2.59	kilometres squared	km^2
ac	acres	0.395	hectares	ha
MASS (weight)				
oz	ounces	28.35	grams	g
lb	pounds	0.454	kilograms	kg
T	short tons (2000 lb)	0.907	megagrams	Mg
VOLUME				
fl oz	fluid ounces	29.57	millilitres	mL
gal	gallons	3.785	litres	L
ft^3	cubic feet	0.0328	metres cubed	m^3
yd^3	cubic yards	0.0765	metres cubed	m^3

NOTE: Volumes greater than 1000 L shall be shown in m^3.

TEMPERATURE (exact)

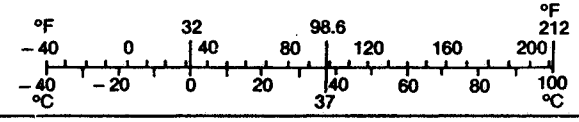
°F	Fahrenheit temperature	5/9 (after subtracting 32)	Celsius temperature	°C

APPROXIMATE CONVERSIONS TO SI UNITS

Symbol	When You Know	Multiply By	To Find	Symbol
LENGTH				
mm	millimetres	0.039	inches	in
m	metres	3.28	feet	ft
m	metres	1.09	yards	yd
km	kilometres	0.621	miles	mi
AREA				
mm^2	millimetres squared	0.0016	square inches	in^2
m^2	metres squared	10.764	square feet	ft^2
km^2	kilometres squared	0.39	square miles	mi^2
ha	hectores (10 000 m^2)	2.53	acres	ac
MASS (weight)				
g	grams	0.0353	ounces	oz
kg	kilograms	2.205	pounds	lb
Mg	megagrams (1 000 kg)	1.103	short tons	T
VOLUME				
mL	millilitres	0.034	fluid ounces	fl oz
L	litres	0.264	gallons	gal
m^3	metres cubed	35.315	cubic feet	ft^3
m^3	metres cubed	1.308	cubic yards	yd^3

TEMPERATURE (exact)

°C	Celsius temperature	9/5 (then add 32)	Fahrenheit temperature	°F



These factors conform to the requirement of FHWA Order 5190.1A.

* SI is the symbol for the International System of Measurements

DISCLAIMER

The statements and claims expressed in this document are statements and claims of the authors, and do not necessarily represent the official position of The State of Texas, The State Department of Highways and Public Transportation, or any political subdivision of the State or Federal Government regarding the subject matter.

ABSTRACT

This volume of the final report is the programmers manual for the Texas Flexible Pavement Database System. It describes the system from the programmmer's point of view and includes a system overview, general program narratives, flow diagrams, program specifications, file layouts, report layouts, sample reports, sample screens and program listings.

TABLE OF CONTENTS

CHAPTER 6: DATABASE MAINTENANCE SUBSYSTEM

LIST OF APPENDICES

APPENDIX A: File Layouts Page

LIST OF EXHIBITS

LIST OF FIGURES

LIST OF TABLES

INTRODUCTION

The development of the Texas Flexible Pavement Database System was sponsored by the Texas Department of Highways and Public Transportation and conducted by Texas Transportation Institute. Whereas Volume 1 of this report explains the purpose of the system and provides a User's Manual, Volume 2 is a detailed explanation of the system from the programmer's point of view. Volume 2 assumes the reader is familiar with Volume 1 of this report, and with dBASE III PLUS and Borland TURBO Pascal.

The Texas Flexible Pavement Database System is a microcomputer relational database system written for an IBM XT in dBASE III PLUS and Pascal. The dBASE database contains location, inventory, environmental, pavement condition (monitoring), and traffic information for randomly selected sections of pavement in Texas. Sections average two (2) miles in length. The system is used to edit and update, report, display on the monitor, and analyze the data. A system overview diagram is shown in Figure 1. The following are primary inputs to the system:

a) Pavement Evaluation System (PES) data - used to update the Skid, Serviceability Index, Falling Weight, Visual Rating, and Location Files.
b) Roadway Information (RIFILE) data - used to update the Traffic File.
c) Roadlife maps and District Maintenance records - used to update the Location, Geometric and Shoulder, Layer Identification, Surface, Subgrade, and Layer Thickness Across the Road Files. Unlike the Pes and RIFILE data, this information must be input through data entry screens.

The following are primary outputs of the system:

a) Data Inquiry Screens - any information stored in the database can be viewed on the monitor.
b) Performance versus Accumulated 18 KIP Equivalent Axle Loads Graph - alligator cracking, rutting, or PSI versus 18 KEAL points are plotted on a graph which is displayed on the monitor. A curve is then fit through these points and also displayed on the monitor.
c) Model Development File - this file contains rho and beta constants which describe the shape of the degradation curve mentioned in (b) and independent variables (layer thickness, environmental factors, traffic levels, etc.) which are needed to develop performance models. This file can be used directly by the SAS programming language.
d) Various Reports including the following:
    1) 'Raw' data listings - listings of all information contained in any of the master files and table files
    2) Summary listings - information about a section printed on a single page
    3) Inventory Update Forms - a two-part report in which the first page contains inventory information about

TEXAS FLEXIBLE PAVEMENT DATABASE SYSTEM

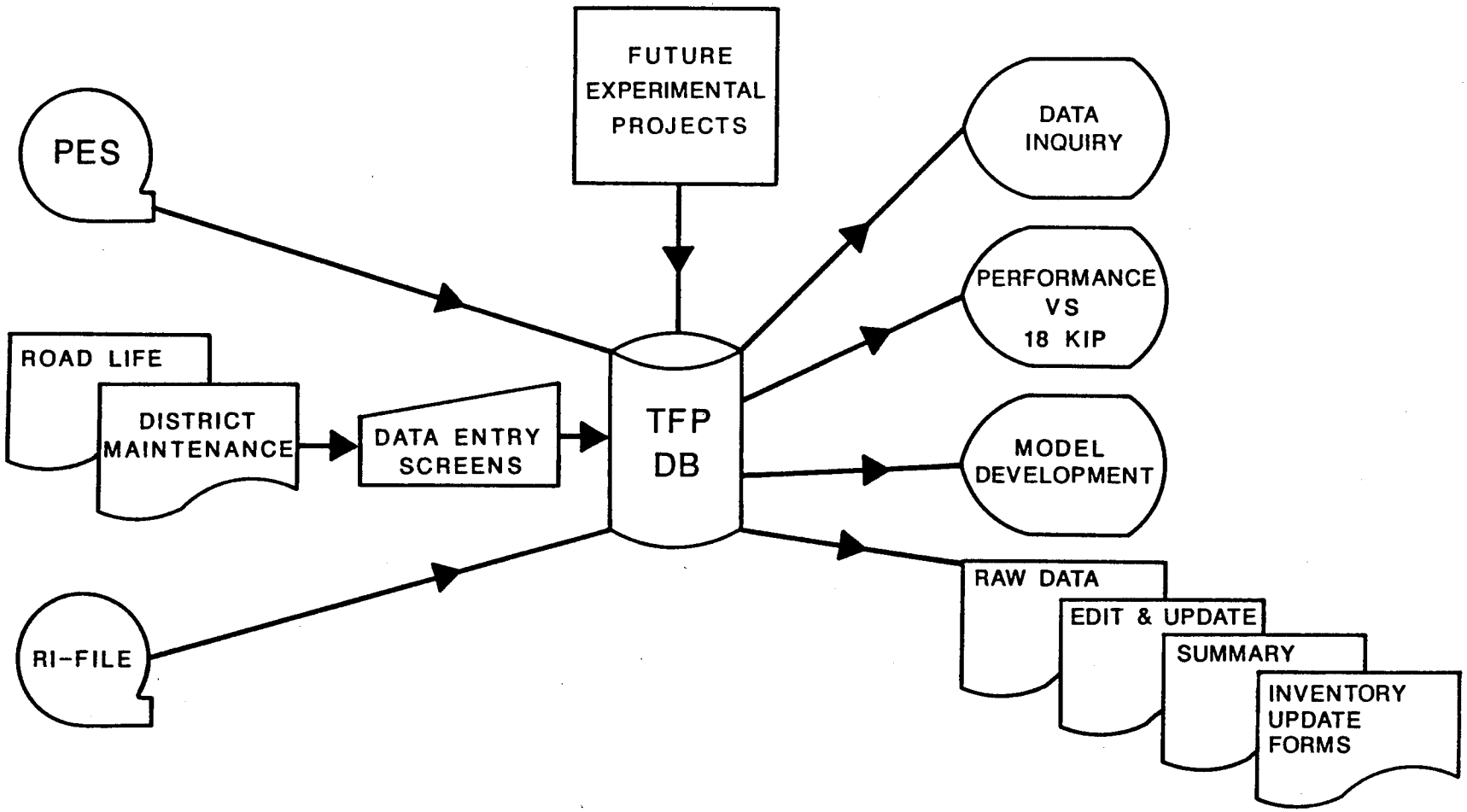


FIGURE 1

a section on one or two sheets, and the second page is a blank form which can be used by the District Offices to record maintenance activities to update inventory data in the database

4) Edit and Update Reports - reports which list errors or inconsistencies in data to be added to or changed in the database

Table 1 lists the relational database master files, the table files, and two primary auxiliary files.

TABLE 1
List of Master Files

dBASE III Name	Type of Information	Data Category
DYNAFLLD	Dynaflect	Monitoring
ENV	Environment	Environmental
FALLWGHT	Falling Weight	Monitoring
GEOSHO	Geometric and Shoulder	Inventory
LAYER	Layer Identification	Inventory
LAYTHICK	Layer Thickness Across the Road	Inventory
LOCATION	Location	Inventory
SI	Serviceability Index	Monitoring
SKID	Skid Measurements	Monitoring
SUBGRADE	Subgrade Layer	Inventory
SURFACE	Surface Layer	Inventory
TRAFFIC	Traffic	Traffic
VISUAL	Visual Rating	Monitoring
WEATHER	Weather	Environmental

List of Table Files

CNTYTBL	County Name
DISTTEMP	District Temperature Constant
FUNCLTBL	Functional Classification Table
LAYERTBL	Layer Description Table
MATLTBL	Material Type Table
PAVETYPE	Pavement Type Table
WIDENFLG	Widening Flag Table

List of Primary Auxiliary Files

DISTRESS	Distress and 18 KIP
MODEL	Model (used to develop performance equations)

Record layouts for the files listed in Table 1 are provided in Appendix A. These files are stored in the subdirectory \PAVEDB\FILES. All dBASE

master files are indexed by the key fields which are indicated on the file layouts. The index files are stored in the subdirectory \PAVEDB\INDEXES. Figure 2 shows the master index structure for the database master files. Appendix B lists in alphabetical order all fields contained in the master files and in which files the fields are found.

The system is composed of a main menu program and five subsystems which are as follows:

a) Inquiry - displays on the monitor master file and table file data
b) Report - produces hard copies of data contained in the files
c) Edit and Update - modifies the data in the database
d) Applications - produces graphs and a model file both of which can be used in the development of performance models
e) Database Maintenance - reindexes the master files, sets up the defaults for the printer and path, and copies the master files onto floppy diskettes for backup purposes

The main menu initiates each subsystem.

This manual is organized in the same order as the subsystems are listed above. Each subsystem is contained in a separate chapter. The following is provided for each subsystem:

General Narratives
Program Flow Diagrams
Program Specifications
Screen Layouts
File Layouts
Program Listings

Sample reports are provided in Chapter 3. File layouts for files listed in Table 1 are not listed in the chapters since they can be found in Appendix A. Table 2 explains the standard file extensions used in the Texas Flexible Pavement System to identify the various types of files.

TABLE 2
Standard File Extensions

Extension	File Type
.BAT	Batch File
.DAT	Data File NOT Identified by Other Extensions
.DBF	dBASE III PLUS File
.DLM	Comma Delimited File
.FMT	dBASE III PLUS Screen Format File
.FRM	dBASE III PLUS Report Form File
.MEM	dBASE III PLUS Memory Variable File
.NDX	dBASE III PLUS Index File
.PAS	TURBO Pascal Program File
.PRG	dBASE III PLUS Program File
.VUE	dBASE III PLUS View File

TEXAS FLEXIBLE PAVEMENT SYSTEM
RELATIONAL DATABASE INDEX STRUCTURE

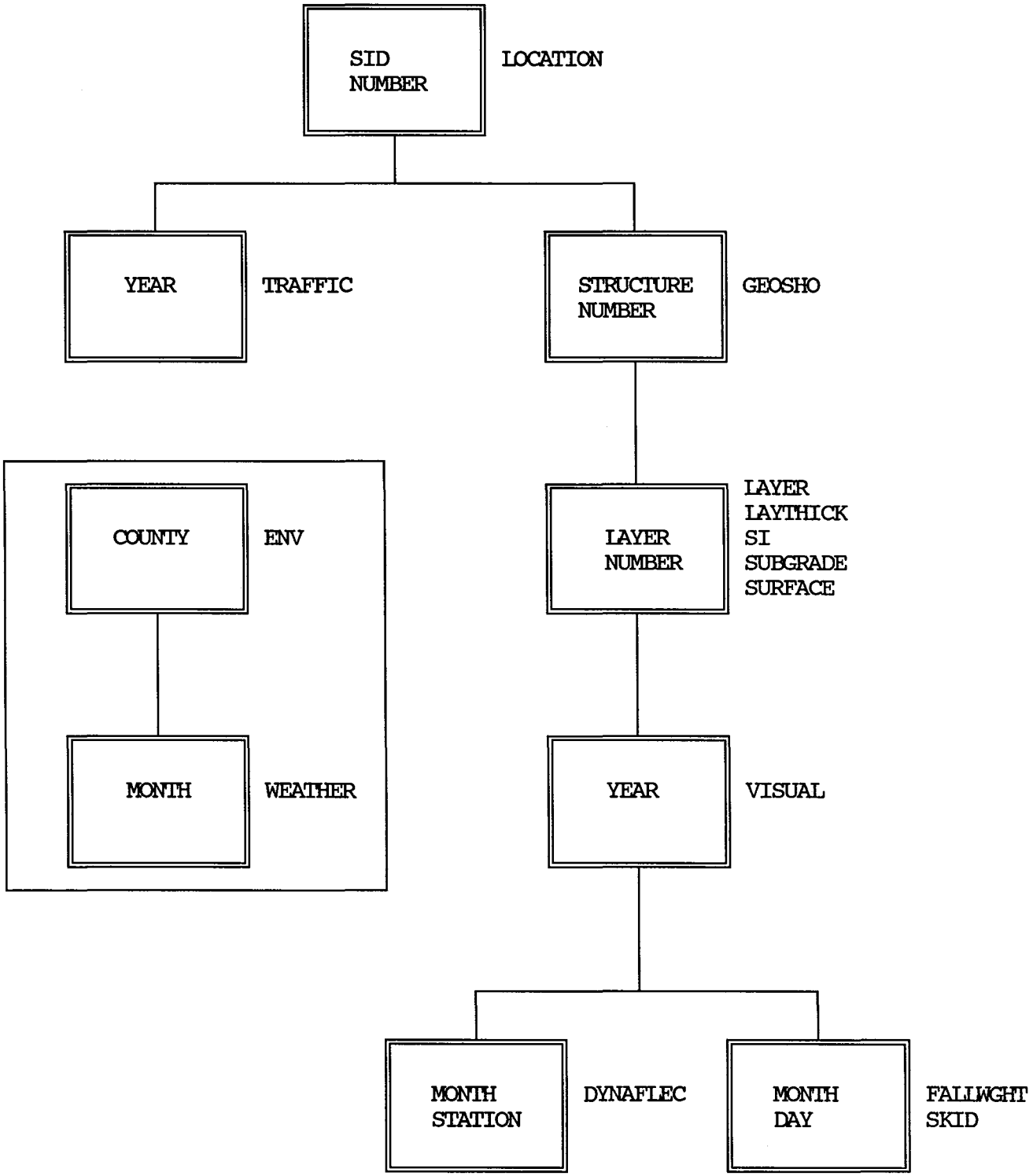


FIGURE 2

Figure 3 illustrates and explains the symbols used in the manual to diagram program flow. Figure 4 shows the subdirectory structure of the system. The minimum system configuration is listed in Appendix C. The initial Texas Flexible Pavement System installation programs are documented in Appendix D.

PROGRAM FLOW DIAGRAM SYMBOLS

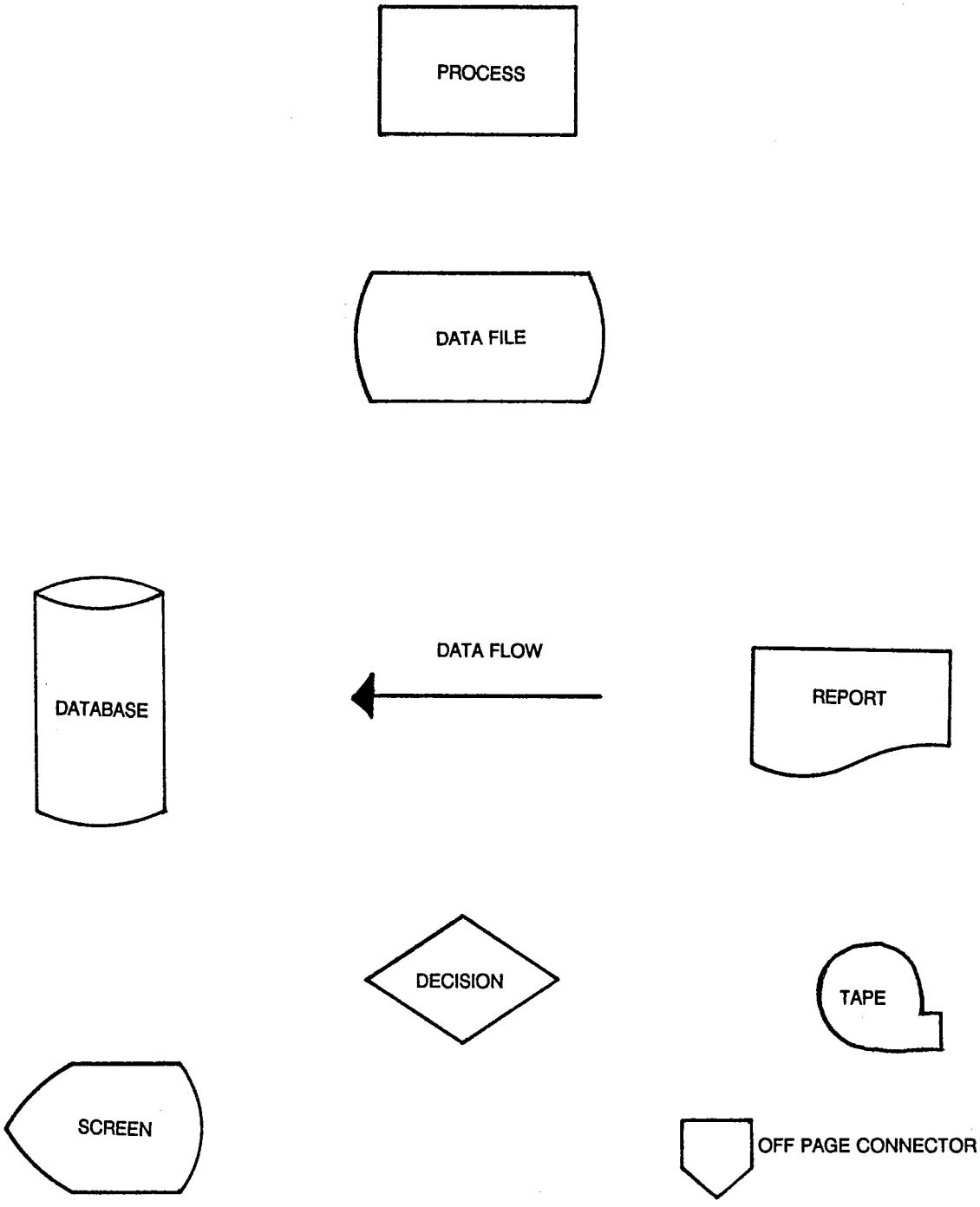


FIGURE 3

SUBDIRECTORY STRUCTURE

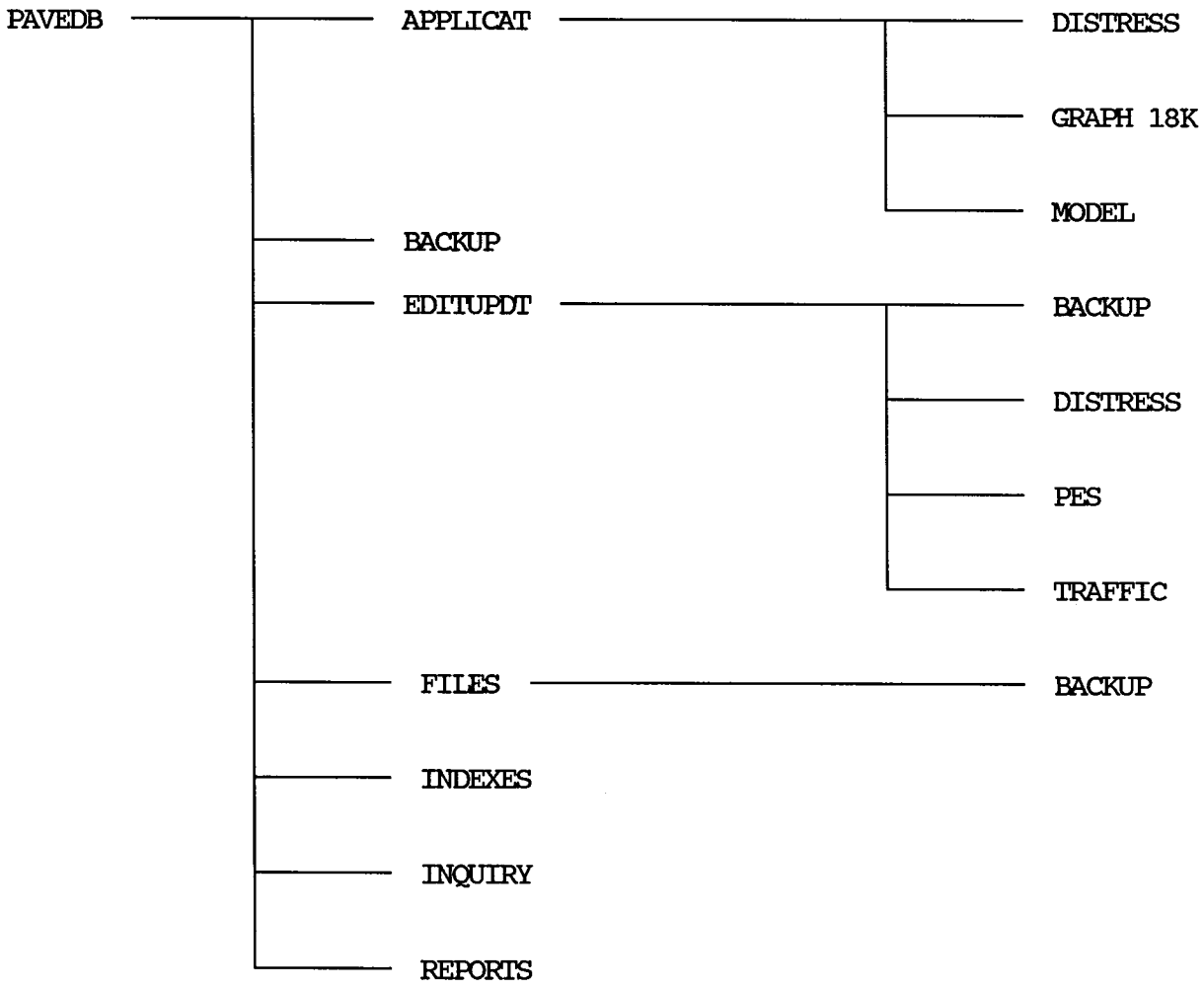


FIGURE 4

CHAPTER 1

MAIN MENU DRIVER PROGRAM

Main Menu Driver Program Narrative

The program DBMAIN.PRG is the first program called by dBASE. The Flexible Pavement System is activated by FLEXPAVE.BAT which consists of the following command: DBASE DBMAIN. This command starts up dBASE and calls the main driver program DBMAIN.PRG.

DBMAIN.PRG checks to see if all the master files are present. It also verifies that the memory variable file INSTALL.MEM with the installation defaults is present. If it is not, the installation menu is run automatically before the main menu is brought up. If the file is present, DBMAIN.PRG displays the main menu on the screen and calls the appropriate program for the choice selected.

DBMAIN.PRG also activates the program ERROR.PRG which is called when an error occurs in the system. These programs and INSTALL.MEM can be found in subdirectory \PAVEDB.

Main Menu Driver - Program Flow Diagram

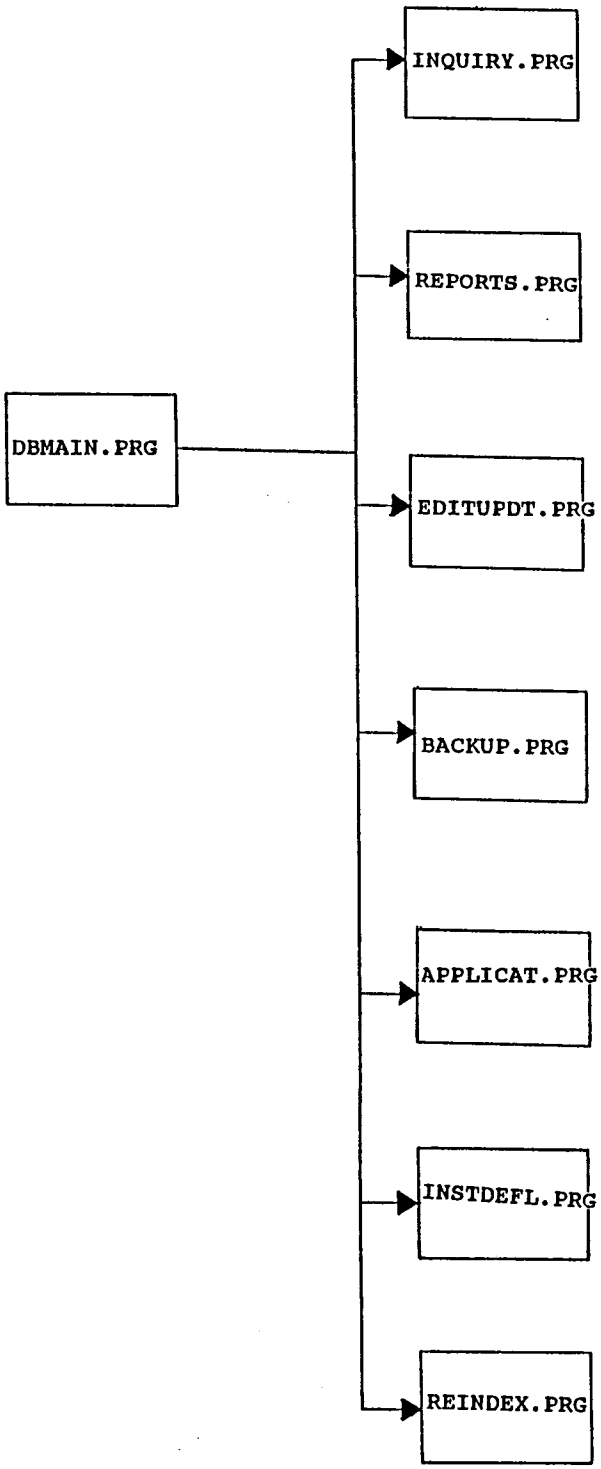


FIGURE 5

PROGRAM SPECIFICATION

Program Name: DBMAIN.PRG

Purpose: To check for the existence of the Master dBASE files, to check for the Installation default file and to call the appropriate programs from the main menu.

Input\Output Files:

INSTALL.MEM

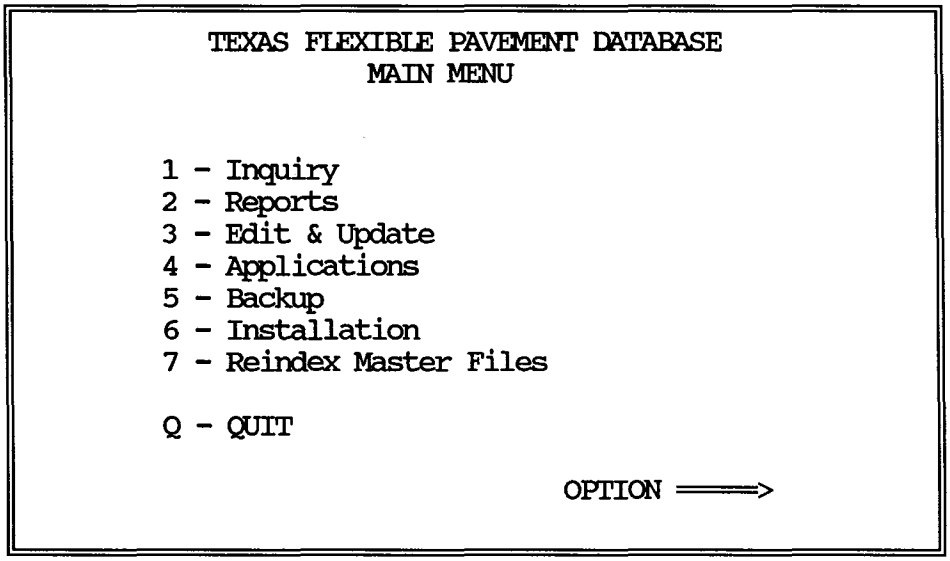
dBASE Programs Called (See Program Flow Diagram):

The following programs called are subsystems by themselves. For more information on each program, refer to the appropriate chapter of this manual.

INQUIRY.PRG
REPORTS.PRG
EDITUPDT.PRG
BACKUP.PRG
APPLICAT.PRG
INSTDEFL.PRG
REINDEX.PRG
ERROR.PRG - This program is called only when an error occurs

MAIN MENU SCREEN

The following screen is produced by DBMAIN.PRG.



```
*
* MAIN PROGRAM DRIVER
* PROGRAM NAME:          DBMAIN.PRG      06/01/88
* MODIFIED ON:           09/19/88
* PROJECT 2456 - TEXAS FLEXIBLE PAVEMENT DATABASE CONVERSION
* TAMU/TTI
* AUTHOR:                TREVOR X. PEREIRA
* PURPOSE:               TO SET UP THE MAIN MENU FOR THE DATABASE
*                        MAINTAINENCE SYSTEM.
*

SET TALK OFF
SET ECHO OFF
SET STAT OFF
SET BELL OFF
SET HELP OFF
SET CONFIRM ON
SET ESCAPE OFF
SET SAFETY ON
SET TYPE TO 20
ON ERROR DO \PAVEDB\ERROR.PRG
CLOSE ALL
PUBLIC MDRIVE, MDRIVE2, MPORT1, MPORT2
SET FORMAT TO
STORE " " TO MOPTION
STORE .T. TO NOTFINI
CLEAR
* Get drive that is being used
IF FILE('\PAVEDB\INSTALL.MEM')
   RESTORE FROM \PAVEDB\INSTALL ADDI
   MDRIVE = IDRIVE
   MDRIVE2 = IDRIVE2
   MPORT1  = IPORT1
   MPORT2  = IPORT2
   RELE ALL LIKE I*
   SET DEFAULT TO &MDRIVE
ELSE
   CLEAR
   @ 10, 5 SAY "INSTALL.MEM File not found.  Please go through the "
   @ 11, 5 SAY "  Installation Menu and choose the defaults."
   WAIT
   IDRIVE = " "
   IDRIVE2 = " "
   IPORT1 = "    "
   IPORT2 = "    "
   DO \PAVEDB\INSTDEFL
ENDIF

? "Checking to see if Master files are present.  Please Wait . . ."
IF .NOT. FILE('\PAVEDB\FILES\LOCATION.DBF')
   ? "LOCATION file is not found.  Please Check . . ."
```

```
   WAIT
   RETURN
ENDIF
IF .NOT. FILE('\PAVEDB\FILES\LAYER.DBF')
   ? "LAYER file is not found.  Please Check . . ."
   WAIT
   RETURN
ENDIF
IF .NOT. FILE('\PAVEDB\FILES\LAYTHICK.DBF')
   ? "LAYTHICK file is not found.  Please Check . . ."
   WAIT
   RETURN
ENDIF
IF .NOT. FILE('\PAVEDB\FILES\GEOSHO.DBF')
   ? "GEOSHO file is not found.  Please Check . . ."
   WAIT
   RETURN
ENDIF
IF .NOT. FILE('\PAVEDB\FILES\SURFACE.DBF')
   ? "SURFACE file is not found.  Please Check . . ."
   WAIT
   RETURN
ENDIF
IF .NOT. FILE('\PAVEDB\FILES\SUBGRADE.DBF')
   ? "SUBGRADE file is not found.  Please Check . . ."
   WAIT
   RETURN
ENDIF
IF .NOT. FILE('\PAVEDB\FILES\VISUAL.DBF')
   ? "VISUAL file is not found.  Please Check . . ."
   WAIT
   RETURN
ENDIF
IF .NOT. FILE('\PAVEDB\FILES\DYNAFLLD.DBF')
   ? "DYNAFLLD file is not found.  Please Check . . ."
   WAIT
   RETURN
ENDIF
IF .NOT. FILE('\PAVEDB\FILES\SI.DBF')
   ? "SI file is not found.  Please Check . . ."
   WAIT
   RETURN
ENDIF
IF .NOT. FILE('\PAVEDB\FILES\TRAFFIC.DBF')
   ? "TRAFFIC file is not found.  Please Check . . ."
   WAIT
   RETURN
ENDIF
IF .NOT. FILE('\PAVEDB\FILES\WEATHER.DBF')
   ? "WEATHER file is not found.  Please Check . . ."
   WAIT
   RETURN
ENDIF
IF .NOT. FILE('\PAVEDB\FILES\FALLWGHT.DBF')
```

```
   ? "FALLWGHT file is not found.  Please Check . . ."
   WAIT
   RETURN
ENDIF
IF .NOT. FILE('\PAVEDB\FILES\ENV.DBF')
   ? "ENV file is not found.  Please Check . . ."
   WAIT
   RETURN
ENDIF
IF .NOT. FILE('\PAVEDB\FILES\SKID.DBF')
   ? "SKID file is not found.  Please Check . . ."
   WAIT
   RETURN
ENDIF
IF .NOT. FILE('\PAVEDB\INDEXES\LOCSID.NDX')
   ? "LOCATION INDEX is not found.  Please Check . . ."
   WAIT
   RETURN
ENDIF
IF .NOT. FILE('\PAVEDB\INDEXES\LAYNDX.NDX')
   ? "LAYER INDEX is not found.  Please Check . . ."
   WAIT
   RETURN
ENDIF
IF .NOT. FILE('\PAVEDB\INDEXES\LAYTNDX.NDX')
   ? "LAYTHICK INDEX is not found.  Please Check . . ."
   WAIT
   RETURN
ENDIF
IF .NOT. FILE('\PAVEDB\INDEXES\GEONDX.NDX')
   ? "GEOSHO INDEX is not found.  Please Check . . ."
   WAIT
   RETURN
ENDIF
IF .NOT. FILE('\PAVEDB\INDEXES\SURFNDX.NDX')
   ? "SURFACE INDEX is not found.  Please Check . . ."
   WAIT
   RETURN
ENDIF
IF .NOT. FILE('\PAVEDB\INDEXES\SUBGNDX.NDX')
   ? "SUBGRADE INDEX is not found.  Please Check . . ."
   WAIT
   RETURN
ENDIF
IF .NOT. FILE('\PAVEDB\INDEXES\VISUAL.NDX')
   ? "VISUAL INDEX is not found.  Please Check . . ."
   WAIT
   RETURN
ENDIF
IF .NOT. FILE('\PAVEDB\INDEXES\DYNAFLLD.NDX')
   ? "DYNAFLLD INDEX is not found.  Please Check . . ."
   WAIT
   RETURN
ENDIF
```

```
IF .NOT. FILE('\PAVEDB\INDEXES\SI.NDX')
   ? "SI INDEX is not found.  Please Check . . ."
   WAIT
   RETURN
ENDIF
IF .NOT. FILE('\PAVEDB\INDEXES\TRAFFIC.NDX')
   ? "TRAFFIC INDEX is not found.  Please Check . . ."
   WAIT
   RETURN
ENDIF
IF .NOT. FILE('\PAVEDB\INDEXES\WEATHER.NDX')
   ? "WEATHER INDEX is not found.  Please Check . . ."
   WAIT
   RETURN
ENDIF
IF .NOT. FILE('\PAVEDB\INDEXES\FALLWGHT.NDX')
   ? "FALLWGHT INDEX is not found.  Please Check . . ."
   WAIT
   RETURN
ENDIF
IF .NOT. FILE('\PAVEDB\INDEXES\ENV.NDX')
   ? "ENV INDEX is not found.  Please Check . . ."
   WAIT
   RETURN
ENDIF
IF .NOT. FILE('\PAVEDB\INDEXES\SKID.NDX')
   ? "SKID INDEX is not found.  Please Check . . ."
   WAIT
   RETURN
ENDIF

DO WHILE NOTFINI
   STORE " " TO MOPTION
   CLOSE ALL
   * set up the database maintainence main menu screen and do the
   * loop until MOPTION is 1-7 or Q
   DO WHILE .NOT. (MOPTION $ '1234567Q')
       @  0,  0  CLEAR
       @  3, 15  SAY "       TEXAS FLEXIBLE PAVEMENT DATABASE"
       @  4, 32  SAY "MAIN MENU"
       @  7, 19  SAY "1 - Inquiry"
       @  8, 19  SAY "2 - Reports"
       @  9, 19  SAY "3 - Edit & Update"
       @ 10, 19  SAY "4 - Applications"
       @ 11, 19  SAY "5 - Backup"
       @ 12, 19  SAY "6 - Installation"
       @ 13, 19  SAY "7 - Reindex Master Files"
       @ 15, 19  SAY "Q - QUIT"
       @ 17, 44  SAY "OPTION ====> "  GET MOPTION PICTURE "!"
       @  2,  9  TO 19, 67    DOUBLE
       READ
       * check for MOPTION equal to 1 - 7 or Q
       CLEAR TYPE
       IF .NOT. (MOPTION $ '1234567Q')
```

```
            @ 20, 10 SAY "Please enter 1, 2, 3, 4, 5, 6, 7 or  Q"
            WAIT
         ENDIF
   ENDDO
   DO CASE
     CASE MOPTION = "1"
        DO \PAVEDB\INQUIRY\INQUIRY
     CASE MOPTION = "2"
        DO \PAVEDB\REPORTS\REPORTS
     CASE MOPTION = "3"
        DO \PAVEDB\EDITUPDT\EDITUPDT
     CASE MOPTION = "4"
        DO \PAVEDB\APPLICAT\APPLICAT
     CASE MOPTION = "5"
        DO \PAVEDB\BACKUP\BACKUP
     CASE MOPTION = "6"
        DO \PAVEDB\INSTDEFL
     CASE MOPTION = "7"
        MRESPONSE = " "
        @ 10,10 SAY "You are about to REINDEX all the files - takes about 2
hours."
        @ 11,11 SAY "Want to continue (Y/N)? " GET MRESPONSE
        READ
        IF MRESPONSE = "Y"
           DO \PAVEDB\REINDEX
        ENDIF
     CASE MOPTION = "Q"
        QUIT
   ENDCASE
ENDDO
```

PROGRAM LISTING

```
*
* PROGRAM NAME:         ERROR.PRG      07/03/88
* PROJECT 2456 - TEXAS FLEXIBLE PAVEMENT DATABASE CONVERSION
* TAMU/TTI
* AUTHOR:               TREVOR X. PEREIRA
* PURPOSE:              TO GIVE A MESSAGE IF AN ERROR OCCURS ANY WHERE IN
*                       THE SYSTEM
*

CLEAR TYPE
SET TYPE TO 0
CLEAR
CLOSE ALL
@ 10, 10 SAY "          An error has occured in the system."
@ 12, 10 SAY "      Please contact The Texas Transportation Institute."
@ 14, 10 SAY "                        at (409) 845-8408"
@ 15, 10 SAY "          Ask for:  Rebecca Yette or"
@ 16, 10 SAY "                        Trevor Pereira"
CANCEL
```

CHAPTER 2

INQUIRY SUBSYSTEM

GENERAL NARRATIVE

The Inquiry Menu System which is written in dBase III Plus allows the user to view the data in any of the 14 master files and the 7 tables. Changes to the data cannot be made using this part of the system. The Inquiry main menu has the choices Inventory, Monitoring, Traffic, Environment and Tables available. Depending on the file chosen, a dBase form is displayed on the screen with the required information for a particular SID Number. The user can scan through subsequent records or exit the screen.

The program INQUIRY.PRG displays the appropriate menus on the screen and the user chooses the file and SID number he wants to view. The program then uses the appropriate dBase screen format file (form) to display the data. dBase screen format files use the .FMT extension as a naming convention. INQUIRY.PRG and all .FMT files are stored in the subdirectory \PAVEDB\INQUIRY.

This chapter contains the following information about the Inquiry Subsystem:

- Program Flow Diagram
- Program Specification
- Inquiry Screens
- Screen Format Program Listings
- Program Listings.

Inquiry Subsystem - Program Flow Diagram

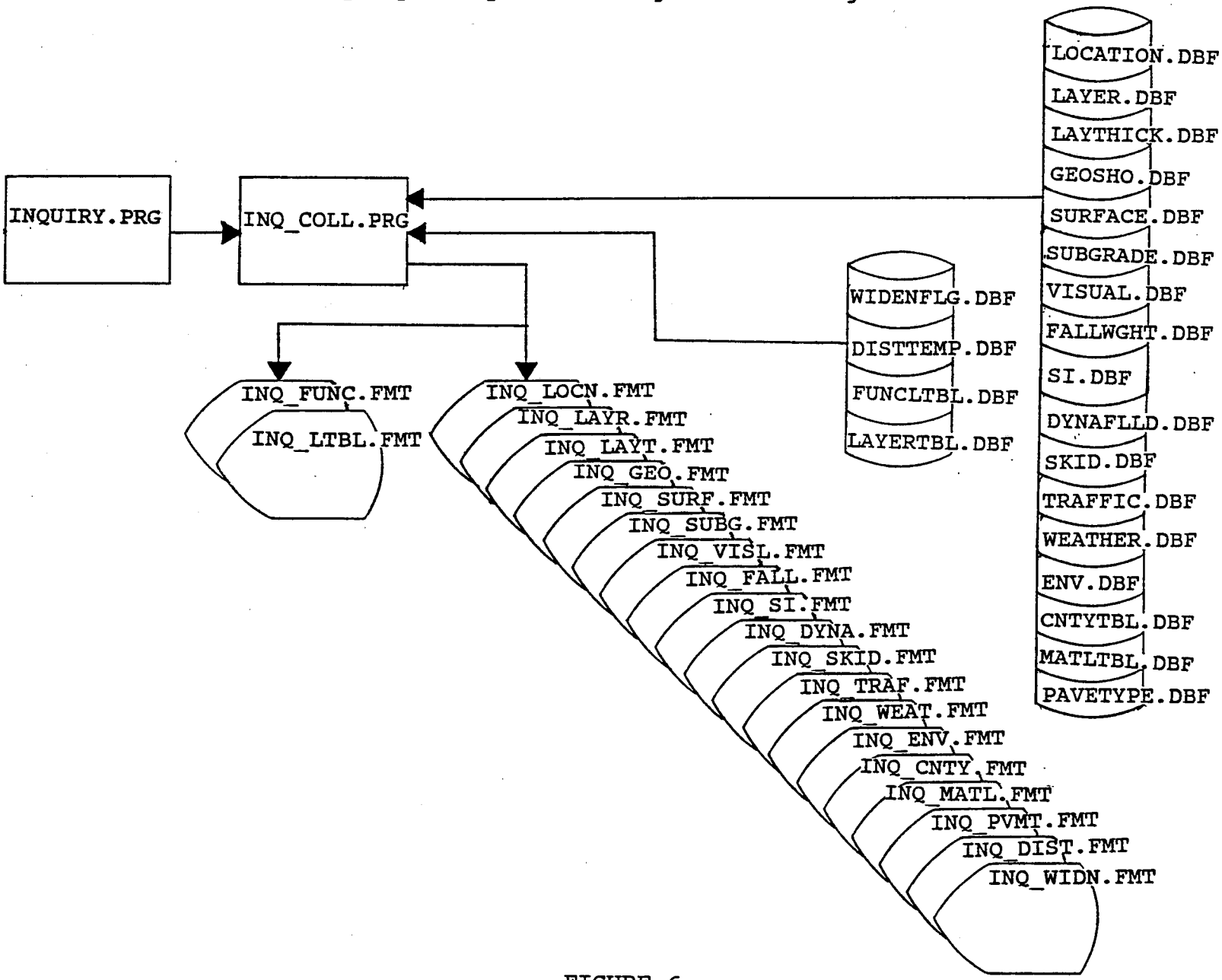


FIGURE 6

PROGRAM SPECIFICATION

Program Name: INQUIRY.PRG

Purpose: To display the data to the screen for the appropriate files and tables on user's request.

Input Files: The following files are used along with their indices:

	Data File	Index File
Location	LOCATION.DBF	LOCSID.NDX
Layer Identification	LAYER.DBF	LAYNDX.NDX
Geometric & Shoulder	GEOSHO.DBF	GEONDX.NDX
Surface	SURFACE.DBF	SURFNDX.NDX
Subgrade	SUBGRADE.DBF	SUBGNDX.NDX
Layer Thickness	LAYTHICK.DBF	LAYTNDX.NDX
Visual Rating	VISUAL.DBF	VISUAL.NDX
Serviceability Index	SI.DBF	SI.NDX
Falling Weight SSI	FALLWGHT.DBF	FALLWGHT.NDX
Dynaflect Measure	DYNAFLLD.DBF	DYNAFLLD.NDX
Skid Measurement	SKID.DBF	SKID.NDX
Environment	ENV.DBF	ENV.NDX
Weather	WEATHER.DBF	WEATHER.NDX
Traffic	TRAFFIC.DBF	TRAFFIC.NDX
County Name Table	CNTYTBL.DBF	CTYTBLNO.NDX
Material Type Table	MATLTBL.DBF	
Pavement Type Table	PAVETYPE.DBF	
District Temperature	DISTTEMP.DBF	
Widening Flag Table	WIDENFLG.DBF	
Layer Description	LAYERTBL.DBF	
Functional Classification	FUNCLTBL.DBF	

dBase Procedure File Called:

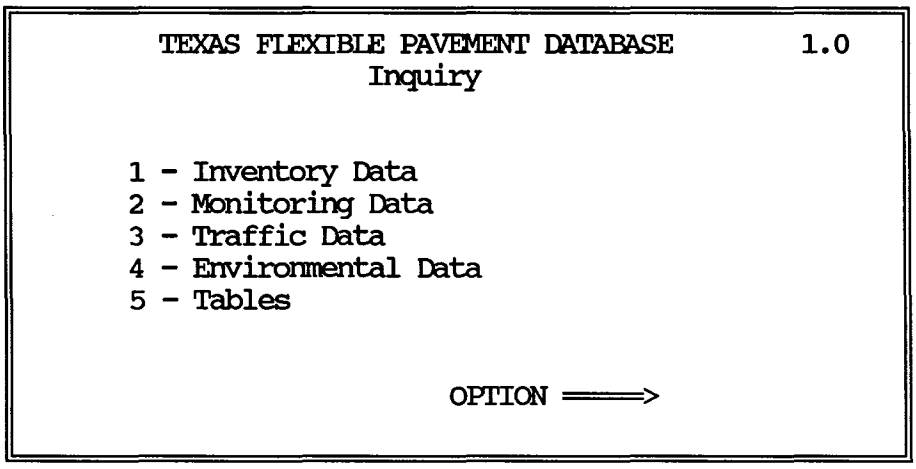
INQ_COLL.PRG - This procedure displays on the screen data for individual files.

Output (Using dBASE III Format Files):

The following format files (forms) are used to display data to the screen:

		Format File
Inventory Data	- Location	INQ_LOCN.FMT
	- Layer Identification	INQ_LAYR.FMT
	- Layer Thickness	INQ_LAYT.FMT
	- Geometric & Shoulder	INQ_GEO.FMT
	- Surface	INQ_SURF.FMT
	- Subgrade	INQ_SUBG.FMT
Monitoring Data	- Visual	INQ_VISL.FMT
	- Serviceability Index	INQ_SI.FMT
	- Falling Weight	INQ_FALL.FMT
	- Dynaflect	INQ_DYNA.FMT
	- Skid	INQ_SKID.FMT
Environmental Data	- Environment	INQ_ENV.FMT
	- Weather	INQ_WEAT.FMT
Traffic Data	- Traffic	INQ_TRAF.FMT
Tables	- County Name	INQ_CNTY.FMT
	- Material Type Class.	INQ_MATL.FMT
	- Type of Pavement	INQ_PVMT.FMT
	- District Temperature	INQ_DIST.FMT
	- Widening Flag	INQ_WIDN.FMT
	- Layer Description	INQ_LYBL.FMT
	- Functional Class.	INQ_FUNC.FMT

INQUIRY SCREENS



Main Menu of the Inquiry Subsystem

```
        TEXAS FLEXIBLE PAVEMENT DATABASE      1.1
                    INQUIRY
                 Inventory Data


    1 - Location
    2 - Layer ID
    3 - Geometric & Shoulder
    4 - Surface
    5 - Subgrade
    6 - Layer Thickness Across the Road


                          OPTION ===========>
```

Inventory **Menu** Screen

```
                TEXAS FLEXIBLE PAVEMENT DATBASE
                    INQUIRY - INVENTORY
                       Location File

SID Number          13          District 1  County  92

Highway Ident. US   82          Control/Section   45/ 4

Mile Post  22 + 0  TO 24 + 0    Lane Identification R

Mile Point 22.000 TO 24.000     Mile Point Date  6/75

HPMS Sample Number              HPMS Section Subdivision 0

Functional Classification  0    Number of Lanes   1

Active ? T  Inactive Date 0/ 0  Previous SID  0    Next SID  0

Comment
```

Inventory - Location File **Data** Screen Display

TEXAS FLEXIBLE PAVEMENT DATABASE
INQUIRY
Inventory Data - Layer Identification

SID Number	13
Structure Number	1
Layer Number	1
Layer Description	7
Center Thickness (Inches)	
Layer Material Classification	44
Job Completed Date (MM/YY)	6/31
Widening Date (MM/YY)	0/0

Inventory - Layer File **Data** Screen Display

TEXAS FLEXIBLE PAVEMENT DATABASE
INQUIRY
Inventory Data - Geometric And Shoulder

SID Number	13
Structure Number	1
Pavement Type	34
Lane Width one way - (ft)	12.0
Outside Shoulder Width (ft)	0.0
Shoulder Surface Type	1
Shoulder Base Type	0
Shoulder Surface Thick (in.)	0.00
Shoulder Base Thick (in.)	0.00
Widening Flag	1

Inventory - Geometric & Shoulder File **Data** Screen Display

TEXAS FLEXIBLE PAVEMENT DATABASE
INQUIRY
Inventory Data - Surface

SID Number	13
Structure Number	1
Layer Number	3
Aggregate Application Rate (S.Y./C.Y.)	0
Admixture Type	AC
Admixture Percent (%)	5.70
Asphalt Application Rate (Gal/S.Y.)	0.00

Inventory - Surface File Data Screen Display

TEXAS FLEXIBLE PAVEMENT DATABASE
INQUIRY
Inventory Data - Subgrade

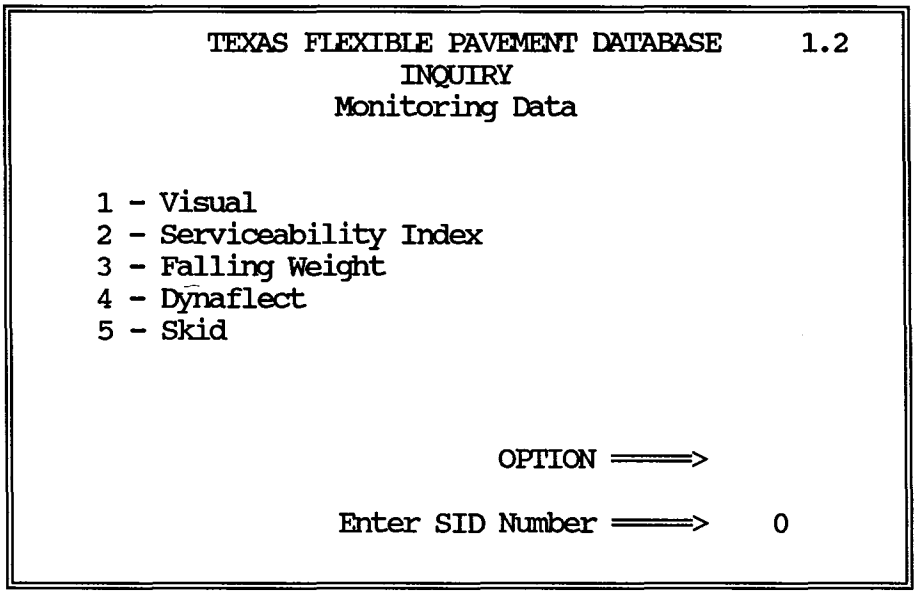
SID Number	13
Structure Number	1
Layer Number	1
Percent Passing No. 200 Sieve (%)	88.8
Plasticity Index	40.4
Liquid Limit	64.5
Texas Triaxial Class	5.3
Permeability Index	0.23

Inventory - Subgrade File Data Screen Display

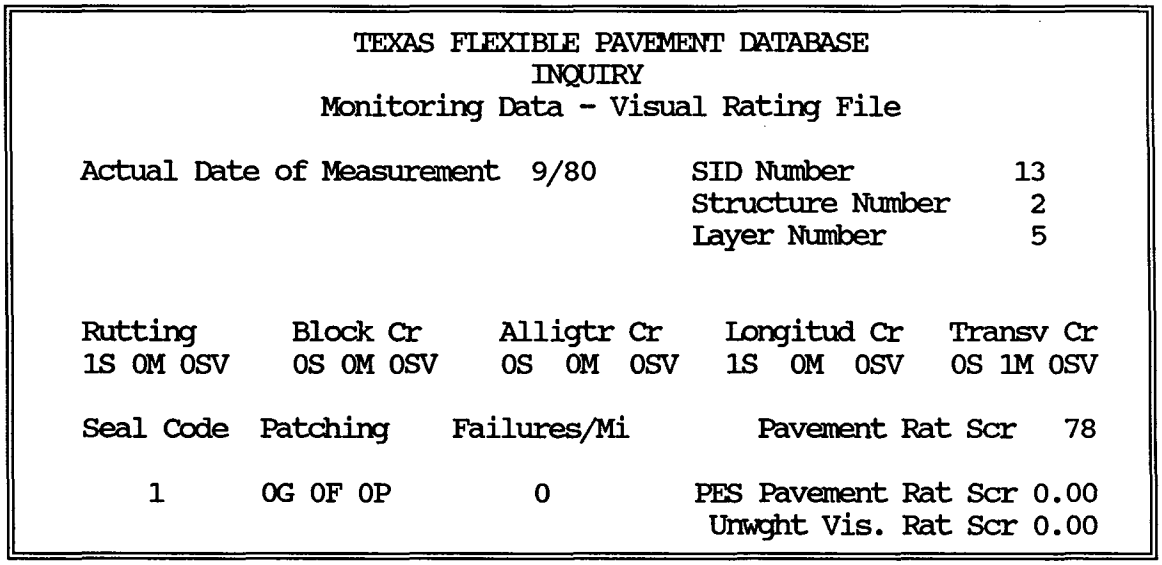
TEXAS FLEXIBLE PAVEMENT DATABASE
INQUIRY
Inventory Data - Layer Thickness Across the Road

SID Number	13
Structure Number	1
Layer Number	2
Thickness - 3rd Pos From Center (in.)	10.00
Thickness - 2nd Pos From Center (in.)	10.00
Thickness - 1st Pos From Center (in.)	6.00
Thickness - Center (in.)	6.00
Distance From Center - 3rd Position (ft.)	12.00
Distance From Center - 2nd Position (ft.)	9.0
Distance From Center - 1st Position (ft.)	5.0

Inventory - Layer Thickness Across The Road File Data Screen Display



Monitoring **Menu** Screen



Monitoring - Visual File **Data** Screen Display

TEXAS FLEXIBLE PAVEMENT DATABASE
INQUIRY
Monitoring Data - Serviceability Index

SID Number	13
Structure Number	2
Layer Number	5
Date	9/16/80
Count of Observation	8
Mean	3.17500
Standard Deviation	0.38079
Low Value	2.5
High Value	3.6

Monitoring - Serviceability Index File **Data** Screen Display

TEXAS FLEXIBLE PAVEMENT DATABASE
INQUIRY
Monitoring Data - Falling Weight SSI

Date 0/ 0/ 0 (MM/DD/YY)
SID Number 26
Structure Number 1
Layer Number 5
Average SSI 0.0
Temperature 0

	Reading 1	Reading 2	Reading 3	Reading 4	Reading 5
Geophone 1	0.00	0.00	0.00	0.00	0.00
Geophone 2	0.00	0.00	0.00	0.00	0.00
Geophone 3	0.00	0.00	0.00	0.00	0.00
Geophone 4	0.00	0.00	0.00	0.00	0.00
Geophone 5	0.00	0.00	0.00	0.00	0.00
Geophone 6	0.00	0.00	0.00	0.00	0.00
Geophone 7	0.00	0.00	0.00	0.00	0.00

Monitoring - Falling Weight SSI File **Data** Screen Display

TEXAS FLEXIBLE PAVEMENT DATABASE
INQUIRY
Monitoring Data - Dynaflect Measurement

SID Number	13
Structure Number	1
Layer Number	4
Date	8/10/76

STATION	1
Reading for Sensor 1	0.570
Reading for Sensor 2	0.540
Reading for Sensor 3	0.470
Reading for Sensor 4	0.380
Reading for Sensor 5	0.320

Monitoring - Dynaflect Measurement File **Data** Screen Display

TEXAS FLEXIBLE PAVEMENT DATABASE
INQUIRY
Monitoring Data - Skid Measurement

SID Number	13
Structure Number	1
Layer Number	4
Date	4/71

Mean	38
High	42
Low	32

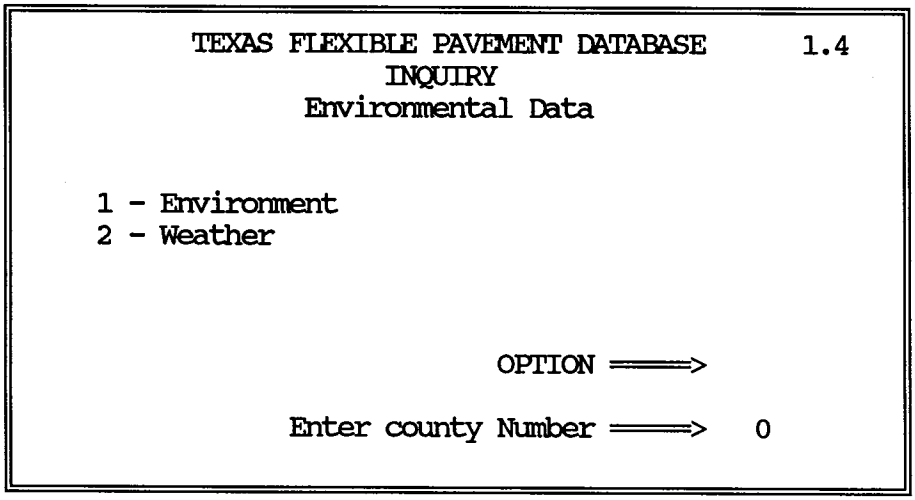
Monitoring - Skid Measurement File **Data** Screen Display

```
TEXAS FLEXIBLE PAVEMENT DATABASE
INQUIRY
Traffic Data

Please Enter SID Number:    0
```

Traffic - Traffic File Screen to Enter the SID number

```
TEXAS FLEXIBLE PAVEMENT DATABASE
INQUIRY
Traffic Data

                SID Number   13
                Year       1933

Annual Average Daily Traffic        386
Annual Cummulative 18 Keal
     - one way                    54259
Percent trucks                     17.7
```

Traffic - Traffic File Data Screen Display



Environment **Menu** Screen

TEXAS FLEXIBLE PAVEMENT DATABASE
INQUIRY
Weather Measurement

County Number 1
Month 1

	No. of Yrs Avg	Mean	Std. Dev.
Precipitation	19	3.179	2.053
Total Freeze Thaw Cycle	18	9.111	3.160
Wet Freeze Thaw Cycle	18	1.889	1.183
Maximum Temperature	18	55.611	3.791
Averaged Temperature	18	45.056	3.369

Environment - Weather Measurement File **Data** Screen Display

TEXAS FLEXIBLE PAVEMENT DATABASE
INQUIRY
Environment Measurement

County Number	1
Thornthwaite Index Mean	12.510
Thornthwaite Index - No. of Years Averaged	20
Thornthwaite Index - Standard Deviation	26.102

Environment - Environment Measurement File **Data** Screen Display

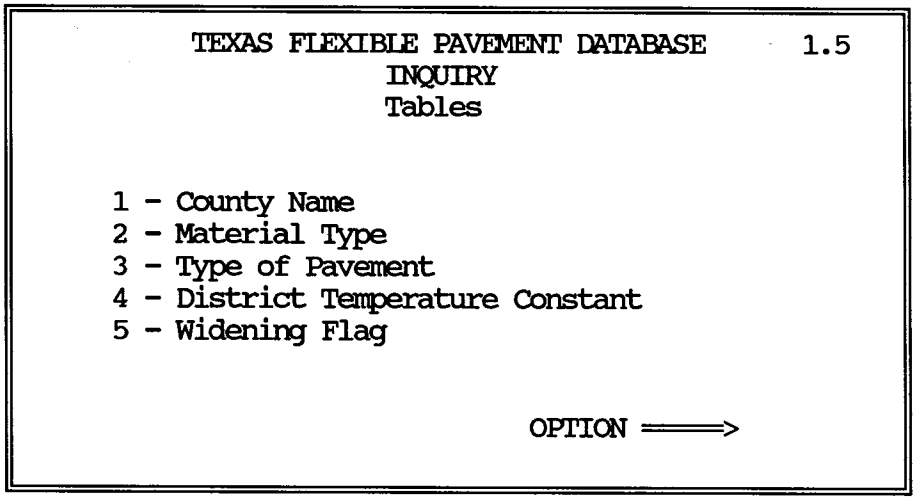


Table **Menu** Screen

```
TEXAS FLEXIBLE PAVEMENT DATABASE
INQUIRY
County Table

County Number     1
County Name       ANDERSON
```

Tables - County Name File **Data** Screen Display

```
TEXAS FLEXIBLE PAVEMENT DATABASE
INQUIRY
Material Type Classification Table

Material Code          1
Material Description   HOT MIX - HOT LAID
Material Short Form    HMAC
Layer Description      S
```

Tables - Material Type Classification File **Data** Screen Display

```
TEXAS FLEXIBLE PAVEMENT DATABASE
INQUIRY
Type of Pavement Table

Pavement Code     1
Type of Base          GRANULAR BASE
Surface Thickness     SURFACE TREATED
Surface Seal
```

Tables - Type of Pavement File **Data** Screen Display

TEXAS FLEXIBLE PAVEMENT DATABASE
INQUIRY
District Temperature Table

District Number 1
Temperature Constant 21

Tables - District Temperature File **Data** Screen Display

TEXAS FLEXIBLE PAVEMENT DATABASE
INQUIRY
Widening Table

Code 0
Description No Widening
Comments:
Center Thickness CAN be used

Tables - Widening Flag File **Data** Screen Display

TEXAS FLEXIBLE PAVEMENT DATABASE
INQUIRY
Layer Description Table

Layer Code 1
Code Description OVLY
Four Letter Code Overlay

Tables - Layer Description File **Data** Screen Display

Format Programs For Screens
Location

```
@  2, 23  SAY "TEXAS FLEXIBLE PAVEMENT DATABASE"
@  3, 23  SAY "            INQUIRY"
@  4, 23  SAY " Inventory Data - Location File"
@  6,  6  SAY "SID Number"
@  6, 18  SAY  LOCATION->SID_NO PICTURE "9999"
@  6, 44  SAY "District"
@  6, 53  SAY  LOCATION->HWYDIST PICTURE "99"
@  6, 59  SAY "County"
@  6, 66  SAY  LOCATION->CNTYNUM PICTURE "999"
@  8,  6  SAY "Highway Ident."
@  8, 21  SAY  LOCATION->HWYPREFX
@  8, 24  SAY  LOCATION->HWYNUM PICTURE "9999"
@  8, 29  SAY  LOCATION->HWYSUFFX
@  8, 44  SAY "Control/Section"
@  8, 60  SAY  LOCATION->CONTROL PICTURE "9999"
@  8, 64  SAY "/"
@  8, 65  SAY  LOCATION->SECTION PICTURE "99"
@ 10,  6  SAY "Mile Post"
@ 10, 17  SAY  LOCATION->BEGMPST PICTURE "999"
@ 10, 20  SAY  LOCATION->BDISSIGN
@ 10, 21  SAY  IIF(LEN(LTRIM(STR(BMPSTDIS,2))) = 1,"0" +
LTRIM(STR(BMPSTDIS,2)), STR(BMPSTDIS,2))
@ 10, 24  SAY "TO"
@ 10, 27  SAY  LOCATION->ENDMPST PICTURE "999"
@ 10, 30  SAY  LOCATION->EDISSIGN
@ 10, 31  SAY  IIF(LEN(LTRIM(STR(EMPSTDIS,2))) = 1,"0" + LTRIM(STR
(EMPSTDIS,2)), STR(EMPSTDIS,2))
@ 10, 44  SAY "Lane Identification"
@ 10, 64  SAY  LOCATION->LANEID
@ 12,  6  SAY "Mile Point"
@ 12, 17  SAY  LOCATION->BEGMPNT PICTURE "99.999"
@ 12, 24  SAY "TO"
@ 12, 27  SAY  LOCATION->ENDMPNT PICTURE "99.999"
@ 12, 44  SAY "Mile Point Date"
@ 12, 61  SAY  LOCATION->MPNTMO PICTURE "99"
@ 12, 63  SAY "/"
@ 12, 64  SAY  LOCATION->MPNTYR PICTURE "99"
@ 14,  6  SAY "HPMS Sample Number"
@ 14, 25  SAY  LOCATION->HPMSSAM
@ 14, 44  SAY "HPMS Section Subdivision"
@ 14, 69  SAY  LOCATION->HPMSSEC PICTURE "9"
@ 16,  6  SAY "Functional Classification"
@ 16, 31  SAY  LOCATION->FUNCLAS PICTURE "99"
@ 16, 44  SAY "Number of Lanes"
@ 16, 61  SAY  LOCATION->NUMLANES PICTURE "99"
@ 18,  6  SAY "Active ?"
@ 18, 15  SAY  LOCATION->ACTVFLAG PICTURE "L"
@ 18, 19  SAY "Inactive Date"
@ 18, 33  SAY  LOCATION->INACIMO  PICTURE "99"
```

```
@ 18, 35  SAY "/"
@ 18, 36  SAY  LOCATION->INACTYR PICTURE "99"
@ 18, 41  SAY "Previous SID"
@ 18, 54  SAY  LOCATION->PREVSID PICTURE "9999"
@ 18, 61  SAY "Next SID"
@ 18, 70  SAY  LOCATION->NEXTSID PICTURE "9999"
@ 20,  6  SAY "Comment"
@ 20, 15  SAY  LOCATION->COMMENT PICTURE
"XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX"
@  1,  2  TO 21, 77    DOUBLE
```

Format Programs For Screens
Layer Identification

```
@  3, 25  SAY "TEXAS FLEXIBLE PAVEMENT DATABASE"
@  4, 37  SAY "INQUIRY"
@  5, 22  SAY "Inventory Data - Layer Identification"
@  8, 37  SAY "SID Number"
@  8, 55  SAY  LAYER->SID_NO
@  9, 37  SAY "Structure Number"
@  9, 57  SAY  LAYER->STRUCNUM
@ 10, 37  SAY "Layer Number"
@ 10, 57  SAY  LAYER->LAYNUM
@ 12, 23  SAY "Layer Description"
@ 12, 57  SAY  LAYER->LAYDESC  FUNCTION "Z"
@ 13, 23  SAY "Center Thickness (inches)"
@ 13, 54  SAY  LAYER->CENTTHK  FUNCTION "Z"
@ 14, 23  SAY "Layer Material Classification"
@ 14, 57  SAY  LAYER->LAYMATCL  FUNCTION "Z"
@ 15, 23  SAY "Job Completed Date (MM/YY)"
@ 15, 54  SAY  LAYER->JOBCMPMO  FUNCTION "Z"
@ 15, 56  SAY "/"
@ 15, 57  SAY  LAYER->JOBCMPYR  FUNCTION "Z"
@ 16, 23  SAY "Widening Date (MM/YY)"
@ 16, 54  SAY  LAYER->WIDENLMO
@ 16, 56  SAY "/"
@ 16, 57  SAY  LAYER->WIDENLYR  FUNCTION "B"
@  2, 15  TO 18, 65    DOUBLE
```

Format Programs For Screens
Layer Thickness Across The Road

```
@  3, 24  SAY "TEXAS FLEXIBLE PAVEMENT DATABASE"
@  4, 35  SAY "INQUIRY"
@  5, 15  SAY "Inventory Data - Layer Thickness Across The Road"
@  8, 36  SAY "SID Number"
@  8, 58  SAY  LAYTHICK->SID_NO
@  9, 36  SAY "Structure Number"
@  9, 60  SAY  LAYTHICK->STRUCNUM
@ 10, 36  SAY "Layer Number"
@ 10, 60  SAY  LAYTHICK->LAYNUM
@ 12, 14  SAY "Thickness - 3rd Pos. From Center (in.)"
@ 12, 57  SAY  LAYTHICK->FC3THK
@ 13, 14  SAY "Thickness - 2nd Pos. From Center (in.)"
@ 13, 57  SAY  LAYTHICK->FC2THK
@ 14, 14  SAY "Thickness - 1st Pos. From Center (in.)"
@ 14, 57  SAY  LAYTHICK->FC1THK
@ 15, 14  SAY "Thickness - Center (in.)"
@ 15, 57  SAY  LAYTHICK->CENTTHK
@ 16, 14  SAY "Distance From Center - 3rd Position (ft)"
@ 16, 58  SAY  LAYTHICK->FC3DIS
@ 17, 14  SAY "Distance From Center - 2nd Position (ft)"
@ 17, 58  SAY  LAYTHICK->FC2DIS
@ 18, 14  SAY "Distance From Center - 1st Position (ft)"
@ 18, 58  SAY  LAYTHICK->FC1DIS
@  2,  9  TO 20, 68    DOUBLE
```

Format Programs For Screens
Geometric & Shoulder Information

```
@  3, 24  SAY "TEXAS FLEXIBLE PAVEMENT DATABASE"
@  4, 35  SAY "INQUIRY"
@  5, 20  SAY "Inventory Data - Geometric And Shoulder"
@  8, 36  SAY "SID Number"
@  8, 54  SAY  GEOSHO->SID_NO
@  9, 36  SAY "Structure Number"
@  9, 56  SAY  GEOSHO->STRUCNUM
@ 11, 20  SAY "Pavement Type"
@ 11, 52  SAY  GEOSHO->PAVETYP
@ 12, 20  SAY "Lane Width - one way (ft)"
@ 12, 50  SAY  GEOSHO->LANEWID
@ 13, 20  SAY "Outside Shoulder Width (ft)"
@ 13, 50  SAY  GEOSHO->OUTSHOWD
@ 14, 20  SAY "Shoulder Surface Type"
@ 14, 53  SAY  GEOSHO->SHOSFTYP
@ 15, 20  SAY "Shoulder Base Type"
@ 15, 52  SAY  GEOSHO->SHOBSTYP
@ 16, 20  SAY "Shoulder Surface Thick (in.)"
@ 16, 49  SAY  GEOSHO->SHOSFTHK
@ 17, 20  SAY "Shoulder Base Thick (in.)"
@ 17, 49  SAY  GEOSHO->SHOBSTHK
@ 18, 20  SAY "Widening Flag"
@ 18, 53  SAY  GEOSHO->WIDENFLG
@  2, 14  TO 20, 64    DOUBLE
```

Format Programs For Screens
Surface

```
@  4, 23  SAY "TEXAS FLEXIBLE PAVEMENT DATABASE"
@  5, 35  SAY "INQUIRY"
@  6, 26  SAY "Inventory Data - Surface"
@  9, 36  SAY "SID Number"
@  9, 54  SAY  SURFACE->SID_NO
@ 10, 36  SAY "Structure Number"
@ 10, 56  SAY  SURFACE->STRUCNUM
@ 11, 36  SAY "Layer Number"
@ 11, 56  SAY  SURFACE->LAYNUM
@ 13, 20  SAY "Aggregate Appl. Rate (S.Y./C.Y.)"
@ 13, 55  SAY  SURFACE->AGAPPLRT  PICTURE "999"
@ 14, 20  SAY "Admixture Type"
@ 14, 46  SAY  SURFACE->ADMXTYP
@ 15, 20  SAY "Admixture Percent (%)"
@ 15, 53  SAY  SURFACE->ADMXPER
@ 16, 20  SAY "Asphalt Appl. Rate (Gal/S.Y.)"
@ 16, 54  SAY  SURFACE->ASAPPLRT
@  3, 14  TO 18, 64    DOUBLE
```

Format Programs For Screens
Subgrade

```
@  4, 24  SAY "TEXAS FLEXIBLE PAVEMENT DATABASE"
@  5, 35  SAY "INQUIRY"
@  6, 27  SAY "Inventory Data - Subgrade"
@  9, 36  SAY "SID Number"
@  9, 54  SAY  SUBGRADE->SID_NO
@ 10, 36  SAY "Structure Number"
@ 10, 56  SAY  SUBGRADE->STRUCNUM
@ 11, 36  SAY "Layer Number"
@ 11, 56  SAY  SUBGRADE->LAYNUM
@ 13, 20  SAY "Percent Passing No. 200 Sieve (%)"
@ 13, 54  SAY  SUBGRADE->PPSV200
@ 14, 20  SAY "Plasticity Index"
@ 14, 54  SAY  SUBGRADE->PLASTIX
@ 15, 20  SAY "Liquid Limit"
@ 15, 54  SAY  SUBGRADE->LIQLIM
@ 16, 20  SAY "Texas Triaxial Class"
@ 16, 55  SAY  SUBGRADE->TXTRIAXL
@ 17, 20  SAY "Permeability Index"
@ 17, 53  SAY  SUBGRADE->PERMIX
@  3, 14  TO 19, 64    DOUBLE
```

Format Programs For Screens
Visual Rating

```
@  2, 24  SAY "TEXAS FLEXIBLE PAVEMENT DATABASE"
@  3, 35  SAY "INQUIRY"
@  4, 22  SAY "Monitoring Data - Visual Rating File"
@  7,  7  SAY "Actual Date of Measurement"
@  7, 34  SAY  VISUAL->ACIMONTH  PICTURE "99"
@  7, 36  SAY "/"
@  7, 37  SAY  VISUAL->ACTYEAR
@  7, 45  SAY "SID Number"
@  7, 63  SAY  VISUAL->SID_NO
@  8, 45  SAY "Structure Number"
@  8, 65  SAY  VISUAL->STRUCNUM
@  9, 45  SAY "Layer Number"
@  9, 65  SAY  VISUAL->LAYNUM
@ 12,  7  SAY "Rutting      Block Cr     Alligtr Cr    Longitud Cr   Transv
Cr"
@ 13,  7  SAY  VISUAL->RUTTSL
@ 13,  8  SAY "S"
@ 13, 10  SAY  VISUAL->RUTTMD
@ 13, 11  SAY "M"
@ 13, 13  SAY  VISUAL->RUTTSV
@ 13, 14  SAY "SV"
@ 13, 20  SAY  VISUAL->BLKCRKSL
@ 13, 21  SAY "S"
@ 13, 23  SAY  VISUAL->BLKCRKMD
@ 13, 24  SAY "M"
@ 13, 26  SAY  VISUAL->BLKCRKSV
@ 13, 27  SAY "SV"
@ 13, 33  SAY  VISUAL->ALLGCRSL
@ 13, 34  SAY "S"
@ 13, 37  SAY  VISUAL->ALLGCRMD
@ 13, 38  SAY "M"
@ 13, 41  SAY  VISUAL->ALLGCRSV
@ 13, 42  SAY "SV"
@ 13, 47  SAY  VISUAL->LONGCRSL
@ 13, 48  SAY "S"
@ 13, 51  SAY  VISUAL->LONGCRMD
@ 13, 52  SAY "M"
@ 13, 55  SAY  VISUAL->LONGCRSV
@ 13, 56  SAY "SV"
@ 13, 61  SAY  VISUAL->TRANCRSL
@ 13, 62  SAY "S"
@ 13, 64  SAY  VISUAL->TRANCRMD
@ 13, 65  SAY "M"
@ 13, 67  SAY  VISUAL->TRANCRSV
@ 13, 68  SAY "SV"
@ 16,  7  SAY "Seal Code  Patching    Failures/Mi         Pavement Rat Scr"
@ 16, 67  SAY  VISUAL->PRS
@ 17, 11  SAY  VISUAL->SEALCRCD
@ 17, 18  SAY  VISUAL->PATCHGD
```

```
@ 17, 19  SAY "G"
@ 17, 21  SAY  VISUAL->PATCHFR
@ 17, 22  SAY "F"
@ 17, 24  SAY  VISUAL->PATCHPR
@ 17, 25  SAY "P"
@ 17, 35  SAY  VISUAL->FAILMILE
@ 17, 45  SAY "PES Pavement Rat Scr"
@ 17, 66  SAY  VISUAL->PESPVTRS
@ 18, 46  SAY "Unwght Vis. Rat Scr"
@ 18, 66  SAY  VISUAL->UVURS
@  1,  2  TO 20, 73    DOUBLE
```

Format Programs For Screens
Serviceability Index

```
@  3, 24  SAY "TEXAS FLEXIBLE PAVEMENT DATABASE"
@  4, 35  SAY "INQUIRY"
@  5, 21  SAY "Monitoring Data - Serviceability Index"
@  7, 36  SAY "SID Number"
@  7, 54  SAY  SI->SID_NO
@  8, 36  SAY "Structure Number"
@  8, 56  SAY  SI->STRUCNUM
@  9, 36  SAY "Layer Number"
@  9, 56  SAY  SI->LAYNUM
@ 10, 36  SAY "Date"
@ 10, 50  SAY  SI->ACIMONTH  FUNCTION "z"
@ 10, 52  SAY "/"
@ 10, 53  SAY  SI->ACIDAY  FUNCTION "Z"
@ 10, 55  SAY "/"
@ 10, 56  SAY  SI->ACIYEAR  FUNCTION "Z"
@ 12, 22  SAY "Count of Observation"
@ 12, 55  SAY  SI->SICOUNT
@ 13, 22  SAY "Mean"
@ 13, 51  SAY  SI->SIMEAN
@ 14, 22  SAY "Standard Deviation"
@ 14, 51  SAY  SI->SISD
@ 15, 22  SAY "Low Value"
@ 15, 55  SAY  SI->SILOWVAL
@ 16, 22  SAY "High Value"
@ 16, 55  SAY  SI->SIHIVAL
@  2, 14  TO 18, 64    DOUBLE
```

Format Programs For Screens
Falling Weight SSI

```
@  2, 24  SAY "TEXAS FLEXIBLE PAVEMENT DATABASE"
@  3, 35  SAY "INQUIRY"
@  4, 21  SAY "Monitoring Data - Falling Weight SSI"
@  7,  6  SAY "Date"
@  7, 12  SAY  FALLWGHT->MONTH
@  7, 14  SAY "/"
@  7, 15  SAY  FALLWGHT->DAY
@  7, 17  SAY "/"
@  7, 18  SAY  FALLWGHT->YEAR
@  7, 25  SAY "SID Number"
@  7, 43  SAY  FALLWGHT->SID_NO
@  7, 57  SAY "Average SSI"
@  7, 70  SAY  FALLWGHT->RWSSIAVG
@  8, 11  SAY "(MM/DD/YY)     Structure Number"
@  8, 45  SAY  FALLWGHT->STRUCNUM
@  8, 57  SAY "Temperature"
@  8, 71  SAY  FALLWGHT->SSITEMP
@  9, 25  SAY "Layer Number"
@  9, 45  SAY  FALLWGHT->LAYNUM
@ 11, 18  SAY "Reading 1   Reading 2   Reading 3   Reading 4   Reading 5"
@ 12,  5  SAY "Geophone 1"
@ 12, 20  SAY  FALLWGHT->SSIGP11
@ 12, 32  SAY  FALLWGHT->SSIGP21  PICTURE "99.99"
@ 12, 44  SAY  FALLWGHT->SSIGP31
@ 12, 56  SAY  FALLWGHT->SSIGP41
@ 12, 68  SAY  FALLWGHT->SSIGP51
@ 13,  5  SAY "Geophone 2"
@ 13, 20  SAY  FALLWGHT->SSIGP12
@ 13, 32  SAY  FALLWGHT->SSIGP22
@ 13, 44  SAY  FALLWGHT->SSIGP32
@ 13, 56  SAY  FALLWGHT->SSIGP42
@ 13, 68  SAY  FALLWGHT->SSIGP52
@ 14,  5  SAY "Geophone 3"
@ 14, 20  SAY  FALLWGHT->SSIGP13
@ 14, 32  SAY  FALLWGHT->SSIGP23  PICTURE "99.99"
@ 14, 44  SAY  FALLWGHT->SSIGP33
@ 14, 56  SAY  FALLWGHT->SSIGP43
@ 14, 68  SAY  FALLWGHT->SSIGP53
@ 15,  5  SAY "Geophone 4"
@ 15, 20  SAY  FALLWGHT->SSIGP14
@ 15, 32  SAY  FALLWGHT->SSIGP24
@ 15, 44  SAY  FALLWGHT->SSIGP34
@ 15, 56  SAY  FALLWGHT->SSIGP44
@ 15, 68  SAY  FALLWGHT->SSIGP54
@ 16,  5  SAY "Geophone 5"
@ 16, 20  SAY  FALLWGHT->SSIGP15
@ 16, 32  SAY  FALLWGHT->SSIGP25
@ 16, 44  SAY  FALLWGHT->SSIGP35
@ 16, 56  SAY  FALLWGHT->SSIGP45
```

```
@ 16, 68  SAY  FALLWGHT->SSIGP55
@ 17,  5  SAY "Geophone 6"
@ 17, 20  SAY  FALLWGHT->SSIGP16
@ 17, 32  SAY  FALLWGHT->SSIGP26
@ 17, 44  SAY  FALLWGHT->SSIGP36
@ 17, 56  SAY  FALLWGHT->SSIGP46
@ 17, 68  SAY  FALLWGHT->SSIGP56
@ 18,  5  SAY "Geophone 7"
@ 18, 20  SAY  FALLWGHT->SSIGP17
@ 18, 32  SAY  FALLWGHT->SSIGP27
@ 18, 44  SAY  FALLWGHT->SSIGP37
@ 18, 56  SAY  FALLWGHT->SSIGP47
@ 18, 68  SAY  FALLWGHT->SSIGP57
@  1,  2  TO 20, 77    DOUBLE
```

Format Programs For Screens
Dynaflect Measurement

```
@  3, 24  SAY "TEXAS FLEXIBLE PAVEMENT DATABASE"
@  4, 35  SAY "INQUIRY"
@  5, 20  SAY "Monitoring Data - Dynaflect Measurement"
@  7, 38  SAY "SID Number"
@  7, 55  SAY  DYNAFLLD->SID_NO
@  8, 38  SAY "Structure Number"
@  8, 57  SAY  DYNAFLLD->STRUCNUM
@  9, 38  SAY "Layer Number"
@  9, 57  SAY  DYNAFLLD->LAYNUM
@ 10, 38  SAY "Date"
@ 10, 51  SAY  DYNAFLLD->MONTH
@ 10, 53  SAY "/"
@ 10, 54  SAY  DYNAFLLD->DAY
@ 10, 56  SAY "/"
@ 10, 57  SAY  DYNAFLLD->YEAR
@ 12, 21  SAY "STATION"
@ 12, 53  SAY  DYNAFLLD->STATION
@ 13, 21  SAY "Reading for Sensor 1"
@ 13, 50  SAY  DYNAFLLD->SENS1RD
@ 14, 21  SAY "Reading for Sensor 2"
@ 14, 50  SAY  DYNAFLLD->SENS2RD
@ 15, 21  SAY "Reading for Sensor 3"
@ 15, 50  SAY  DYNAFLLD->SENS3RD
@ 16, 21  SAY "Reading for Sensor 4"
@ 16, 50  SAY  DYNAFLLD->SENS4RD
@ 17, 21  SAY "Reading for Sensor 5"
@ 17, 50  SAY  DYNAFLLD->SENS5RD
@  2, 14  TO 19, 64    DOUBLE
```

Format Programs For Screens
Skid

```
@  3, 24  SAY "TEXAS FLEXIBLE PAVEMENT DATABASE"
@  4, 35  SAY "INQUIRY"
@  5, 23  SAY "Monitoring Data - Skid Measurement"
@  8, 34  SAY "SID Number"
@  8, 52  SAY  SKID->SID_NO
@  9, 34  SAY "Structure Number"
@  9, 54  SAY  SKID->STRUCNUM
@ 10, 34  SAY "Layer Number"
@ 10, 54  SAY  SKID->LAYNUM
@ 11, 34  SAY "Date"
@ 11, 51  SAY  SKID->MONTH
@ 11, 53  SAY "/"
@ 11, 54  SAY  SKID->YEAR
@ 13, 29  SAY "Mean"
@ 13, 41  SAY  SKID->SKIDNUMM
@ 14, 29  SAY "High"
@ 14, 41  SAY  SKID->SKIDNUMH
@ 15, 29  SAY "Low"
@ 15, 41  SAY  SKID->SKIDNUML
@  2, 16  TO 17, 63    DOUBLE
```

Format Programs For Screens
Traffic

```
@  3, 24  SAY "TEXAS FLEXIBLE PAVEMENT DATABASE"
@  4, 35  SAY "INQUIRY"
@  5, 32  SAY "Traffic Data"
@  7, 39  SAY "SID Number"
@  7, 50  SAY  TRAFFIC->SID_NO
@  8, 39  SAY "Year"
@  8, 50  SAY  TRAFFIC->YEAR
@ 10, 21  SAY "Annual Average Daily Traffic"
@ 10, 53  SAY  TRAFFIC->AADT1WAY
@ 11, 21  SAY "Annual Cummulative 18 Keal"
@ 12, 28  SAY "- one way"
@ 12, 50  SAY  TRAFFIC->A18KEAL
@ 13, 21  SAY "Percent trucks"
@ 13, 55  SAY  TRAFFIC->PCTTRK
@  2, 14  TO 15, 65    DOUBLE
```

Format Programs For Screens
Weather

```
@  4, 24  SAY "TEXAS FLEXIBLE PAVEMENT DATABASE"
@  5, 35  SAY "INQUIRY"
@  6, 29  SAY "Weather Measurement"
@  8,  9  SAY "County Number"
@  8, 26  SAY  WEATHER->CNTYNUM
@  9,  9  SAY "Month"
@  9, 27  SAY  WEATHER->MONTH
@ 11, 32  SAY "No. of Yrs Avg    Mean      Std. Dev."
@ 12,  9  SAY "Precipitation"
@ 12, 38  SAY  WEATHER->PRECYRS
@ 12, 49  SAY  WEATHER->PRECMN
@ 12, 60  SAY  WEATHER->PRECSD
@ 13,  9  SAY "Total Freeze Thaw Cycle"
@ 13, 38  SAY  WEATHER->TFTCYRS
@ 13, 49  SAY  WEATHER->TFTCMN
@ 13, 60  SAY  WEATHER->TFTCSD
@ 14,  9  SAY "Wet Freeze Thaw Cycle"
@ 14, 38  SAY  WEATHER->WFTCYRS
@ 14, 49  SAY  WEATHER->WFTCMN
@ 14, 60  SAY  WEATHER->WFTCSD
@ 15,  9  SAY "Maximum Temperature"
@ 15, 38  SAY  WEATHER->MTMPYRS
@ 15, 49  SAY  WEATHER->MTMPMN
@ 15, 60  SAY  WEATHER->MTMPSD
@ 16,  9  SAY "Averaged Temperature"
@ 16, 38  SAY  WEATHER->ATMPYRS
@ 16, 49  SAY  WEATHER->ATMPMN
@ 16, 60  SAY  WEATHER->ATMPSD
@  3,  4  TO 18, 72    DOUBLE
```

Format Programs For Screens
Environment

```
@  5, 24  SAY "TEXAS FLEXIBLE PAVEMENT DATABASE"
@  6, 35  SAY "INQUIRY"
@  7, 27  SAY "Environment Measurement"
@ 10, 23  SAY "County Number"
@ 10, 53  SAY  ENV->CNTYNUM
@ 11, 23  SAY "Thornthwaite Index Mean"
@ 11, 50  SAY  ENV->THORNMN  PICTURE "9999999.999"
@ 12, 23  SAY "Thornthwaite Index"
@ 13, 28  SAY "- No. of Years Averaged"
@ 13, 55  SAY  ENV->THORNYRS
@ 14, 23  SAY "Thornthwaite Index"
@ 15, 28  SAY "- Standard Deviation"
@ 15, 51  SAY  ENV->THORNSD
@  4, 14  TO 17, 64    DOUBLE
```

Format Programs For Screens
County Name

```
@  7, 23  SAY "TEXAS FLEXIBLE PAVEMENT DATABASE"
@  8, 34  SAY "INQUIRY"
@  9, 32  SAY "County Table"
@ 12, 24  SAY "County Number"
@ 12, 40  SAY  CNTYTBL->CNTYNUM
@ 13, 24  SAY "County Name"
@ 13, 40  SAY  CNTYTBL->CNTYNAME
@  6, 15  TO 15, 61    DOUBLE
@ 15, 36  TO 15, 36
```

Format Programs For Screens
Material Type Classification

```
@  4, 24  SAY "TEXAS FLEXIBLE PAVEMENT DATABASE"
@  5, 35  SAY "INQUIRY"
@  6, 23  SAY "Material Type Classification Table"
@  9, 21  SAY "Material Code"
@  9, 42  SAY  MATLTBL->MATCODE
@ 10, 21  SAY "Material Description"
@ 10, 42  SAY  MATLTBL->MATDESC
@ 11, 21  SAY "Material Short Form"
@ 11, 42  SAY  MATLTBL->MATSHRT
@ 12, 21  SAY "Layer Description"
@ 12, 42  SAY  MATLTBL->LAYRDES
@  3, 14  TO 14, 64    DOUBLE
```

Format Programs For Screens
Type of Pavement

```
@  4, 24  SAY "TEXAS FLEXIBLE PAVEMENT DATABASE"
@  5, 35  SAY "INQUIRY"
@  6, 27  SAY "Type of Pavement Table"
@ 10, 20  SAY "Pavement Code"
@ 10, 36  SAY  PAVETYPE->PAVECODE
@ 11, 20  SAY "Type of Base"
@ 11, 36  SAY  PAVETYPE->BASETYPE
@ 12, 20  SAY "Surface Thickness"
@ 12, 36  SAY  PAVETYPE->BASETHK
@ 13, 20  SAY "Base Seal"
@ 13, 36  SAY  PAVETYPE->BASESEAL
@  3, 14  TO 15, 64    DOUBLE
```

Format Programs For Screens
District Temperature Constant

```
@  6, 24  SAY "TEXAS FLEXIBLE PAVEMENT DATABASE"
@  7, 36  SAY "INQUIRY"
@  8, 27  SAY "District Temperature Table"
@ 11, 27  SAY "District Number"
@ 11, 51  SAY  DISTTEMP->DISTRICT
@ 12, 27  SAY "Temperature Constant"
@ 12, 51  SAY  DISTTEMP->TEMPCONS
@  5, 17  TO 14, 62    DOUBLE
```

Format Programs For Screens
Widening Flag

```
@  5, 24  SAY "TEXAS FLEXIBLE PAVEMENT DATABASE"
@  6, 35  SAY "INQUIRY"
@  7, 31  SAY "Widening Table"
@ 10,  8  SAY "Code"
@ 10, 21  SAY  WIDENFLG->WIDNCODE
@ 11,  8  SAY "Description"
@ 11, 21  SAY  WIDENFLG->WIDNDESC
@ 12,  8  SAY "Comments:"
@ 13, 10  SAY  WIDENFLG->COMMENTS
@  4,  4  TO 15, 73    DOUBLE
```

Format Programs For Screens
Layer Description

```
@  4, 21  SAY "TEXAS FLEXIBLE PAVEMENT DATABASE"
@  5, 33  SAY "INQUIRY"
@  6, 26  SAY "Layer Description Table"
@  9, 16  SAY "Code"
@  9, 33  SAY  LAYERTBL->CODE
@ 10, 16  SAY "Short Descriptn"
@ 10, 33  SAY  LAYERTBL->CODE_DESC
@ 11, 16  SAY "Description"
@ 11, 33  SAY  LAYERTBL->CODE_LAYR
@  3, 12  TO 13, 61    DOUBLE
```

Format Programs For Screens
Functional Classification

```
@  6, 25  SAY "TEXAS FLEXIBLE PAVEMENT DATABASE"
@  7, 37  SAY "INQUIRY"
@  8, 25  SAY "Functional Classification Table"
@  11, 13  SAY "Code"
@  11, 27  SAY  FUNCLTBL->CODE  PICTURE "9"
@  12, 13  SAY "Description"
@  12, 27  SAY  FUNCLTBL->DESCRIPT
@  5, 10  TO 14, 69    DOUBLE
```

Program Listing

```
*
* SUBSYSTEM:        INQUIRY MAIN MENU
* PROGRAM NAME:     INQUIRY.PRG              05/31/88
* MODIFIED ON:      09/26/88
* PROJECT 2456 -    TEXAS FLEXIBLE PAVEMENT DATABASE CONVERSION
* TAMU/TTI
* AUTHOR:           TREVOR X. PEREIRA
* PURPOSE:          TO RETREIVE DATA TO THE SCREEN FOR THE FOLLOWING:
*                         - Monitoring data
*                         - Environment data
*                         - Inventory data
*                         - Traffic data
*                         - Tables
*

* set parameters and initialize variables
STORE 13 TO MSID_NO
CLOSE DATABASES
SET FORMAT TO
SET PROCEDURE TO \PAVEDB\INQUIRY\INQ_COLL
MREPEAT2 = .T.

DO WHILE MREPEAT2
 STORE " " TO Inqpick
 * set up the INQUIRY menu screen and do the loop until INQPICK is 1-5
 DO WHILE .NOT. (INQPICK $ '12345')
  @  0,  0  CLEAR
  @  4, 19  SAY "TEXAS FLEXIBLE PAVEMENT DATABASE        1.0"
  @  5, 32  SAY "Inquiry"
  @  9, 17  SAY "1 - Inventory Data"
  @ 10, 17  SAY "2 - Monitoring Data"
  @ 11, 17  SAY "3 - Traffic Data"
  @ 12, 17  SAY "4 - Environmental Data"
  @ 13, 17  SAY "5 - Tables"
  @ 16, 37  SAY "OPTION ====> "  GET INQPICK
  @  3,  9  TO 18, 65    DOUBLE
  READ

  IF READKEY() = 12
     RETURN TO MASTER
  ENDIF
 ENDDO

 IF MREPEAT2
  * according to the response received from the INQUIRY menu, the appropriate
  * commands are executed
  DO CASE
  * Display the Inventory data to the screen
   CASE INQPICK = "1"
    MREPEAT = .T.
    DO WHILE MREPEAT
```

```
    CLEAR
    INQCOLL = " "
    MSID_NO = 13
    DO WHILE .NOT. (INQCOLL $ '123456')
      @  3, 21  SAY "TEXAS FLEXIBLE PAVEMENT DATABASE        1.1"
      @  4, 33  SAY "INQUIRY"
      @  5, 30  SAY "Inventory Data"
      @  7, 16  SAY "1 - Location"
      @  8, 16  SAY "2 - Layer ID"
      @  9, 16  SAY "3 - Geometric & Shoulder"
      @ 10, 16  SAY "4 - Surface"
      @ 11, 16  SAY "5 - Subgrade"
      @ 12, 16  SAY "6 - Layer Thickness Across the Road"
      @ 15, 37  SAY "OPTION ====>" GET INQCOLL
      @ 17, 27  SAY "Enter SID Number ====>" GET MSID_NO PICTURE "9999"
      @  2,  9  TO 19, 65    DOUBLE
      READ
      IF READKEY() = 12
        MREPEAT = .F.
        EXIT
      ENDIF
    ENDDO
    IF MREPEAT
       DO COLLECT1
       CLEA TYPE
    ENDIF
    SET FORMAT TO
   ENDDO

* Display the Monitoring data to the screen if the choice is 2 on the
* Inquiry main menu
CASE INQPICK = "2"
  MREPEAT = .T.
  DO WHILE MREPEAT
    CLEAR
    INQCOLL = " "
    MSID_NO = 13
    DO WHILE .NOT. (INQCOLL $ '12345')
      @  3, 22  SAY "TEXAS FLEXIBLE PAVEMENT DATABASE       1.2"
      @  4, 34  SAY "INQUIRY"
      @  5, 30  SAY "Monitoring Data"
      @  8, 15  SAY "1 - Visual"
      @  9, 15  SAY "2 - Serviceability Index"
      @ 10, 15  SAY "3 - Falling Weight"
      @ 11, 15  SAY "4 - Dynaflect"
      @ 12, 15  SAY "5 - Skid"
      @ 14, 40  SAY "OPTION ====> " GET INQCOLL
      @ 16, 30  SAY "Enter SID Number ====> " GET MSID_NO PICTURE "9999"
      @  2,  9  TO 18, 65    DOUBLE
      READ
      IF READKEY() = 12
         MREPEAT = .F.
         EXIT
      ENDIF
```

```
    ENDDO
    IF MREPEAT
      DO COLLECT2
      CLEA TYPE
    ENDIF
    SET FORMAT TO
   ENDDO

* If choice 3 is chosen, display data for the traffic file
CASE INQPICK = "3"
    CLEAR
    * Display TRAFFIC data for a SID number to the screen
    MREPEAT = .T.
    STORE "Traffic Data" TO MENUHEAD
    MENUHEAD = TRIM(MENUHEAD)
    MCENTER = 37 - (LEN(MENUHEAD)/2)
    MSID_NO = 13
    DO SID_INQ
    IF MREPEAT
      USE \PAVEDB\FILES\TRAFFIC INDE \PAVEDB\INDEXES\TRAFFIC
      SEEK STR(MSID_NO,4)
      IF FOUND()
         SET FORMAT TO \PAVEDB\INQUIRY\INQ_TRAF
         CHANGE
      ELSE
         @ 20, 15 SAY "SID number not found"
         WAIT
      ENDIF
    ENDIF
    USE
    SET FORMAT TO

* if choice 4 is chosen, display one of the 5 INQUIRY table screens
CASE INQPICK = "4"
   MREPEAT = .T.
   DO WHILE MREPEAT
    CLEAR
    INQCOLL = " "
    MCNTYNUM = 1
    DO WHILE .NOT. (INQCOLL $ '12')
      @  4, 21  SAY "TEXAS FLEXIBLE PAVEMENT DATABASE        1.4"
      @  5, 33  SAY "INQUIRY"
      @  6, 28  SAY "Environmental Data"
      @  9, 15  SAY "1 - Environment"
      @ 10, 15  SAY "2 - Weather"
      @ 14, 40  SAY "OPTION ====> " GET INQCOLL
      @ 16, 27  SAY "Enter county Number ====> " GET MCNTYNUM PICTURE "999"
      @  3,  9  TO 18, 65    DOUBLE
      READ
      IF READKEY() = 12
         MREPEAT = .F.
         EXIT
      ENDIF
    ENDDO
```

```
    IF MREPEAT
     DO CASE
      CASE INQCOLL = "1"
        * Display ENVIRONMENT data for a SID number to the screen
        USE \PAVEDB\FILES\ENV INDE \PAVEDB\INDEXES\ENV
        SEEK mcntynum
        IF FOUND()
          MFOUND = .T.
        ENDIF
        IF MCNTYNUM = 0
          MFOUND = .T.
          GOTO TOP
        ENDIF
        IF MFOUND
          SET FORMAT TO \PAVEDB\INQUIRY\INQ_ENV
          CHANGE
        ELSE
          @ 20, 15 SAY "COUNTY number not found"
          WAIT
        ENDIF
        USE

      CASE INQCOLL = "2"
        * Display WEATHER data for a SID number to the screen
        USE \PAVEDB\FILES\WEATHER INDE \PAVEDB\INDEXES\WEATHER
        SEEK STR(MCNTYNUM,3)
        IF FOUND()
          MFOUND = .T.
        ENDIF
        IF MCNTYNUM = 0
          MFOUND = .T.
          GOTO TOP
        ENDIF
        IF MFOUND
          SET FORMAT TO \PAVEDB\INQUIRY\INQ_WEAT
          CHANGE
        ELSE
          @ 20,15 SAY "COUNTY number not found"
          WAIT
        ENDIF
     ENDCASE
    ENDIF
    SET FORMAT TO
    USE
   ENDDO

* if choice 5 is chosen, display one of the 5 INQUIRY table screens
CASE INQPICK = "5"
   DO \PAVEDB\EDITUPDT\TABLFILE
   MREPEAT = .T.
   DO WHILE MREPEAT
    CLEAR
    INQCOLL = " "
    MCNTYNUM = 1
```

```
DO WHILE .NOT. (INQCOLL $ '1234567')
  @  3, 21  SAY "TEXAS FLEXIBLE PAVEMENT DATABASE        1.5"
  @  4, 33  SAY "INQUIRY"
  @  5, 33  SAY "Tables"
  @  8, 16  SAY "1 - County Name"
  @  9, 16  SAY "2 - Material Type"
  @ 10, 16  SAY "3 - Type of Pavement"
  @ 11, 16  SAY "4 - District Temperature Constant"
  @ 12, 16  SAY "5 - Widening Flag"
  @ 13, 16  SAY "6 - Layer Description"
  @ 14, 16  SAY "7 - Functional Classification"
  @ 17, 42  SAY "OPTION ======> " GET INQCOLL
  @  2,  9  TO 18, 65    DOUBLE
  READ
  IF READKEY() = 12
    MREPEAT = .F.
    EXIT
  ENDIF
ENDDO
IF MREPEAT
 DO CASE
  CASE INQCOLL = "1"
     @ 19, 15 SAY "Please enter the COUNTY number: "
     @ 19, 49 GET MCNTYNUM PICTURE "999"
     READ
     CLEAR
     USE \PAVEDB\FILES\CNTYTBL INDEX \PAVEDB\INDEXES\CTYTBLNO
     LOCATE FOR CNTYNUM = MCNTYNUM
     IF MCNTYNUM = 0
        GOTO TOP
     ENDIF
     SET FORMAT TO \PAVEDB\INQUIRY\INQ_CNTY
     CHANGE
     USE
  CASE INQCOLL = "2"
     CLEAR
     USE \PAVEDB\FILES\MATLTBL
     SET FORMAT TO \PAVEDB\INQUIRY\INQ_MATL
     CHANGE
     USE
  CASE INQCOLL = "3"
     CLEAR
     USE \PAVEDB\FILES\PAVETYPE
     SET FORMAT TO \PAVEDB\INQUIRY\INQ_PVMT
     CHANGE
     USE
  CASE INQCOLL = "4"
     MDIST = 1
     @ 19, 15 SAY "Please enter the DISTRICT number: "
     @ 19, 49 GET MDIST PICTURE "99"
     READ
     CLEAR
     USE \PAVEDB\FILES\DISTTEMP
     LOCATE FOR DISTRICT = MDIST
```

```
            IF MDIST = 0
                GOTO TOP
            ENDIF
            SET FORMAT TO \PAVEDB\INQUIRY\INQ_DIST
            CHANGE
            USE
         CASE INQCOLL = "5"
            CLEAR
            USE \PAVEDB\FILES\WIDENFLG
            SET FORMAT TO \PAVEDB\INQUIRY\INQ_WIDN
            CHANGE
            USE
         CASE INQCOLL = "6"
            CLEAR
            USE \PAVEDB\FILES\LAYERTBL
            SET FORMAT TO \PAVEDB\INQUIRY\INQ_LTBL
            CHANGE
            USE
         CASE INQCOLL = "7"
            CLEAR
            USE \PAVEDB\FILES\FUNCLTBL
            SET FORMAT TO \PAVEDB\INQUIRY\INQ_FUNC
            CHANGE
            USE
        ENDCASE
       ENDIF
       SET FORMAT TO
      ENDDO
    ENDCASE
    MREPEAT2 = .F.
  ENDIF
  MREPEAT2 = .T.
ENDDO
SET FORMAT TO
SET PROCEDURE TO
CLOSE DATABASES
CLEAR
RETURN TO MASTER
```

Program Listing

```
*
* SUBSYSTEM:            INQUIRY OF INVENTORY AND MONITORING DATA
* PROGRAM NAME:         INQ_COLL.PRG                    05/31/88
* MODIFIED ON:          09/26/88
* CALLED FROM:          INQUIRY.PRG
* PROJECT 2456 - TEXAS FLEXIBLE PAVEMENT DATABASE CONVERSION
* TAMU/TTI
* AUTHOR:               TREVOR X. PEREIRA
* PURPOSE:              THIS IS A PROCEDURE FILE FOR THE MAIN INQUIRY
*                       PROGRAM.  RESPECTIVE PROCEDURES ARE CALLED TO
*                       DISPLAY DATA ON THE SCREEN FOR INDIVIDUAL FILES.
*

*****************************
* PROCEDURE GET SID NUMBER FOR INQUIRY
*****************************
PROCEDURE SID_INQ
STORE 0 TO MSID_NO, VAR1, VAR2, VAR3, COMPARE
STORE .F. TO VALID

* display Sid number entry screen
DO WHILE .NOT. VALID
    @  0,  0  CLEAR
    @  6, 22  SAY "TEXAS FLEXIBLE PAVEMENT DATABASE"
    @  7, 34  SAY "INQUIRY"
    @  8, MCENTER SAY MENUHEAD
    @ 11, 24 SAY "Please Enter SID Number:"
    @ 11, 49  GET  MSID_NO PICTURE "9999"
    @  5, 15 TO 13, 60    DOUBLE

    * get Sid Number
    READ

    * calculates the correct Sid Number
    VAR1 = INT(MSID_NO/1000)
    VAR2 = MOD(INT(MSID_NO/100),10)
    VAR2 = VAR2 * 2
    VAR3 = MOD(INT(MSID_NO/10),10)
    VAR3 = VAR3 * 3
    VAR4 = MOD(VAR1,10)+VAR2+VAR3
    COMPARE = MOD(VAR4,10)

    * sets up the escape key if want to abort
    IF READKEY() = 12
       MREPEAT = .F.
       RETURN
    ENDIF

    * compares calculated Sid number with Sid number entered
    IF COMPARE = MOD(MSID_NO,10)
```

```
        STORE .T. TO VALID
    ELSE
        @ 20, 10 SAY "Invalid SID Number"
        WAIT
    ENDIF

    * verifies that Sid number has been entered
    IF MSID_NO = 0
        STORE .F. TO VALID
        @ 20, 10 SAY "Invalid SID Number"
        WAIT
    ENDIF
ENDDO

* returns to the calling program
RETURN

*****************************
* PROCEDURE COLLECT 1
*****************************
PROCEDURE COLLECT1
MFOUND = .F.
DO CASE
  CASE INQCOLL = "1"
    * Display location data for a SID number to the screen
    USE \PAVEDB\FILES\LOCATION INDE \PAVEDB\INDEXES\LOCSID
    SEEK MSID_NO
    IF FOUND()
       MFOUND = .T.
    ENDIF
    IF MSID_NO = 0
       MFOUND = .T.
       GOTO TOP
    ENDIF
    IF MFOUND
      SET FORMAT TO \PAVEDB\INQUIRY\INQ_LOCN
      CHANGE
    ELSE
      @ 20, 15 SAY "SID number not found"
      wait
    ENDIF
  CASE INQCOLL = "2"
    * Display LAYER IDENTIFICATION data for a SID number to the screen
    USE \PAVEDB\FILES\LAYER INDE \PAVEDB\INDEXES\LAYNDX
    SEEK STR(MSID_NO,4)
    IF FOUND()
       MFOUND = .T.
    ENDIF
    IF MSID_NO = 0
       MFOUND = .T.
       GOTO TOP
    ENDIF
    IF MFOUND .AND. (.NOT. EOF())
      SET FORMAT TO \PAVEDB\INQUIRY\INQ_LAYR
```

```
    CHANGE
  ELSE
    @ 20, 15 SAY "SID number not found"
    WAIT
  ENDIF
CASE INQCOLL = "3"
  * Display GEOMETRIC AND SHOULDER data for a SID number to the screen
  USE \PAVEDB\FILES\GEOSHO INDE \PAVEDB\INDEXES\GEONDX
  SEEK STR(MSID_NO,4)
  IF FOUND()
     MFOUND = .T.
  ENDIF
  IF MSID_NO = 0
     MFOUND = .T.
     GOTO TOP
  ENDIF
  IF MFOUND .AND. (.NOT. EOF())
    SET FORMAT TO \PAVEDB\INQUIRY\INQ_GEO
    CHANGE
  ELSE
    @ 20, 15 SAY "SID number not found"
    WAIT
  ENDIF
CASE INQCOLL = "4"
  * Display SURFACE data for a SID number to the screen
  USE \PAVEDB\FILES\SURFACE INDE \PAVEDB\INDEXES\SURFNDX
  SEEK STR(MSID_NO,4)
  IF FOUND()
     MFOUND = .T.
  ENDIF
  IF MSID_NO = 0
     MFOUND = .T.
     GOTO TOP
  ENDIF
  IF MFOUND .AND. (.NOT. EOF())
    SET FORMAT TO \PAVEDB\INQUIRY\INQ_SURF
    CHANGE
  ELSE
       @ 20, 15 SAY "SID number not found"
       WAIT
  ENDIF
CASE INQCOLL = "5"
  * Display SUBGRADE data for a SID number to the screen
  USE \PAVEDB\FILES\SUBGRADE INDE \PAVEDB\INDEXES\SUBGNDX
  SEEK STR(MSID_NO,4)
  IF FOUND()
     MFOUND = .T.
  ENDIF
  IF MSID_NO = 0
     MFOUND = .T.
     GOTO TOP
  ENDIF
  IF MFOUND .AND. (.NOT. EOF())
    SET FORMAT TO \PAVEDB\INQUIRY\INQ_SUBG
```

```
      CHANGE
    ELSE
         @ 20, 15 SAY "SID number not found"
         WAIT
    ENDIF
  CASE INQCOLL = "6"
    * Display LAYER THICKNESS ACROSS THE ROAD data for a SID number to the
screen
    USE \PAVEDB\FILES\LAYTHICK INDE \PAVEDB\INDEXES\LAYTNDX
    SEEK STR(MSID_NO,4)
    IF FOUND()
       MFOUND = .T.
    ENDIF
    IF MSID_NO = 0
       MFOUND = .T.
       GOTO TOP
    ENDIF
    IF MFOUND .AND. (.NOT. EOF())
      SET FORMAT TO \PAVEDB\INQUIRY\INQ_LAYT
      CHANGE
    ELSE
         @ 20, 15 SAY "SID number not found"
         WAIT
    ENDIF
ENDCASE
CLEAR
RETURN

**************************
* PROCEDURE COLLECT2
**************************
PROCEDURE COLLECT2
MFOUND = .F.
DO CASE
  CASE INQCOLL = "1"
    * Display VISUAL data for a SID number to the screen
    USE \PAVEDB\FILES\VISUAL INDE \PAVEDB\INDEXES\VISUAL
    SEEK STR(MSID_NO,4)
    IF FOUND()
       MFOUND = .T.
    ENDIF
    IF MSID_NO = 0
       MFOUND = .T.
       GOTO TOP
    ENDIF
    IF MFOUND .AND. (.NOT. EOF())
      SET FORMAT TO \PAVEDB\INQUIRY\INQ_VISL
      CHANGE
    ELSE
         @ 20, 15 SAY "SID number not found"
         WAIT
    ENDIF
  CASE INQCOLL = "2"
    * Display SERVICEABILITY INDEX data for a SID number to the screen
```

```
  USE \PAVEDB\FILES\SI INDE \PAVEDB\INDEXES\SI
  SEEK STR(MSID_NO,4)
  IF FOUND()
     MFOUND = .T.
  ENDIF
  IF MSID_NO = 0
     MFOUND = .T.
     GOTO TOP
  ENDIF
  IF MFOUND .AND. (.NOT. EOF())
    SET FORMAT TO \PAVEDB\INQUIRY\INQ_SI
    CHANGE
  ELSE
       @ 20, 15 SAY "SID number not found"
       WAIT
  ENDIF
CASE INQCOLL = "3"
  * Display FALLING WEIGHT data for a SID number to the screen
  USE \PAVEDB\FILES\FALLWGHT INDE \PAVEDB\INDEXES\FALLWGHT
  SEEK STR(MSID_NO,4)
  IF FOUND()
     MFOUND = .T.
  ENDIF
  IF MSID_NO = 0
     MFOUND = .T.
     GOTO TOP
  ENDIF
  IF MFOUND .AND. (.NOT. EOF())
    SET FORMAT TO \PAVEDB\INQUIRY\INQ_FALL
    CHANGE
  ELSE
       @ 20, 15 SAY "SID number not found"
       WAIT
  ENDIF

CASE INQCOLL = "4"
  * Display DYNAFLECT data for a SID number to the screen
  USE \PAVEDB\FILES\DYNAFLLD INDE \PAVEDB\INDEXES\DYNAFLLD
  SEEK STR(MSID_NO,4)
  IF FOUND()
     MFOUND = .T.
  ENDIF
  IF MSID_NO = 0
     MFOUND = .T.
     GOTO TOP
  ENDIF
  IF MFOUND .AND. (.NOT. EOF())
    SET FORMAT TO \PAVEDB\INQUIRY\INQ_DYNA
    CHANGE
  ELSE
       @ 20, 15 SAY "SID number not found"
       WAIT
  ENDIF
```

```
  CASE INQCOLL = "5"
    * Display SKID data for a SID number to the screen
    USE \PAVEDB\FILES\SKID INDE \PAVEDB\INDEXES\SKID
    SEEK STR(MSID_NO,4)
    IF FOUND()
       MFOUND = .T.
    ENDIF
    IF MSID_NO = 0
       MFOUND = .T.
       GOTO TOP
    ENDIF
    IF MFOUND .AND. (.NOT. EOF())
      SET FORMAT TO \PAVEDB\INQUIRY\INQ_SKID
      CHANGE
    ELSE
         @ 20, 15 SAY "SID number not found"
         WAIT
    ENDIF
ENDCASE
CLEAR
RETURN
```

CHAPTER 3

REPORT SUBSYSTEM

GENERAL NARRATIVE

The Report Menu System written in dBase III Plus allows the user to print reports for any of the 14 master files and 7 tables. In addition to these 23 general reports, two specific reports have been set up - Summary and Inventory Update. The reports have been divided into 7 categories: Summary, Inventory Update, Inventory Data, Monitoring Data, Traffic Data, Environment Data and Tables. When a report is chosen, the program accesses the appropriate file and retrieves the data. The report is then sent to the printer as well as the screen. The reports program (REPORTS.PRG) generates all the reports except Summary and Inventory Update for which it calls separate programs. REPORTS.PRG also produces all of the screens contained in the Report Subsystem. All of the Report Subsystem programs are stored in the subdirectory \PAVEDB\REPORTS. The data and view files are stored in the subdirectory \PAVEDB\FILES. The index files are stored in \PAVEDB\INDEXES.

The Summary and Inventory Update Reports have the option of being printed by Individual SID number (SUMMSID.PRG or SUMM2SID.PRG), by District (SUMMDIST.PRG or SUM2DIST.PRG) or by ALL SID numbers (SUMMALL.PRG or SUM2FILE.PRG). For the individual SID numbers, the program gets the SID number from the user, accesses the appropriate files to obtain the data and prints the report. For reports by district, the program obtains the district number from the user, gets the SID numbers for that district, accesses the applicable files to obtain the data and prints the report. For reports on all SID numbers, the program automatically gets the SID number, retrieves the data from the pertinent files and prints out the report.

The following information about the Report Subsystem is provided in this chapter:

- Program Flow Diagram
- Program Specifications
- Menu Screens
- Sample Reports
- Program Listings.

Report Subsystem - Program Flow Diagram

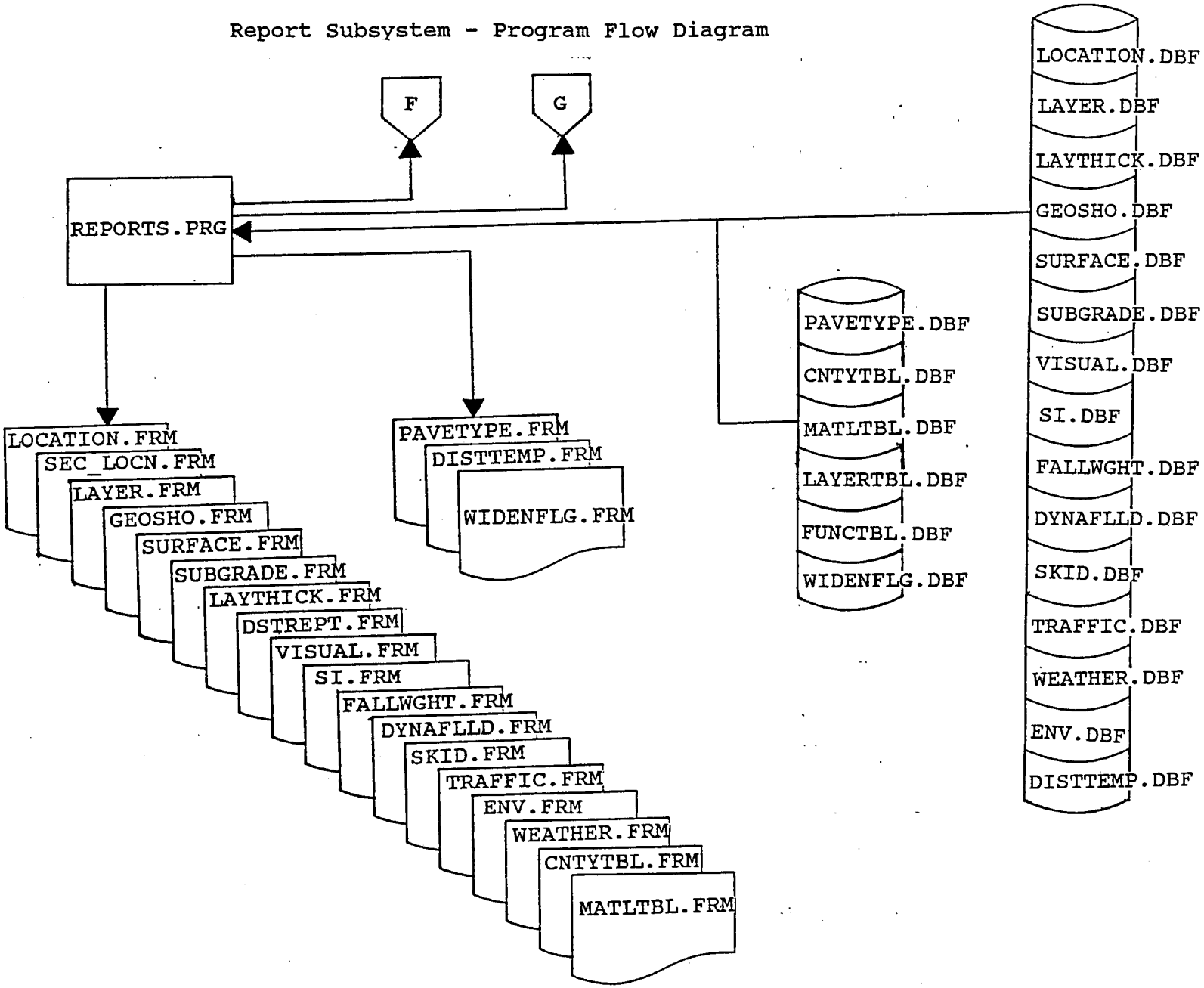


FIGURE 7

Report Subsystem - Program Flow Diagram

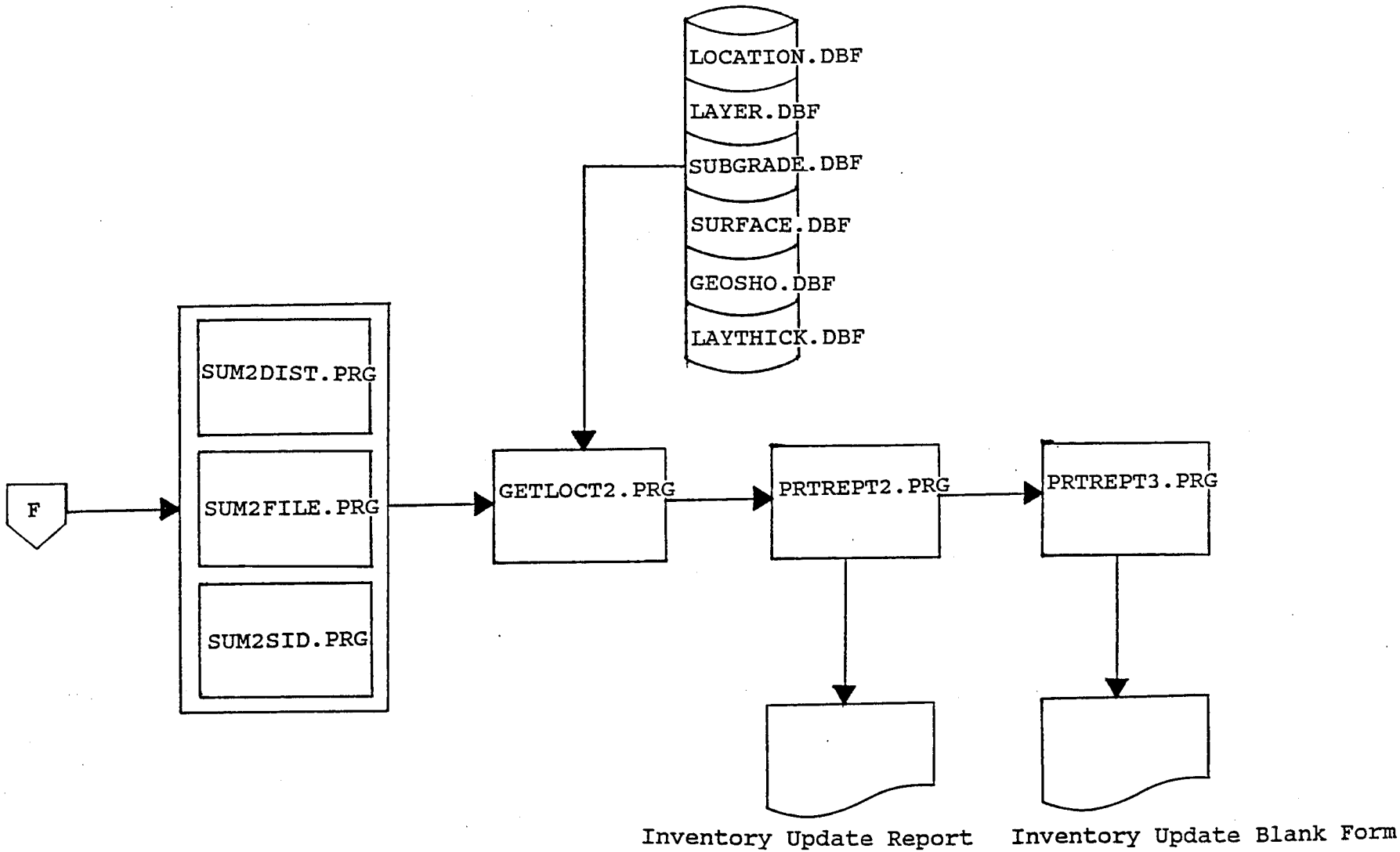


FIGURE 8

Report Subsystem - Program Flow Diagram

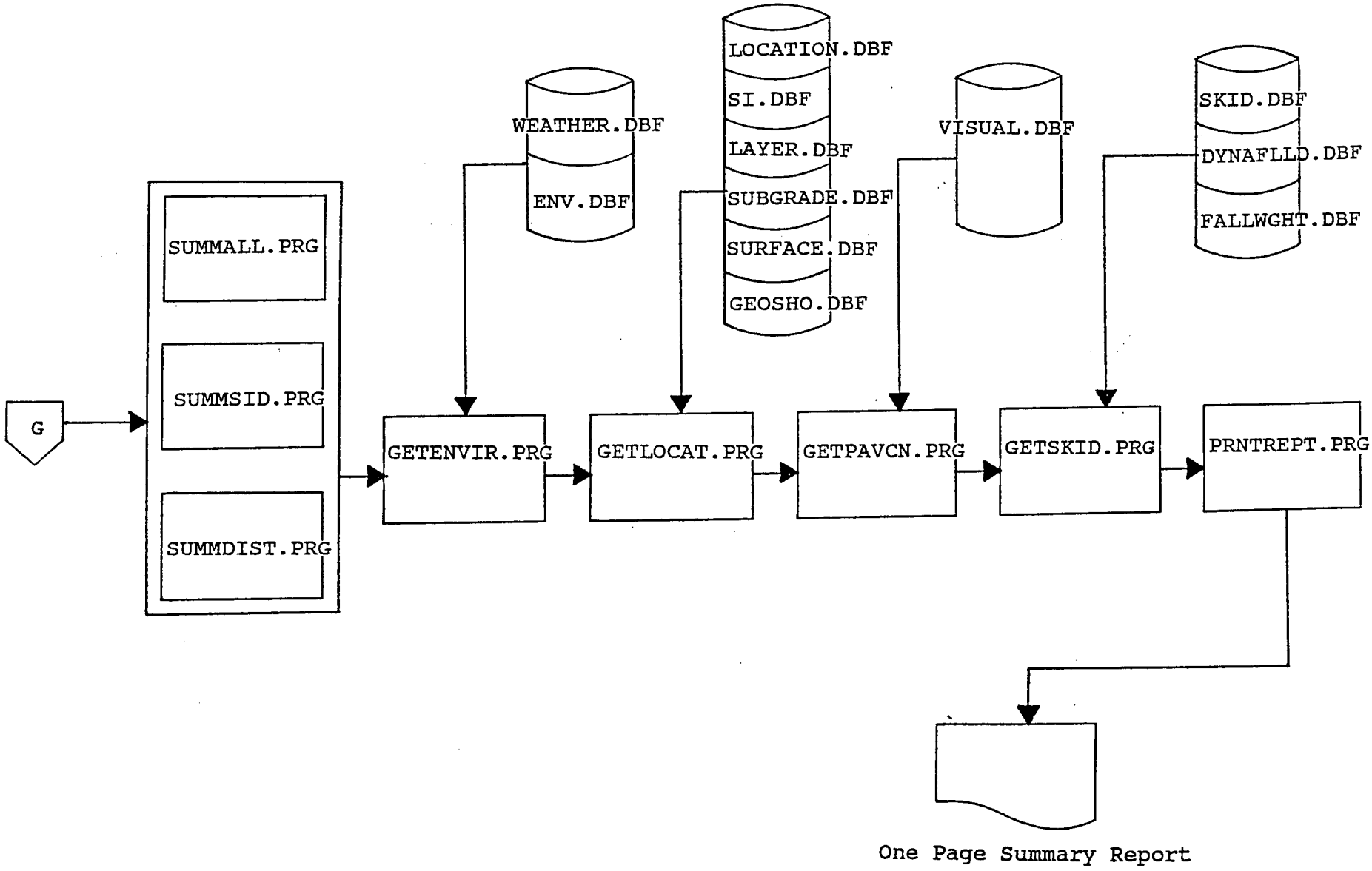


FIGURE 9

PROGRAM SPECIFICATION

Program Name: REPORTS.PRG

Purpose: Display report request screens and produce user requested reports.

Input Files: The following files are used along with their indices:

	Data File	Index File
Location	LOCATION.DBF	LOCSID.NDX
Layer Identification	LAYER.DBF	LAYNDX.NDX
Geometric & Shoulder	GEOSHO.DBF	GEONDX.NDX
Surface	SURFACE.DBF	SURFNDX.NDX
Subgrade	SUBGRADE.DBF	SUBGNDX.NDX
Layer Thickness	LAYTHICK.DBF	LAYTNDX.NDX
Visual Rating	VISUAL.DBF	VISUAL.NDX
Serviceability Index	SI.DBF	SI.NDX
Falling Weight SSI	FALLWGHT.DBF	FALLWGHT.NDX
Dynaflect Measure	DYNAFLLD.DBF	DYNAFLLD.NDX
Skid Measurement	SKID.DBF	SKID.NDX
Environment	ENV.DBF	ENV.NDX
Weather	WEATHER.DBF	WEATHER.NDX
Traffic	TRAFFIC.DBF	TRAFFIC.NDX
County Name Table	CNTYTBL.DBF	CTYTBLNO.NDX
Material Type Table	MATLTBL.DBF	
Pavement Type Table	PAVETYPE.DBF	
District Temperature	DISTTEMP.DBF	
Widening Flag Table	WIDENFLG.DBF	
Layer Description	LAYERTBL.DBF	
Functional Class.	FUNCLTBL.DBF	
Structural Section	STRUCSEC.VUE	
Structural Section 2	STRCSEC2.VUE	

dBase Programs Called (See Program Flow Diagram):

Program Name	Description
SUMMSID.PRG	Gets SID Number for the Summary Report
SUMMDIST.PRG	Gets SID Numbers of a District for the Summary Report
SUMMALL.PRG	Gets all SID Number's for the Summary
GETSKID.PRG	Gets Skid Data for Summary Report
GETENVIR.PRG	Gets Environment Data for Summary Report
GETLOCAT.PRG	Gets Location Data for Summary Report
GETPAVCN.PRG	Gets Pavement Condition Data for the Summary Report

dBase Programs Called (See Program Flow Diagram): (continued)

Program Name	Description
PRNTREPT.PRG	Prints the Summary Report
SUMM2SID.PRG	Gets SID Number for the Inventory Update Report
SUMM2DIST.PRG	Gets SID Numbers for a District for Inventory Update Report
SUM2FILE.PRG	Gets all SID Numbers for the Inventory Update Report
GETLOCT2.PRG	Gets Data for the Inventory Update Report.
PRTREPT2.PRG	Prints the Inventory Update Report
PRTREPT3.PRT	Prints a Blank Inventory Update Form.

NOTE: REPORTS.PRG produces all of the reports except the Summary Report and the Inventory Update Report.

Output Files: None

Reports (Using dBASE III Procedures):

The following reports are printed on request:

		REPORT FILES
Inventory Data	- Location	SEC_LOCN.FRM
	- Location Section	LOCATION.FRM
	- Layer Identification	LAYER.FRM
	- Layer Thickness	LAYTHICK.FRM
	- Geometric & Shoulder Info	GEOSHO.FRM
	- Surface	SURFACE.FRM
	- Subgrade	SUBGRADE.FRM
	- Double Surface Treated	DSTREPT.FRM
Summary	- For All SID Numbers	SUMMALL.PRG
	- For Individual SID Number	SUMMSID.PRG
	- For a Particular District	SUMMDIST.PRG
Inventory Update	- For Individual SID numbers	SUMM2SID.PRG
	- For a Particular District	SUM2DIST.PRG
	- For All SID Numbers	FILE.PRG
Monitoring Data	- Visual	VISUAL.FRM
	- Serviceability Index	SI.FRM
	- Falling Weight	FALLWGHT.FRM & FALLWGH2.FRM
	- Dynaflect	DYNAFLLD.FRM
	- Skid	SKID.FRM
Environment Data	- Environment	ENV.FRM
	- Weather	WEATHER.FRM
Traffic Data	- Traffic	TRAFFIC.FRM
Tables	- County Name	CNTYTYBL.FRM
	- Material Type	MATLTBL.FRM

Reports (Using dBASE III Procedures): (continued)

	REPORT FILES
- Type of Pavement	PAVETYPE.FRM
- District Temperature Constant	DISTTEMP.FRM
- Widening Flag	WIDENFLG.FRM
- Functional Classification	FUNCLTBL.FRM
- Layer Description	LAYERTBL.FRM

REPORT SCREENS

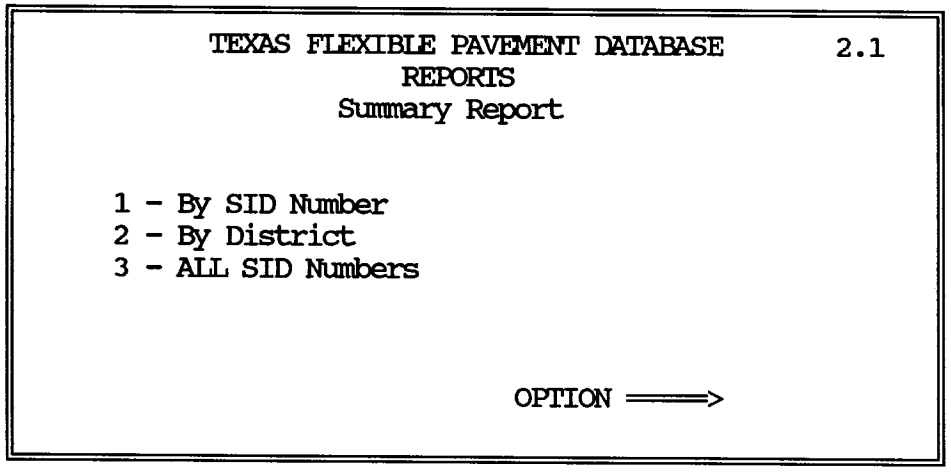
Other than the Main Menu, all of the screens illustrated in this section are produced by REPORTS.PRG.

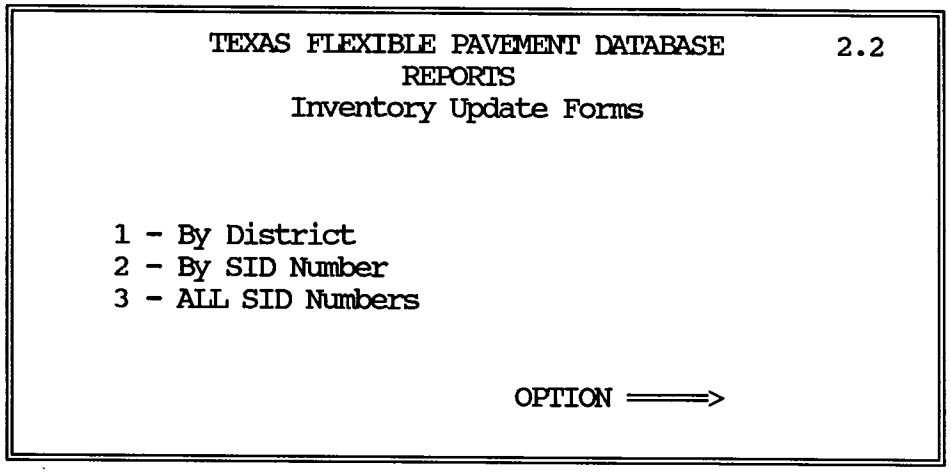
```
        TEXAS FLEXIBLE PAVEMENT DATABASE
                    MAIN MENU


          1 - Inquiry
          2 - Reports
          3 - Edit & Update
          4 - Applications
          5 - Backup
          6 - Installation
          7 - Reindex Master Files

          Q - QUIT

                                  OPTION ===>
```

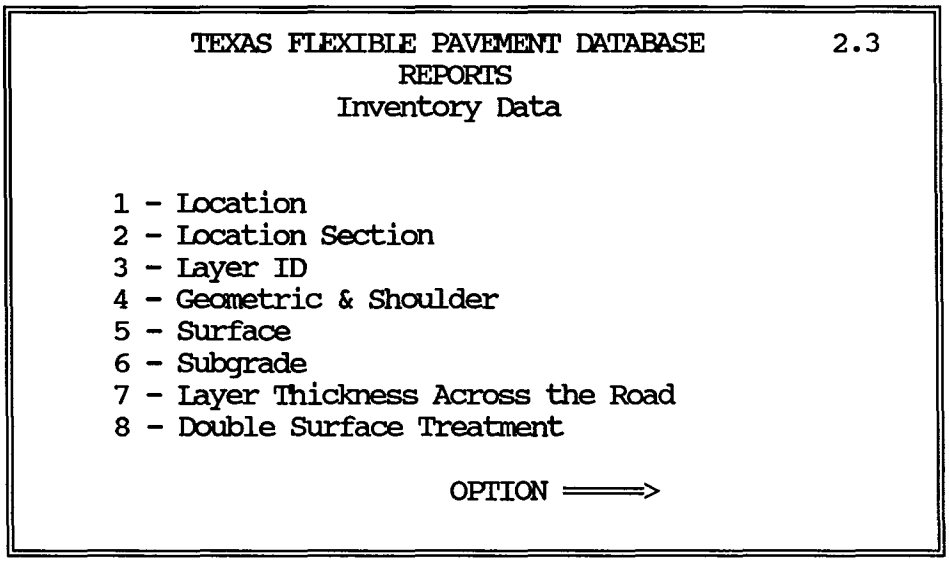
Above is the Main Menu of the Texas Flexible Pavement Database System. When the Reports option is chosen, the next screen 2.0 is displayed. Choice 1 - Summary displays the screen 2.1 on the next page while choice 2 - Inventory Update Forms displays screen 2.2.

```
        TEXAS FLEXIBLE PAVEMENT DATABASE      2.0
                    Reports


          1 - Summary
          2 - Inventory Update Forms
          3 - Inventory Data
          4 - Monitoring Data
          5 - Traffic Data
          6 - Environmental Data
          7 - Tables


                          OPTION ===>
```

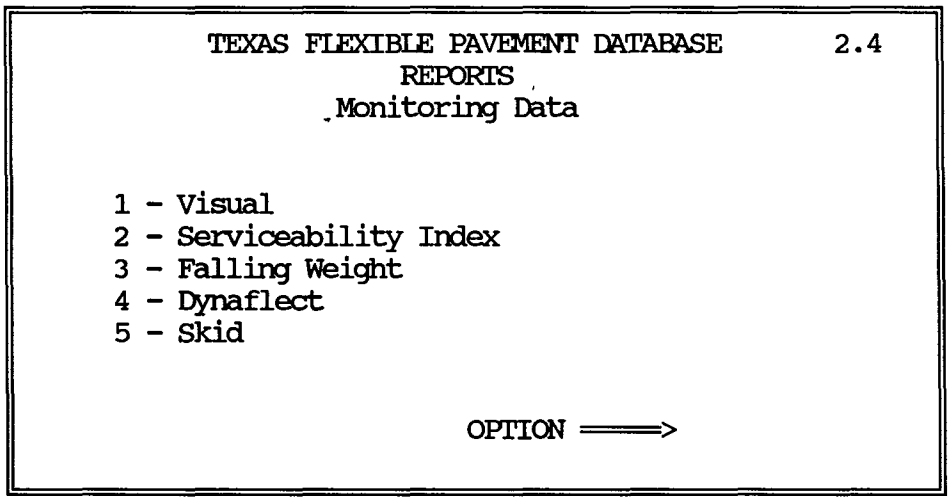
REPORT SCREENS (continued)







Choice 3 - Inventory Data on the Reports Menu brings up the above screen 2.3, while choice 4 - Monitoring Data brings up the next screen 2.4.



```
          TEXAS FLEXIBLE PAVEMENT DATABASE          2.6
                       REPORTS
                  Environmental Data


    1 - Environment
    2 - Weather

                                   OPTION ===========>
```

Choice 6 - Environmental Data displays the above screen 2.6 while choice 7 - Tables displays the next screen 2.7.

```
          TEXAS FLEXIBLE PAVEMENT DATABASE          2.7
                       REPORTS
                         Tables


    1 - County Name
    2 - Material Type
    3 - Type of Pavement
    4 - District Temperature Constant
    5 - Widening Flag
    6 - Layer Description
    7 - Functional Classification

                                   OPTION ===========>
```

REPORT SUBSYSTEM
SAMPLE REPORTS

Texas Flexible Pavement Database
Location File

SID No.	Actv Flag	Highway District	County Number	Beg. Mile Post	End Mile Post	Highway Number	Lane ID	Control/ Section	Milepoint Begn to End	Mile Point Date	Prev SID	Next SID	Inactv Date	No. of Lanes	Comments
13	.T.	1	92	22+00	24+00	US 82	R	45/4	22.000 TO 24.000	6/75	0	0	0/ 0	1	
26	.T.	1	92	4+00	6+00	FM 2729	R	2798/3	10.720 TO 12.720	6/75	0	0	0/ 0	1	
39	.T.	1	117	106+00	109+00	IH 30	R	9/13	27.800 TO 29.800	11/76	0	0	0/ 0	2	
42	.T.	1	117	28+00	30+00	SH 34	R	173/6	0.026 TO 1.850	11/76	0	0	0/ 0	1	
55	.T.	1	117	2+00	4+00	FM 1566	L	1495/1	2.000 TO 3.980	6/75	0	0	0/ 0	1	
68	.T.	1	117	2-20	2+00	FM 2736	R	2732/1	0.000 TO 2.010	2/76	0	0	0/ 0	1	
71	.T.	1	139	6+00	8+00	US 271	R	136/8	5.620 TO 7.560	6/75	0	0	0/ 0	2	
84	.F.	1	139	14+00	14+16	FM 905	R	730/3	14.790 TO 16.440	11/76	0	5110	6/81	1	RECONSTRUCTION
97	.T.	1	139	14+00	16+00	FM 79	L	688/2	14.000 TO 16.000	6/75	0	0	0/ 0	1	
102	.T.	1	190	4+11	8+00	US 69	R	203/3	6.140 TO 8.140	6/75	0	0	0/ 0	1	
115	.T.	1	190	2+00	4+00	FM 779	R	2606/1	2.000 TO 4.000	11/76	0	0	0/ 0	1	
128	.T.	2	73	8+00	10+00	SH 6	R	258/1	8.000 TO 10.000	6/75	0	0	0/ 0	1	
131	.T.	2	73	4+00	6+00	FM 2157	R	1990/1	4.000 TO 6.000	6/75	0	0	0/ 0	1	
144	.T.	2	120	36+00	38+00	US 281	L	249/7	41.100 TO 43.100	6/75	0	0	0/ 0	1	
157	.T.	2	120	6+00	4+00	FM 206	L	391/7	4.000 TO 6.000	2/76	0	0	0/ 0	1	
160	.T.	2	127	28+00	30+00	US 67	R	259/4	2.790 TO 4.790	6/75	0	0	0/ 0	1	
173	.F.	2	127	2+00	4+00	FM 917	R	1181/2	2.000 TO 4.000	6/75	0	5089	4/78	1	RECONSTRUCTION
186	.T.	2	220	0+00	2+00	US 377	R	80/7	8.591 TO 10.589	6/77	0	0	0/ 0	1	
199	.T.	2	220	0+00	2+00	SH 303S	L	2208/1	12.144 TO 14.144	6/77	0	0	0/ 0	2	
204	.T.	2	220	2+00	4+00	FM 1709	R	1603/3	2.000 TO 4.000	4/76	0	0	0/ 0	1	
217	.T.	2	127	18+00	20+00	IH 35W	R	14/4	4.680 TO 6.680	2/76	0	0	0/ 0	2	
220	.T.	3	39	4+00	6+00	SH 79	L	282/2	3.940 TO 6.330	2/76	0	0	0/ 0	1	
233	.T.	3	39	10+00	12+00	FM 1197	R	1350/1	9.960 TO 11.940	6/75	0	0	0/ 0	1	
246	.T.	3	169	20+00	22+00	SH 59	L	239/2	16.950 TO 18.950	6/75	0	0	0/ 0	1	
259	.T.	3	169	6+00	8+00	FM 455	R	845/1	4.010 TO 5.970	11/76	0	0	0/ 0	1	
262	.T.	3	224	34+00	34+20	US 183	L	404/1	33.840 TO 35.830	6/75	0	0	0/ 0	1	MISSING INVENTORY FOLDER SINCE 3/87
275	.T.	3	224	0-01	2+00	FM 2651	R	2645/1	6.000 TO 7.750	2/76	0	0	0/ 0	1	
288	.F.	3	244	28+00	30+00	US 183	L	147/1	4.170 TO 5.570	6/75	0	0	1/58	1	WIDENING FLAG = 2
291	.T.	3	244	10+00	10+20	FM 91	R	702/1	9.940 TO 11.930	6/75	0	0	0/ 0	1	
306	.T.	4	33	104+00	106+00	IH 40	L	275/4	0.174 TO 1.180	2/76	0	0	0/ 0	2	
319	.T.	4	33	26+00	26+18	US 60	L	169/5	7.100 TO 8.860	10/75	0	0	0/ 0	2	
322	.T.	4	33	10+00	12+08	FM 1342	L	1884/1	9.700 TO 11.630	6/75	0	0	0/ 0	1	
335	.T.	4	104	4+00	6+00	US 87	L	41/1	5.070 TO 7.010	6/75	0	0	0/ 0	1	
348	.T.	4	104	2+00	4+00	FM 998	R	1622/2	2.040 TO 4.000	6/75	0	0	0/ 0	1	
351	.T.	4	118	6+00	8+00	SH 152	L	557/2	6.023 TO 7.870	2/76	0	0	0/ 0	1	
364	.T.	4	118	2-18	2+00	FM 1598	L	1515/3	1.488 TO 3.328	10/76	0	0	0/ 0	1	
377	.T.	4	148	2+00	4+00	SH 305	R	582/1	11.980 TO 13.890	6/75	0	0	0/ 0	1	
380	.T.	4	148	28+00	30+01	SH 23	R	1337/2	26.700 TO 28.550	5/81	0	0	0/ 0	1	
393	.T.	4	180	20+00	22+00	IH 40	R	90/3	5.430 TO 7.430	2/76	0	0	0/ 0	2	
408	.T.	4	180	4+00	6+00	US 385	L	226/2	4.930 TO 6.930	6/75	0	0	0/ 0	1	
411	.T.	4	180	4+00	4+16	SH 214	L	461/13	8.950 TO 10.575	5/81	0	0	0/ 0	1	
424	.T.	4	104	34+00	36+00	US 54	R	238/2	14.640 TO 16.580	6/75	0	0	0/ 0	1	
437	.T.	5	96	24+00	26+00	US 87	R	67/6	24.230 TO 26.230	2/76	0	0	0/ 0	2	
440	.T.	5	96	6+00	8+00	SH 194	R	439/4	6.000 TO 8.000	6/75	0	0	0/ 0	1	
453	.T.	5	96	26+00	28+00	FM 400	R	1041/1	35.360 TO 37.360	6/75	0	0	0/ 0	1	
466	.T.	5	96	0+00	2+00	FM 1612	L	2332/2	0.000 TO 2.000	6/75	0	0	0/ 0	1	

EXHIBIT 1

SAMPLE LOCATION FILE

Texas Flexible Database
Test Section Location

Standard ID Number	Highway District	County Number	Highway Number	Beginning Milepost	Ending Milepost
13	1	92	US 82	22+00	24+00
26	1	92	FM2729	4+00	6+00
39	1	117	IH 30	106+00	109+00
42	1	117	SH 34	28+00	30+00
55	1	117	FM1566	2+00	4+00
68	1	117	FM2736	2-20	2+00
71	1	139	US 271	6+00	8+00
84	1	139	FM 905	14+00	14+16
97	1	139	FM 79	14+00	16+00
102	1	190	US 69	4+11	8+00
115	1	190	FM 779	2+00	4+00
128	2	73	SH 6	8+00	10+00
131	2	73	FM2157	4+00	6+00
144	2	120	US 281	36+00	38+00
157	2	120	FM 206	6+00	4+00
160	2	127	US 67	28+00	30+00
173	2	127	FM 917	2+00	4+00
186	2	220	US 377	0+00	2+00
199	2	220	SH 303S	0+00	2+00
204	2	220	FM1709	2+00	4+00
217	2	127	IH 35W	18+00	20+00
220	3	39	SH 79	4+00	6+00
233	3	39	FM1197	10+00	12+00
246	3	169	SH 59	20+00	22+00
259	3	169	FM 455	6+00	8+00
262	3	224	US 183	34+00	34+20
275	3	224	FM2651	0-01	2+00
288	3	244	US 183	28+00	30+00
291	3	244	FM 91	10+00	10+20
306	4	33	IH 40	104+00	106+00
319	4	33	US 60	26+00	26+18
322	4	33	FM1342	10+00	12+08
335	4	104	US 87	4+00	6+00
348	4	104	FM 998	2+00	4+00
351	4	118	SH 152	6+00	8+00
364	4	118	FM1598	2-18	2+00
377	4	148	SH 305	2+00	4+00

SAMPLE TEST LOCATION FILE REPORT

EXHIBIT 2

Texas Flexible Database
Layer Identification File

SID Number	Struc No.	Layer Number	Layer Description	Layer Center Thickness	Material Type Class.	Date Job Completed	Date Layer Widened
13	1	1	7	0.00	44	6/31	0/ 0
13	1	2	5	6.00	17	6/31	8/46
13	1	3	4	2.00	1	7/47	0/ 0
13	1	4	3	1.30	1	1/63	0/ 0
13	2	5	2	0.40	11	8/79	0/ 0
26	1	1	7	0.00	44	10/65	0/ 0
26	1	2	5	6.00	21	10/65	0/ 0
26	1	3	5	5.20	21	6/66	0/ 0
26	1	4	10	0.36	6	6/66	0/ 0
26	1	5	3	0.25	6	6/66	0/ 0
39	1	1	7	0.00	44	9/52	0/ 0
39	1	2	5	6.00	31	9/52	0/ 0
39	1	3	3	10.00	17	9/52	9/52
39	1	4	1	2.50	1	8/67	0/ 0
39	1	5	3	1.50	1	8/67	0/ 0
42	1	1	7	0.00	44	6/45	0/ 0
42	1	2	5	6.00	25	6/45	0/ 0
42	1	3	10	0.25	5	6/45	0/ 0
42	1	4	5	4.00	21	12/54	0/ 0
42	1	5	10	0.26	6	12/54	0/ 0
42	1	6	10	0.26	6	12/54	0/ 0
42	1	7	5	4.00	21	7/64	0/ 0
42	1	8	10	0.36	6	7/64	0/ 0
42	1	9	10	0.25	6	7/64	0/ 0
42	1	10	3	0.30	11	9/72	0/ 0
42	2	11	2	0.40	11	7/82	0/ 0
55	1	1	7	0.00	44	9/51	0/ 0
55	1	2	5	4.00	21	9/51	0/ 0
55	1	3	10	0.20	6	9/51	0/ 0
55	1	4	10	0.20	6	9/51	0/ 0
55	1	5	2	0.20	11	7/56	0/ 0
55	1	6	3	0.33	11	6/63	0/ 0
68	1	1	7	0.00	44	9/65	0/ 0
68	1	2	5	6.00	21	9/65	0/ 0
68	1	3	10	0.36	6	9/65	0/ 0
68	1	4	3	0.25	6	9/65	0/ 0

SAMPLE LAYER IDENTIFICATION FILE REPORT

EXHIBIT 3

Page No. 1
11/09/88

Texas Flexible Pavement Database

Section Identification Numbers with
Two Course Surface Treatment (DST)

Section Identification Number	Date Job Completed (Month/Year)
26	6/66
26	6/66
42	12/54
42	12/54
42	7/64
42	7/64
55	9/51
55	9/51
68	9/65
68	9/65
84	6/71
84	6/71
97	8/55
102	6/66
102	6/66
115	7/62
115	7/62
128	5/40
128	5/40
144	8/53
144	8/53
160	5/41
160	5/41
186	7/47
186	7/47
199	9/60
199	9/60
217	8/66
217	8/66
220	11/41
220	11/41
246	5/42
246	5/42
259	6/49

SAMPLE SECTION IDENTIFICATION NUMBERS
WITH TWO COURSE SURFACE TREATMENT (DST) REPORT

EXHIBIT 4

Texas Flexible Pavement Database
Layer Thickness Across The Road File

SID No	Struc No	Layr No	Thickness From Center 3rd Pos	Thickness From Center 2nd Pos	Thickness From Center 1st Pos	Thick From Centr	Distance From Center 3rd Pos	Distance From Center 2nd Pos	Distance From Center 1st Pos
13	1	2	10.0	10.0	6.0	6.0	12.0	9.0	5.0
13	1	3	0.0	0.0	2.0	2.0	0.0	0.0	12.0
13	1	4	0.0	0.0	1.3	1.3	0.0	0.0	12.0
13	2	5	0.0	0.0	0.4	0.4	0.0	0.0	12.0
26	1	2	0.0	0.0	6.0	6.0	0.0	0.0	10.0
26	1	3	0.0	0.0	5.2	5.2	0.0	0.0	10.0
26	1	4	0.0	0.0	0.4	0.4	0.0	0.0	10.0
26	1	5	0.0	0.0	0.3	0.3	0.0	0.0	10.0
39	1	2	0.0	6.0	6.0	6.0	29.0	12.0	12.0
39	1	3	0.0	0.0	10.0	10.0	0.0	0.0	12.0
39	1	4	0.0	0.0	2.5	2.5	0.0	0.0	12.0
39	1	5	0.0	0.0	1.5	1.5	0.0	0.0	12.0
42	1	2	0.0	0.0	6.0	6.0	0.0	0.0	13.0
42	1	3	0.0	0.0	0.3	0.3	0.0	0.0	11.0
42	1	4	0.0	0.0	4.0	4.0	0.0	0.0	12.0
42	1	5	0.0	0.0	0.3	0.3	0.0	0.0	12.5
42	1	6	0.0	0.0	0.3	0.3	0.0	0.0	12.0
42	1	7	0.0	0.0	4.0	4.0	0.0	0.0	22.0
42	1	8	0.0	0.0	0.4	0.4	0.0	0.0	22.0
42	1	9	0.0	0.0	0.3	0.3	0.0	0.0	13.0
42	1	10	0.0	0.0	0.3	0.3	0.0	0.0	13.0
42	2	11	0.0	0.0	0.4	0.4	0.0	0.0	13.0
55	1	2	0.0	0.0	4.0	4.0	0.0	0.0	9.5
55	1	3	0.0	0.0	0.2	0.2	0.0	0.0	9.0
55	1	4	0.0	0.0	0.2	0.2	0.0	0.0	9.0
55	1	5	0.0	0.0	0.2	0.2	0.0	0.0	9.0
55	1	6	0.0	0.0	0.3	0.3	0.0	0.0	9.0
68	1	2	0.0	0.0	6.0	6.0	0.0	0.0	10.0
68	1	3	0.0	0.0	0.4	0.4	0.0	0.0	10.0
68	1	4	0.0	0.0	0.3	0.3	0.0	0.0	10.0
71	1	2	0.0	6.0	6.0	6.0	27.0	12.0	12.0
71	1	3	0.0	0.0	9.0	9.0	0.0	0.0	12.0
71	1	4	0.0	0.0	2.0	2.0	0.0	0.0	12.0
71	1	5	0.0	0.0	1.5	1.5	0.0	0.0	12.0
84	1	2	0.0	0.0	4.0	4.0	0.0	0.0	12.5
84	1	3	0.0	0.0	4.0	4.0	0.0	0.0	11.0

SAMPLE LAYER THICKNESS ACROSS THE ROAD
FILE REPORT

EXHIBIT 5

Texas Flexible Pavement Database
Geometric And Shoulder Information

SID No	Struc No	Type Of Pavmnt	Lane Width	Outside Shouldr Width	Shoulder Surface Type	Shoulder Base Type	Shoulder Surface Thicknss	Shoulder Base Thicknss	Widen Flag
13	1	34	12.0	0.0	1	0	0.00	0.00	1
13	2	35	12.0	0.0	1	0	0.00	0.00	1
26	1	1	10.0	0.0	1	0	0.00	0.00	0
39	1	34	12.0	17.0	2	21	0.00	8.00	1
42	1	1	12.0	10.0	5	21	0.36	8.00	0
42	2	1	12.0	10.0	5	21	0.36	8.00	0
55	1	1	9.0	0.0	1	0	0.00	0.00	0
68	1	1	10.0	0.0	1	0	0.00	0.00	0
71	1	34	12.0	15.0	2	31	0.00	3.70	1
84	1	1	10.0	0.0	1	0	0.00	0.00	0
97	1	1	10.0	0.0	1	0	0.00	0.00	0
102	1	12	12.0	10.0	5	21	0.36	6.00	0
102	2	13	12.0	10.0	5	21	0.36	6.00	0
115	1	21	10.0	0.0	1	0	0.00	0.00	0
128	1	1	11.0	6.0	2	21	0.00	6.00	0
128	2	1	11.0	6.0	5	21	0.30	6.00	0
128	3	1	11.0	6.0	5	21	0.60	6.00	0
131	1	1	10.0	3.0	2	21	0.00	6.00	0
131	2	1	10.0	3.0	2	21	0.00	6.00	0
131	3	1	10.0	3.0	2	21	0.00	6.00	0
144	1	1	12.0	7.0	2	21	0.00	9.00	0
144	2	1	12.0	7.0	2	21	0.00	6.00	0
144	3	2	12.0	7.0	2	21	0.00	6.00	0
157	1	1	8.0	4.0	2	21	0.00	3.00	0
157	2	1	10.0	0.0	1	0	0.00	0.00	0
157	3	1	10.0	0.0	1	0	0.00	0.00	0
160	1	2	12.0	10.0	2	21	0.00	9.00	1
160	2	3	12.0	10.0	2	21	0.00	9.00	1
173	1	1	9.0	7.0	2	21	0.00	4.00	0
173	2	1	9.0	7.0	2	21	0.00	4.00	0
173	3	1	9.0	7.0	2	21	0.00	4.00	0
186	1	1	12.0	8.0	2	21	0.00	6.50	0
186	2	4	12.0	8.0	2	21	0.00	6.50	0
199	1	4	12.0	4.3	2	21	0.00	6.57	0
199	2	4	12.0	4.3	2	21	0.00	6.57	0
199	3	4	12.0	4.3	2	21	0.00	6.57	0

SAMPLE GEOMETRIC AND SHOULDER INFORMATION
FILE REPORT

EXHIBIT 6

Texas Flexible Pavement Database
Surface File

SID Number	Structure Number	Layer Number	Aggregate Application Rate	Admixture Type	Percent Admixture	Asphalt Application Rate
13	1	3	0	AC	5.70	0.00
13	1	4	0	AC	5.40	0.00
13	2	5	75	AC-10	0.00	0.45
26	1	4	80	AC-10	0.00	0.30
26	1	5	120	AC-10	0.00	0.25
39	1	3	0		0.00	0.00
39	1	4	0	AC	4.70	0.00
39	1	5	0	AC	4.90	0.00
42	1	3	120	OA-230	0.00	0.30
42	1	5	110	OA-230	0.00	0.22
42	1	6	110	OA-230	0.00	0.22
42	1	8	80	OA-135	0.00	0.30
42	1	9	120	OA-135	0.00	0.25
42	1	10	95	AC-5	0.00	0.25
42	2	11	90	AC-10	0.00	0.35
55	1	3	140	RC-2	0.00	0.30
55	1	4	145	RC-2	0.00	0.20
55	1	5	0		0.00	0.00
55	1	6	85	OA-135	0.00	0.30
68	1	3	80	OA-135	0.00	0.30
68	1	4	120	OA-135	0.00	0.25
71	1	3	0		0.00	0.00
71	1	4	0	AC	4.97	0.00
71	1	5	0	AC	4.60	0.00
84	1	4	80	AC-5	0.00	0.35
84	1	5	120	AC-5	0.00	0.25
97	1	3	0		0.00	0.00
102	1	4	80	OA-135	0.00	0.30
102	1	5	120	OA-135	0.00	0.25
102	1	6	0		0.00	0.00
102	2	7	75	AC-10	0.00	0.40
115	1	3	120	OA-135	0.00	0.30
115	1	4	140	OA-135	0.00	0.20
128	1	3	90	OA-135	0.00	0.20
128	1	4	180	OA-135	0.00	0.30
128	1	5	180	OA-230	0.00	0.25

SAMPLE SURFACE FILE REPORT

EXHIBIT 7

Texas Flexible Pavement Database
Subgrade File

SID Number	Struc Number	Layer Number	Percent Passing No 200 Sieve	Plast. Index	Liquid Limit	Texas Triaxial Class	Permeability Index
13	1	1	88.8	40.4	64.5	5.3	0.23
26	1	1	63.5	27.5	50.1	5.2	0.36
39	1	1	85.7	39.1	61.3	5.6	0.60
42	1	1	79.3	29.7	48.0	5.1	0.06
55	1	1	81.8	33.8	53.6	5.3	0.06
68	1	1	79.3	29.7	48.0	5.1	0.06
71	1	1	70.9	26.7	36.6	4.2	0.27
84	1	1	85.0	41.8	65.5	0.0	0.06
97	1	1	79.1	29.9	48.0	5.6	0.06
102	1	1	78.0	24.0	48.0	4.8	0.06
115	1	1	92.9	35.0	50.0	0.0	0.16
128	1	1	58.5	21.9	42.6	4.7	0.40
131	1	1	41.7	15.5	31.4	0.0	2.19
144	1	1	90.6	23.9	45.5	4.9	0.21
157	1	1	71.9	20.7	41.0	4.8	1.58
160	1	1	82.5	32.4	55.5	4.8	0.23
173	1	1	66.5	24.6	43.7	0.0	0.75
186	1	1	85.8	34.7	56.5	4.9	0.15
199	1	1	55.6	21.3	43.1	4.1	0.64
204	1	1	69.6	25.9	46.3	4.7	0.29
217	1	1	90.0	46.5	69.0	0.0	0.06
220	1	1	51.7	10.2	28.6	4.1	2.16
233	1	1	87.2	28.7	48.6	4.2	0.15
259	1	1	66.2	23.9	40.3	3.8	0.40
275	1	1	85.7	28.6	51.7	4.7	0.33
291	1	1	33.0	8.4	24.9	3.9	3.09
306	1	1	87.5	27.5	47.5	4.5	0.11
319	1	1	87.5	27.5	47.5	4.5	0.11
322	1	1	87.5	27.5	47.5	4.5	0.11
335	1	1	51.3	12.8	28.8	3.9	1.30
348	1	1	51.3	12.8	28.8	4.2	1.30
351	1	1	55.4	9.1	26.9	3.8	1.05
364	1	1	80.3	22.2	41.2	4.4	0.57
377	1	1	87.5	27.5	47.5	4.5	0.11
380	1	1	71.4	20.2	37.9	5.0	0.48
393	1	1	87.5	27.5	47.5	4.5	0.11

SAMPLE SUBGRADE FILE REPORT

EXHIBIT 8

Texas Flexible Pavement Database
Visual Rating File

SID Num.	Struc No.	Layr Num.	Year	Act. Year	Act. Mnth	Rutt S	Rutt M	Rutt S	Block S	Block M	Block S	Allig Crack S	Allig Crack M	Allig Crack S	Long Crack S	Long Crack M	Long Crack S	Trans Crack S	Trans Crack M	Trans Crack S	Crac Seal Code	Patch G	Patch F	Patch P	Fail Mile	PRS	PES Rating Score	UVU Rating Score
13	1	4	73	73	7	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	2	1	0	0	0	97	0.00	0.00
13	1	4	74	74	8	0	0	0	0	0	0	0	0	0	0	0	3	0	0	2	2	1	0	0	0	65	0.00	0.00
13	1	4	75	75	8	0	0	0	0	0	0	0	0	0	0	0	3	0	0	2	2	1	0	0	0	65	0.00	0.00
13	1	4	76	76	7	0	0	0	0	0	0	0	0	0	0	0	2	0	3	0	2	0	0	0	0	65	0.00	0.00
13	1	4	77	77	8	1	0	0	0	0	0	0	0	0	0	1	0	0	2	0	2	0	0	0	0	70	0.00	0.00
13	2	5	80	80	9	1	0	0	0	0	0	0	0	0	1	0	0	0	1	0	1	0	0	0	0	78	0.00	0.00
26	1	5	73	73	7	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	2	0	0	0	75	0.00	0.00
26	1	5	74	74	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	88	0.00	0.00
26	1	5	75	75	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	80	0.00	0.00
26	1	5	76	76	7	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	92	0.00	0.00
26	1	5	77	77	8	2	0	0	0	0	0	0	0	0	1	0	0	0	0	0	3	0	0	0	0	78	0.00	0.00
26	1	5	80	80	9	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	95	0.00	0.00
26	1	5	87	87	8	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0.00	0.00
39	1	5	74	74	10	2	0	0	0	0	0	0	0	0	1	0	0	0	2	0	3	0	1	0	0	76	0.00	0.00
39	1	5	76	76	7	2	0	0	0	0	0	0	0	0	1	0	0	0	2	0	3	0	0	0	0	76	0.00	0.00
39	1	5	77	77	8	2	0	0	0	0	0	0	0	0	0	1	0	0	1	0	3	0	0	0	0	69	0.00	0.00
39	1	5	80	80	9	0	2	0	0	0	0	0	0	0	0	0	1	0	0	2	3	0	0	0	0	63	0.00	0.00
39	1	5	87	87	9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.13	1.00
42	1	10	73	73	7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	92	0.00	0.00
42	1	10	74	74	10	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	88	0.00	0.00
42	1	10	75	75	8	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	80	0.00	0.00
42	1	10	76	76	7	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	70	0.00	0.00
42	1	10	77	78	3	3	0	0	0	0	0	0	0	1	1	0	0	0	0	0	3	0	0	1	1	36	0.00	0.00
42	1	10	80	80	9	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	92	0.00	0.00
55	1	6	73	73	7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	95	0.00	0.00
55	1	6	74	74	10	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	95	0.00	0.00
55	1	6	75	75	8	0	0	0	0	0	0	2	0	0	2	0	0	0	0	0	3	0	2	0	0	68	0.00	0.00
55	1	6	76	76	7	3	0	0	0	0	0	2	0	0	1	0	0	0	0	0	2	0	2	0	0	67	0.00	0.00

EXHIBIT 9

SAMPLE VISUAL RATING FILE

Texas Flexible Pavement Database
Serviceability Index File

Section Ident. Number	Struc No	Layr No.	Actl Yr	Act Mnth	Act Day	Actl Year	Count	Mean	Stand. Deviat.	Low Val	High Val.
13	2	5	80	9	16	80	8	3.17500	0.38079	2.5	3.6
13	1	4	77	8	5	77	10	3.50000	0.29440	3.0	3.8
13	1	4	76	7	22	76	10	3.36000	0.35650	2.7	3.9
13	1	4	75	8	24	75	10	3.42000	0.28210	2.9	3.8
13	1	4	74	8	27	74	10	3.38000	0.26600	3.0	3.8
13	1	4	73	7	19	73	10	3.35000	0.32400	2.6	3.8
26	1	5	80	9	16	80	10	2.52999	0.66508	1.4	3.4
26	1	5	77	8	5	77	10	4.02000	0.12290	3.8	4.2
26	1	5	76	7	22	76	10	3.95000	0.15810	3.6	4.1
26	1	5	75	8	24	75	10	3.73000	0.20570	3.3	4.0
26	1	5	74	8	27	74	4	3.58000	0.26300	3.2	3.8
26	1	5	73	7	19	73	9	3.74000	0.20700	3.4	4.0
39	1	5	80	9	16	80	10	2.66000	0.32042	2.1	3.2
39	1	5	77	8	5	77	10	3.30000	0.35590	2.7	3.8
39	1	5	76	7	22	76	10	3.45000	0.58360	2.1	4.0
39	1	5	74	10	26	74	9	3.44000	0.23000	2.9	3.6
42	1	10	80	9	16	80	10	3.71000	0.47714	2.7	4.1
42	1	10	77	3	12	78	9	4.17780	0.26820	3.7	4.5
42	1	10	76	7	21	76	9	4.13330	0.44160	3.4	4.7
42	1	10	75	8	28	75	9	3.96670	0.49750	3.1	4.5
42	1	10	74	10	26	74	10	3.93000	0.46400	3.1	4.4
42	1	10	73	7	20	73	10	3.83000	0.25800	3.4	4.2
55	1	6	80	9	16	80	10	1.60000	0.30912	1.1	2.2
55	1	6	77	8	5	77	10	2.34000	0.45510	1.4	3.0
55	1	6	76	7	22	76	9	1.72220	0.43900	0.5	2.5
55	1	6	75	8	28	75	10	1.86000	0.47190	0.8	2.3
55	1	6	74	10	26	74	9	2.18000	0.42900	1.2	2.6
55	1	6	73	7	20	73	10	2.18000	0.53700	1.0	2.7
68	1	4	80	9	16	80	10	1.74000	0.67856	0.9	2.8
68	1	4	77	8	5	77	8	2.25000	0.70100	1.5	3.5
68	1	4	76	7	22	76	8	1.27500	0.70650	0.5	2.5
68	1	4	75	8	28	75	7	1.77140	0.72960	1.0	2.9
68	1	4	74	10	26	74	9	2.03000	0.52700	1.4	2.8
68	1	4	73	1	10	74	10	2.16000	0.56000	1.2	3.1

SAMPLE SERVICEABILITY INDEX FILE REPORT

EXHIBIT 10

Texas Flexible Pavement Database
Falling Weight SSI File

SID No.	Str No.	Lyr No.	Date MM/DD/YY	SSI Avg.	SSI Temp	SSI Reading 1 Geophone 1 - 7							SSI Reading 2 Geophone 1 - 7						
26	1	5	0/ 0/ 0	0.0	0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
39	1	5	0/ 0/ 0	0.0	0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
55	1	6	0/ 0/ 0	0.0	0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
68	1	4	0/ 0/ 0	0.0	0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
71	1	5	0/ 0/ 0	0.0	0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
97	1	3	0/ 0/ 0	0.0	0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
102	2	7	0/ 0/ 0	0.0	0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
115	1	4	0/ 0/ 0	0.0	0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
128	3	8	0/ 0/ 0	0.0	0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
144	3	8	0/ 0/ 0	0.0	0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
160	2	6	0/ 0/ 0	0.0	0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
186	2	11	0/ 0/ 0	0.0	0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
199	4	10	0/ 0/ 0	0.0	0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
204	3	8	0/ 0/ 0	0.0	0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
217	2	9	0/ 0/ 0	0.0	0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
220	2	12	0/ 0/ 0	0.0	0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
233	1	6	0/ 0/ 0	0.0	0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
246	2	11	0/ 0/ 0	0.0	0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
259	2	7	0/ 0/ 0	0.0	0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
275	1	5	0/ 0/ 0	0.0	0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
288	3	5	0/ 0/ 0	0.0	0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
291	1	6	0/ 0/ 0	0.0	0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
306	3	9	0/ 0/ 0	0.0	0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
319	2	5	0/ 0/ 0	0.0	0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
322	2	7	0/ 0/ 0	0.0	0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
335	2	11	0/ 0/ 0	0.0	0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
348	2	6	0/ 0/ 0	0.0	0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
351	2	6	0/ 0/ 0	0.0	0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
364	2	7	0/ 0/ 0	0.0	0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
377	2	7	0/ 0/ 0	0.0	0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
380	2	7	0/ 0/ 0	0.0	0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
393	2	10	0/ 0/ 0	0.0	0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
408	1	3	0/ 0/ 0	0.0	0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
411	1	6	0/ 0/ 0	0.0	0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
424	1	5	0/ 0/ 0	0.0	0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
440	2	7	0/ 0/ 0	0.0	0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
453	2	7	0/ 0/ 0	0.0	0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
466	1	5	0/ 0/ 0	0.0	0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
479	1	6	0/ 0/ 0	0.0	0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
482	1	5	0/ 0/ 0	0.0	0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
495	1	5	0/ 0/ 0	0.0	0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
526	1	6	0/ 0/ 0	0.0	0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
539	3	10	0/ 0/ 0	0.0	0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
542	1	4	0/ 0/ 0	0.0	0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
555	1	7	0/ 0/ 0	0.0	0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
568	1	6	0/ 0/ 0	0.0	0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

SAMPLE FALLING WEIGHT SSI REPORT FILE
EXHIBIT 11

Texas Flexible Pavement Database
Falling Weight SSI File

SID No.	Str No.	Lyr No.	Date MM/DD/YY	SSI Reading 3 Geophone 1 - 7							SSI Reading 4 Geophone 1 - 7							SSI Reading 5 Geophone 1 - 7				
26	1	5	0/0/ 0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0
39	1	5	0/0/ 0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0
55	1	6	0/0/ 0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0
68	1	4	0/0/ 0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0
71	1	5	0/0/ 0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0
97	1	3	0/0/ 0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0
102	2	7	0/0/ 0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0
115	1	4	0/0/ 0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0
128	3	8	0/0/ 0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0
144	3	8	0/0/ 0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0
160	2	6	0/0/ 0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0
186	2	11	0/0/ 0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0
199	4	10	0/0/ 0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0
204	3	8	0/0/ 0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0
217	2	9	0/0/ 0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0
220	2	12	0/0/ 0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0
233	1	6	0/0/ 0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0
246	2	11	0/0/ 0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0
259	2	7	0/0/ 0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0
275	1	5	0/0/ 0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0
288	3	5	0/0/ 0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0
291	1	6	0/0/ 0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0
306	3	9	0/0/ 0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0
319	2	5	0/0/ 0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0
322	2	7	0/0/ 0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0
335	2	11	0/0/ 0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0
348	2	6	0/0/ 0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0
351	2	6	0/0/ 0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0
364	2	7	0/0/ 0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0
377	2	7	0/0/ 0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0
380	2	7	0/0/ 0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0
393	2	10	0/0/ 0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0
408	1	3	0/0/ 0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0
411	1	6	0/0/ 0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0
424	1	5	0/0/ 0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0
440	2	7	0/0/ 0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0
453	2	7	0/0/ 0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0
466	1	5	0/0/ 0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0
479	1	6	0/0/ 0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0
482	1	5	0/0/ 0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0
495	1	5	0/0/ 0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0
526	1	6	0/0/ 0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0
539	3	10	0/0/ 0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0
542	1	4	0/0/ 0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0
555	1	7	0/0/ 0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0
568	1	6	0/0/ 0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0

SAMPLE FALLING WEIGHT SSI REPORT FILE
EXHIBIT 11 (Continued)

Texas Flexible Pavement Database
Dynaflect Measurement File

SID No.	Struc No.	Layer Numbr	Year	Month	Day	Station	Sensor Readings 1	2	3	4	5
13	1	4	76	8	10	1	0.570	0.540	0.470	0.380	0.320
13	1	4	76	8	10	2	0.620	0.590	0.530	0.430	0.360
13	1	4	76	8	10	3	0.246	0.231	0.189	0.138	0.102
13	1	4	76	8	10	4	0.252	0.228	0.180	0.126	0.090
13	1	4	76	8	10	5	0.560	0.530	0.460	0.370	0.310
13	1	4	76	8	10	6	0.680	0.650	0.560	0.450	0.370
13	1	4	76	8	10	7	0.450	0.430	0.380	0.300	0.240
13	1	4	76	8	10	8	0.470	0.450	0.390	0.320	0.261
13	1	4	76	8	10	9	0.410	0.380	0.320	0.255	0.195
13	1	4	76	8	10	10	0.430	0.400	0.340	0.252	0.192
13	1	4	76	8	10	11	0.360	0.340	0.290	0.222	0.168
13	1	4	76	8	10	12	0.370	0.350	0.300	0.225	0.168
13	1	4	76	8	10	13	0.550	0.530	0.470	0.380	0.300
13	1	4	76	8	10	14	0.680	0.660	0.570	0.440	0.360
26	1	5	76	8	10	1	0.630	0.410	0.210	0.102	0.053
26	1	5	76	8	10	2	0.550	0.400	0.204	0.096	0.053
26	1	5	76	8	10	3	0.470	0.270	0.108	0.046	0.027
26	1	5	76	8	10	4	0.294	0.132	0.033	0.001	0.001
26	1	5	76	8	10	5	0.310	0.126	0.038	0.001	0.001
26	1	5	76	8	10	6	0.270	0.108	0.029	0.001	0.001
26	1	5	76	8	10	7	0.273	0.096	0.028	0.001	0.001
26	1	5	76	8	10	8	0.320	0.165	0.063	0.028	0.001
26	1	5	76	8	10	9	0.650	0.440	0.279	0.183	0.138
26	1	5	76	8	10	10	0.690	0.440	0.249	0.153	0.120
26	1	5	76	8	10	11	0.730	0.450	0.228	0.108	0.050
26	1	5	76	8	10	12	0.660	0.360	0.171	0.078	0.033
26	1	5	76	8	10	13	0.420	0.276	0.126	0.061	0.030
26	1	5	76	8	10	14	0.470	0.297	0.141	0.052	0.026
39	1	5	76	8	9	1	0.500	0.470	0.440	0.380	0.330
39	1	5	76	8	9	2	0.530	0.500	0.470	0.400	0.340
39	1	5	76	8	9	3	0.740	0.700	0.660	0.590	0.530
39	1	5	76	8	9	4	0.790	0.760	0.720	0.660	0.600
39	1	5	76	8	9	5	0.530	0.500	0.470	0.420	0.360
39	1	5	76	8	9	6	0.510	0.480	0.450	0.390	0.350
39	1	5	76	8	9	7	0.600	0.560	0.520	0.460	0.420
39	1	5	76	8	9	8	0.550	0.520	0.470	0.400	0.340

SAMPLE DYNAFLECT MEASUREMENT FILE REPORT

EXHIBIT 12

Texas Flexible Pavement Database
Monitoring Data
Skid Measurement Data

Section Identification Number	Structure Number	Layer Number	Year	Month	Skid Number MEAN	Skid Number HIGH	Skid Number LOW
13	1	4	71	4	38	42	32
13	1	4	74	7	39	49	27
26	1	5	74	8	33	44	22
84	1	5	75	7	31	39	25
102	1	6	71	4	31	0	0
102	1	6	74	8	28	30	23
115	1	4	75	3	26	34	20
128	1	6	75	2	32	38	27
128	1	6	76	9	23	26	20
131	1	4	76	9	55	65	41
144	2	7	75	1	16	24	8
160	1	5	74	11	36	42	32
160	1	5	76	8	28	31	25
173	3	8	76	8	26	48	12
186	1	8	74	11	33	36	29
186	1	8	76	8	30	44	19
220	1	11	75	7	42	60	25
220	1	11	75	10	43	50	37
233	1	6	75	7	37	57	20
233	1	6	75	12	35	42	28
246	0	0	74	8	16	19	13
259	1	6	75	12	30	42	21
275	1	5	76	4	35	48	21
291	1	6	76	5	33	43	20
335	2	11	76	8	48	49	46
351	2	6	76	7	40	46	34
424	1	5	76	8	47	50	45
440	1	6	75	4	39	54	22
440	1	6	75	4	40	52	24
453	1	6	76	1	36	55	18
453	1	6	76	1	34	51	19
479	1	6	74	6	25	29	18
479	1	6	74	6	24	33	19
526	1	6	75	4	39	47	32
526	1	6	75	4	41	48	33

SAMPLE SKID MEASUREMENT DATA REPORT

EXHIBIT 13

Texas Flexible Pavement Database
Traffic File

Section Identification Number	Year	Average Annual Daily Traffic (ONE-WAY)	Annual 18KIP Equivalent Axis Loads (ONE-WAY)	Percent Trucks
13	1933	386	54259	17.7
13	1934	486	63188	17.7
13	1935	586	71027	17.7
13	1936	686	78009	17.7
13	1937	702	79091	17.7
13	1938	719	80187	17.7
13	1939	736	81233	17.7
13	1940	792	84689	17.7
13	1941	849	87994	17.7
13	1942	906	91108	17.7
13	1943	912	91457	17.7
13	1944	919	91830	17.7
13	1945	926	92175	17.7
13	1946	986	95269	17.7
13	1947	1046	98217	17.7
13	1948	1106	101031	17.7
13	1949	1206	105453	17.7
13	1950	1306	109576	17.7
13	1951	1406	113436	17.7
13	1952	1494	116656	17.7
13	1953	1582	119692	17.7
13	1954	1670	122594	17.7
13	1955	1660	122274	17.7
13	1956	1650	121951	17.7
13	1957	1640	121627	17.7
13	1958	1536	118107	17.7
13	1959	1430	114364	17.7
13	1960	1326	110368	17.7
13	1961	1349	111286	17.7
13	1962	1372	112171	17.7
13	1963	1396	113061	17.7
13	1964	1502	116939	17.7
13	1965	1609	120593	17.7
13	1966	1716	124016	17.7
13	1967	1806	126773	17.7

SAMPLE TRAFFIC FILE REPORT

EXHIBIT 14

Texas Flexible Pavement Database
Environmental Data
Weather File

CNTY No.	Mth	PREC YRS	PRECMN	PRECSD	TFTC YRS	TFTCMN	TFTCSD	WFTC YRS	WFTCMN	WFTCSD	MTMP YRS	MTMPMN	MTMPSD	ATMP YRS	ATMPMN	ATMPSD
1	1	19	3.179	2.053	18	9.111	3.160	18	1.889	1.183	18	55.611	3.791	18	45.056	3.369
1	2	18	3.042	1.392	18	5.111	3.142	18	1.111	1.278	18	61.056	3.718	18	49.667	3.614
1	3	20	3.254	2.100	20	1.700	1.949	20	0.250	0.550	20	68.100	5.015	20	56.650	4.344
1	4	20	4.640	2.890	20	0.000	0.000	20	0.000	0.000	20	76.900	2.936	20	66.300	2.831
1	5	20	4.638	3.623	20	0.000	0.000	20	0.000	0.000	20	83.750	1.860	20	73.200	1.766
1	6	20	3.976	3.270	20	0.000	0.000	20	0.000	0.000	20	89.850	2.207	20	79.150	1.631
1	7	20	1.808	1.645	20	0.000	0.000	20	0.000	0.000	20	94.600	3.033	20	82.900	2.150
1	8	20	3.309	2.442	20	0.000	0.000	20	0.000	0.000	20	94.450	3.086	20	82.200	2.118
1	9	19	4.066	2.305	19	0.000	0.000	19	0.000	0.000	19	88.000	3.543	19	76.632	2.608
1	10	20	4.098	3.431	20	0.000	0.000	20	0.000	0.000	20	79.900	3.386	20	67.500	2.705
1	11	20	3.249	1.711	20	1.700	2.155	20	0.200	0.410	20	68.100	3.582	20	56.600	3.440
1	12	20	3.568	1.939	20	6.500	3.547	20	1.050	1.820	20	59.850	3.717	20	48.900	3.655
2	1	20	0.484	0.565	20	16.700	4.932	20	0.850	1.137	20	58.300	4.054	20	43.250	3.307
2	2	19	0.449	0.469	20	10.400	4.784	19	0.684	0.885	20	62.550	4.212	20	47.250	3.522
2	3	20	0.550	0.816	20	5.200	3.888	20	0.450	0.826	20	70.900	5.281	20	54.850	4.416
2	4	20	0.677	0.727	20	0.500	1.318	20	0.000	0.000	20	80.300	3.063	20	64.400	2.927
2	5	20	1.390	0.843	20	0.000	0.000	20	0.000	0.000	20	88.250	2.954	20	72.900	2.614
2	6	20	1.388	1.205	20	0.000	0.000	20	0.000	0.000	20	94.100	2.614	20	79.650	2.777
2	7	20	2.564	1.586	20	0.000	0.000	20	0.000	0.000	20	94.900	2.732	20	81.450	2.482
2	8	20	1.935	1.695	20	0.000	0.000	20	0.000	0.000	20	93.150	3.345	20	79.900	2.918
2	9	20	1.965	1.858	20	0.000	0.000	20	0.000	0.000	20	86.900	3.782	20	73.950	2.819
2	10	20	1.589	1.571	20	0.000	0.000	20	0.000	0.000	20	78.550	3.692	20	64.450	2.351
2	11	20	0.335	0.426	20	4.500	3.035	20	0.400	0.681	20	66.300	4.219	20	52.400	3.152
2	12	20	0.307	0.348	20	13.600	4.935	20	0.800	1.642	20	59.700	3.743	20	45.350	2.777
3	1	20	3.619	2.852	20	8.350	3.183	20	1.150	1.089	20	59.000	3.524	20	48.100	3.355
3	2	20	3.145	1.440	20	4.800	3.105	20	0.550	0.887	20	62.900	4.229	20	51.350	4.017
3	3	20	3.110	1.890	20	1.600	1.536	20	0.100	0.308	20	70.850	4.716	20	58.700	4.144
3	4	20	4.567	2.590	20	0.000	0.000	20	0.000	0.000	20	79.100	2.674	20	67.750	2.900
3	5	20	3.807	1.940	20	0.000	0.000	20	0.000	0.000	20	85.450	1.469	20	74.450	1.433
3	6	20	3.744	2.664	20	0.000	0.000	20	0.000	0.000	20	90.500	1.821	20	79.750	1.410
3	7	20	2.652	1.766	20	0.000	0.000	20	0.000	0.000	20	93.950	2.439	20	83.050	1.539
3	8	20	2.721	1.790	20	0.000	0.000	20	0.000	0.000	20	93.700	2.618	20	82.350	1.631
3	9	18	4.398	3.101	19	0.000	0.000	18	0.000	0.000	19	88.579	2.735	19	77.632	2.034
3	10	20	3.378	2.615	20	0.050	0.224	20	0.000	0.000	20	81.050	2.893	20	68.100	2.490
3	11	20	3.785	2.884	20	2.350	2.601	20	0.050	0.224	20	69.800	3.680	20	57.700	3.658
3	12	20	4.318	2.338	20	5.700	2.958	20	0.300	0.571	20	62.350	3.924	20	50.900	3.582
4	1	20	1.973	1.471	20	1.850	1.843	20	0.400	0.883	20	62.400	3.500	20	54.050	3.605
4	2	20	2.246	1.378	20	0.450	0.686	20	0.050	0.224	20	65.900	3.684	20	57.600	3.858
4	3	20	1.216	1.457	20	0.050	0.224	20	0.000	0.000	20	70.950	3.547	20	63.250	3.323
4	4	20	1.904	2.211	20	0.000	0.000	20	0.000	0.000	20	77.550	2.800	20	71.000	2.616
4	5	20	3.684	2.458	20	0.000	0.000	20	0.000	0.000	20	83.200	2.463	20	76.900	1.714
4	6	20	4.888	3.986	20	0.000	0.000	20	0.000	0.000	20	87.650	2.456	20	81.550	1.395
4	7	20	1.635	1.699	20	0.000	0.000	20	0.000	0.000	20	89.950	2.666	20	83.650	1.309
4	8	20	3.869	2.755	20	0.000	0.000	20	0.000	0.000	20	90.450	2.856	20	83.350	1.565
4	9	19	6.765	5.295	19	0.000	0.000	19	0.000	0.000	19	87.789	2.175	19	80.421	1.982
4	10	20	4.278	3.726	20	0.000	0.000	20	0.000	0.000	20	81.550	2.373	20	73.600	2.501

SAMPLE WEATHER FILE REPORT
EXHIBIT 15

Texas Flexible Pavement Database
Environmental Data
Environment Data

County No	Thornthwaite Index Mean	Thornthwaite Index Std. Dev.	Thornthwaite Index Years
1	12.510	26.102	20
2	-39.322	7.059	20
3	11.692	23.917	20
4	-10.252	18.861	20
5	-16.158	12.329	20
6	-17.534	11.719	20
7	-22.308	16.070	20
8	4.458	23.532	20
9	-21.861	8.537	20
10	-11.434	17.491	20
11	-10.698	18.953	20
12	-18.964	10.214	20
13	-15.863	13.017	20
14	-12.490	16.739	20
15	-15.722	16.517	20
16	-6.667	24.871	20
17	-30.379	7.260	20
18	-10.233	16.502	20
19	49.059	29.631	20
20	38.530	40.910	20
21	4.594	23.644	20
22	-31.617	7.766	20
23	-17.867	12.314	20
24	-28.007	14.497	19
25	-19.138	12.262	20
26	2.098	24.167	20
27	-13.876	15.861	20
28	-6.157	19.694	20
29	1.149	25.127	20
30	-19.656	16.029	20
31	-27.658	11.298	20
32	47.402	42.416	20
33	-18.714	11.884	20
34	49.248	37.978	20
35	-22.058	10.633	20

SAMPLE ENVIRONMENTAL DATA - ENVIRONMENT DATA REPORT

EXHIBIT 16

Date: 11/09/88

LOCATION

SECTION ID NO:	39
DISTRICT NO:	1
COUNTY NO/NAME:	117/HUNT
CONTROL-SECTION:	9-13
HIGHWAY:	IH 30
MILE POSTS:	106+00 TO 109+00
LANE:	R
PREVIOUS SID:	-
NEXT SID:	-
FUNCTIONAL CLASS:	0
TYPE OF PAVEMENT:	PCC
	2.5 =< HMAC < 5.5 (NO SEALS)
	WIDENING

ENVIRONMENT - 20 YEAR SUMMARY (1955-1974)

	JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG	SEP	OCT	NOV	DEC	ANN
THORNTHWAITE INDEX:	-	-	-	-	-	-	-	-	-	-	-	-	41.6
MEAN TEMPERATURE:	41	45	53	63	71	78	83	82	76	66	53	45	63.1
PRECIPITATION:	2.2	2.7	3.3	5.1	5.2	3.5	3.0	2.2	5.3	4.1	3.0	3.0	42.6
WET F-T CYCLES:	2	1	1	0	0	0	0	0	0	0	0	1	6
TOTAL F-T CYCLES:	13	9	4	0	0	0	0	0	0	0	3	11	40.2
DIST TEMP CONSTANT:	-	-	-	-	-	-	-	-	-	-	-	-	21

TRAFFIC

1986 ADT:	7800
1986 PERCENT TRUCKS:	28.4
1954-1986 VEHICLES:	47966822
1954-1986 18K AXLES:	14603862

SERVICEABILITY INDEX

YR	MEAN	STD DEV	N	CV	LOW	HIGH
80	2.66	.320	10	12.0	2.10	3.20
77	3.30	.356	10	10.8	2.70	3.80
76	3.45	.584	10	16.9	2.10	4.00
74	3.44	.230	9	6.7	2.90	3.60

PAVEMENT CONDITION SURVEY

PVMT RATING	80	77	76	74
PRS	63	69	76	76
RUTT	2MO	2SL	2SL	2SL
BLOCK CR				
ALLG CR				
LONG CR	1SE	1MO	1SL	1SL
TRANS CR	2SE	1MO	2MO	2MO
CRACKS	NS	NS	NS	NS
PATCHING				1F
FAIL/MI	0	0	0	0

SKID NUMBER

DATE	AVG	LOW	HIGH
3/75	25	22	29
7/74	33	30	36

DEFLECTION (MEAN VARIABLES)

DATE	D	W1	W2	W3	W4	W5	W6	W7
8/9/76	D	0.56	0.53	0.50	0.44	0.39		
0/0/ 0	F	0.00	0.00	0.00	0.00	0.00	0.00	0.00

STRUCTURAL SECTION

LAYER	STRUCTURE	DESCRIPTION	DATE	MATERIAL TYPE	AGG. RATE	ADMIXTURE TYPE	ADMIXTURE PCNT	APPL RATE	THICK CENT	TTC	LL	PI
5	1	S OS	8/67	HOT MIX - HOT LAID		AC	4.90		1.50			
4	1	S OVLY	8/67	HOT MIX - HOT LAID		AC	4.70		2.50			
3	1	S OS	9/52	PORTLAND CEMENT CONC					10.00			
2	1	SB BSLY	9/52	FLEXIBLE					6.00			
1	1	SG SBGR	9/52	CLAY						5.6	61.3	39.1

SAMPLE SUMMARY REPORT
EXHIBIT 17

Date: 11/09/88

LOCATION

SECTION ID NO:	39
DISTRICT NO:	1
COUNTY NO/NAME:	117/HUNT
CONTROL-SECTION:	9-13
HIGHWAY:	IH 30
MILE POINTS:	27.800-29.800
LANE:	R
MILE POST:	106+00 TO 109+00
PREVIOUS SID:	-
NEXT SID:	-
FUNCTIONAL CLASS:	0
TYPE OF PAVEMENT:	PCC
	2.5 =< HMAC < 5.5
	(NO SEALS)
	WIDENING
INACTIVE SID:	NO
NUMBER OF LANES:	2

GEOMETRIC AND SHOULDER INFORMATION

STRUCTURE NUMBER	TYPE OF PAVEMENT	LANE WIDTH	OUTSIDE SHOULDER WIDTH	SHOULDER SURFACE TYPE	SHOULDER BASE TYPE	SHOULDER SURFACE THICKNESS	SHOULDER BASE THICKNESS	WIDENING FLAG
1	34	12.0	17.0	2	21	0.00	8.00	1

STRUCTURAL SECTION

STRUCTURE	LAYER	DESCRIPTION	THICK CENT	MATERIAL TYPE	DATE	WIDENING DATE	AGG. RATE	ADMIXTURE TYPE	ADMIXTURE PCNT	APPL % PASSING RATE	200 SIEVE	TTC	LL	PI	PERM. INDEX
1	5	S OS	1.50	HOT MIX - HOT LAID	8/67				4.90						
1	4	S OVLY	2.50	HOT MIX - HOT LAID	8/67				4.70						
1	3	S OS	10.00	PORTLAND CEMENT CONC	9/52	9/52									
1	2	SB BSLY	6.00	FLEXIBLE	9/52										
1	1	SG SBGR		CLAY	9/52						85.7	5.6	61.3	39.1	0.60

SAMPLE INVENTORY UPDATE REPORT
EXHIBIT 18

Date: 11/09/88

LOCATION

SECTION ID NO:	39
DISTRICT NO:	1
COUNTY NO/NAME:	117/HUNT
CONTROL-SECTION:	9-13
HIGHWAY:	IH 30
MILE POINTS:	27.800-29.800
LANE:	R
MILE POST:	106+00 TO 109+00
PREVIOUS SID:	-
NEXT SID:	-
FUNCTIONAL CLASS:	0
TYPE OF PAVEMENT:	PCC
	2.5 =< HMAC < 5.5
	(NO SEALS)
	WIDENING
INACTIVE SID:	NO
NUMBER OF LANES:	2

GEOMETRIC AND SHOULDER INFORMATION

STRUCTURE NUMBER	TYPE OF PAVEMENT	LANE WIDTH	OUTSIDE SHOULDER WIDTH	SHOULDER SURFACE TYPE	SHOULDER BASE TYPE	SHOULDER SURFACE THICKNESS	SHOULDER BASE THICKNESS	WIDENING FLAG

LAYER THICKNESS ACROSS THE ROAD — CURRENT SURFACE WIDTH 12.0

STRUCTURE NUMBER	LAYER NUMBER	THICKNESS FROM CENTER 3rd POS	THICKNESS FROM CENTER 2nd POS	THICKNESS FROM CENTER 1st POS	THICKNESS AT CENTER	DISTANCE FROM CENTER 3rd POS	DISTANCE FROM CENTER 2nd POS	DISTANCE FROM CENTER 1st POS

STRUCTURE NUMBER	LAYER NUMBER	DESCRIPTION	THICK CENT	MATERIAL TYPE	DATE	WIDENING DATE	AGG. RATE	ADMIXTURE TYPE	ADMIXTURE PCNT	APPL RATE	% PASSING 200 SIEVE	TTC	LL	PI	PERM. INDEX

SAMPLE INVENTORY UPDATE REPORT
EXHIBIT 18 (Continued)

Texas Flexible Pavement Database
County Name Table

County Number	County Name
1	ANDERSON
2	ANDREWS
3	ANGELINA
4	ARANSAS
5	ARCHER
6	ARMSTRONG
7	ATASCOSA
8	AUSTIN
9	BAILEY
10	BANDERA
11	BASTROP
12	BAYLOR
13	BEE
14	BELL
15	BEXAR
16	BLANCO
17	BORDEN
18	BOSQUE
19	BOWIE
20	BRAZORIA
21	BRAZOS
22	BREWSTER
23	BRISCOE
24	BROOKS
25	BROWN
26	BURLESON
27	BURNET
28	CALDWELL
29	CALHOUN
30	CALLAHAN
31	CAMERON
32	CAMP
33	CARSON
34	CASS
35	CASTRO

SAMPLE COUNTY NAME TABLE REPORT

EXHIBIT 19

Page No. 1
07/06/88

Texas Flexible Pavement Database
Material Type Classification Table

Material Classif. Code	Material Code Description	Short Code	Layer Descr
1	HOT MIX - HOT LAID	HMAC	S
2	HOT MIX-COLD LAID	HMCL	S
4	COLD MIX ROCK ASPH	CMRA	S
5	ONE COURSE SURF TRT	ST	S
6	TWO COURSE SURF TRT	DST	S
7	THREE CORSE SURF TRT	TST	S
9	RUBBER ASPH CONCRETE		S
10	OPEN GRADE FRIC COUR	FC	S
11	SEAL COAT - REGULAR	SC	S
16	BLACK BASE	ASB	S
17	PORTLAND CEMENT CONC	PCC	S
18	BLANK		
21	FLEXIBLE BASE	FB	B
22	LIME STABILIZED	LSB	B
23	CEMENT STABILIZED	CSB	B
24	ASPHALT STAB BASE	ASB	B
25	ASPHLT BASE ROAD MIX	ARM	B
27	FABRIC		B
31	FLEXIBLE	FLEX	SB
32	LIME STABIL SUBGRADE	LSS	SB
33	CEMENT STABIL SUBG	CSS	SB
41	GRAVEL		SG
42	SAND		SG
43	SILT		SG
44	CLAY		SG
45	PEAT		SG

SAMPLE MATERIAL TYPE CLASSIFICATION TABLE REPORT

EXHIBIT 20

Texas Flexible Pavement Database
Type of Pavement Table

Pavement Code	Type of Base	Surface Thickness	Comments
1	GRANULAR BASE	SURFACE TREATED	
2	GRANULAR BASE	HMAC < 2.5"	(NO SEALS)
3	GRANULAR BASE	HMAC < 2.5"	(WITH SEALS)
4	GRANULAR BASE	2.5 =< HMAC < 5.5	(NO SEALS)
5	GRANULAR BASE	2.5 =< HMAC < 5.5	(WITH SEALS)
6	GRANULAR BASE	HMAC >= 5.5	(NO SEALS)
7	GRANULAR BASE	HMAC >= 5.5	(WITH SEALS)
11	STABILIZED (CEMENT/LIME)	SURFACE TREATED	
12	STABILIZED (CEMENT/LIME)	HMAC < 2.5"	(NO SEALS)
13	STABILIZED (CEMENT/LIME)	HMAC < 2.5"	(WITH SEALS)
14	STABILIZED (CEMENT/LIME)	2.5 =< HMAC < 5.5	(NO SEALS)
15	STABILIZED (CEMENT/LIME)	2.5 =< HMAC < 5.5	(WITH SEALS)
16	STABILIZED (CEMENT/LIME)	HMAC >= 5.5	(NO SEALS)
17	STABILIZED (CEMENT/LIME)	HMAC >= 5.5	(WITH SEALS)
21	ASPHALT STABILIZED BASE	SURFACE TREATED	
22	ASPHALT STABILIZED BASE	HMAC < 2.5"	(NO SEALS)
23	ASPHALT STABILIZED BASE	HMAC < 2.5"	(WITH SEALS)
24	ASPHALT STABILIZED BASE	2.5 =< HMAC < 5.5	(NO SEALS)
25	ASPHALT STABILIZED BASE	2.5 =< HMAC < 5.5	(WITH SEALS)
26	ASPHALT STABILIZED BASE	HMAC >= 5.5	(NO SEALS)
27	ASPHALT STABILIZED BASE	HMAC >= 5.5	(WITH SEALS)
31	PCC	SURFACE TREATED	
32	PCC	HMAC < 2.5"	(NO SEALS)
33	PCC	HMAC < 2.5"	(WITH SEALS)
34	PCC	2.5 =< HMAC < 5.5	(NO SEALS)
35	PCC	2.5 =< HMAC < 5.5	(WITH SEALS)
36	PCC	HMAC >= 5.5	(NO SEALS)
37	PCC	HMAC >= 5.5	(WITH SEALS)

SAMPLE TYPE OF PAVEMENT TABLE REPORT

EXHIBIT 21

Texas Flexible Pavement Database
District Temperature Constant Table

District	Temperature Constant
1	21
2	22
3	22
4	9
5	16
6	23
7	26
8	26
9	28
10	24
11	28
12	33
13	33
14	31
15	31
16	36
17	30
18	26
19	25
20	32
21	38
22	31
23	25
24	24
25	19

SAMPLE DISTRICT TEMPERATURE CONSTANT TABLE REPORT

EXHIBIT 22

Texas Flexible Pavement Database
Layer Description Table

Layer Code	Short Code	Layer Description
1	OVLY	Overlay
2	SC	Seal Coat
3	OS	Original Surface
4	HMAC	HMAC Layer
5	BSLY	Base Layer
6	SBLY	Subbase Layer
7	SBGR	Subgrade
8	INTL	Interlayer
9	PRFC	Porous Friction Course
10	ST	Surface Treatment
11	EMBK	Embankment (Fill)
12	RCSF	Recycle Surface
13	PMSF	Partially Milled Surface
14	FABR	Fabric

SAMPLE LAYER DESCRIPTION TABLE REPORT

EXHIBIT 23

Texas Flexible Pavement Database Conversion
Widening Table

Widening Code	Description	Comments
0	No Widening	The center thickness can be used
1	Widening Present	The center thickness can be used for Deflection data - materials have been added to the shoulder
	Special Widening	The center thickness cannot be- the material type changed in the middle of the lane.

SAMPLE WIDENING TABLE REPORT

EXHIBIT 24

Page No. 1
11/02/88

Texas Flexible Pavement Database
Functional Classification Table

Code	Code Description
1	Interstate
2	Other Urban Freeway and Expressway
3	Rural or Urban Principal Arterials
4	Minor Arterial Road or Street
5	Rural Major or Urban Collector Street
6	Rural Minor Collectors
7	Local Road or Street

SAMPLE FUNCTIONAL CLASSIFICATION TABLE REPORT

EXHIBIT 25

```
*
* SUBSYSTEM:        REPORTS MAIN MENU
* PROGRAM NAME:     REPORTS.PRG              05/25/88
* UPDATED ON:       09/04/88
* PROJECT 2456 -    TEXAS FLEXIBLE PAVEMENT DATABASE CONVERSION
* TAMU/TTI
* AUTHOR:           TREVOR X. PEREIRA
* PURPOSE:          TO PRINT REPORTS FOR THE FOLLOWING:
*                         - Monitoring data
*                         - Environment data
*                         - Inventory data
*                         - Traffic data
*                         - Inventory Update forms
*                         - Summary report
*                         - Tables
*

ON ERROR DO \PAVEDB\REPORTS\ERR_PRNT
SET PRINT TO &MPORT1
SET PRINT TO &MPORT2
ON ERROR DO \PAVEDB\ERROR

MREPEAT2 = .T.
DO WHILE MREPEAT2
 * set parameters and initialize variables
 STORE " " TO REPTPICK

 * set up the report menu screen and do the loop until REPTPICK is 1-7
 DO WHILE .NOT. (REPTPICK $ '1234567')
  @  0,  0  CLEAR
  @  4, 22  SAY "TEXAS FLEXIBLE PAVEMENT DATABASE      2.0"
  @  5, 33  SAY "Reports"
  @  8, 16  SAY "1 - Summary"
  @  9, 16  SAY "2 - Inventory Update Forms"
  @ 10, 16  SAY "3 - Inventory Data"
  @ 11, 16  SAY "4 - Monitoring Data"
  @ 12, 16  SAY "5 - Traffic Data"
  @ 13, 16  SAY "6 - Environmental Data"
  @ 14, 16  SAY "7 - Tables"
  @ 17, 36  SAY "OPTION ====>"
  @ 17, 50  GET REPTPICK
  @  3,  9  TO 18, 65    DOUBLE
  READ

  IF READKEY() = 12
     RETURN TO MASTER
  ENDIF
 ENDDO

 * according to the response received from the report menu, the appropriate
 * commands are executed
```

```
 IF MREPEAT2
  DO CASE
  * print summary report
  CASE REPTPICK = "1"
     DO \PAVEDB\EDITUPDT\TABLFILE
     close all
     IF .NOT. FILE('\PAVEDB\FILES\STRUCSEC.VUE')
          CLEAR
          ? "STRUCTURAL SECTION VIEW FILE (STRUCSEC.VUE) not found. Please
check . . ."
          WAIT
          RETURN TO MASTER
     ENDIF
     SET SAFETY OFF
     CLEAR
     REPTCOLL = " "
     DO WHILE .NOT. (REPTCOLL $ '123')
       @  4, 22  SAY "TEXAS FLEXIBLE PAVEMENT DATABASE       2.1"
       @  5, 34  SAY "REPORTS"
       @  6, 30  SAY "Summary Report"
       @  9, 16  SAY "1 - By SID Number"
       @ 10, 16  SAY "2 - By District"
       @ 11, 16  SAY "3 - ALL SID Numbers"
       @ 15, 41  SAY "OPTION ===>"
       @ 15, 55  GET REPTCOLL
       @  3,  9  TO 17, 65    DOUBLE
       READ
       IF READKEY() = 12
         EXIT
       ENDIF
     ENDDO
     DO CASE
       CASE REPTCOLL = "3"
         * run program to print report for all the SID numbers
         DO \PAVEDB\REPORTS\SUMMALL
         CLEA TYPE
       CASE REPTCOLL = "2"
         * run program to print report for a certain district
         DO \PAVEDB\REPORTS\SUMMDIST
         CLEA TYPE
       CASE REPTCOLL = "1"
         * run program to print report for one SID number
         DO \PAVEDB\REPORTS\SUMMSID
         CLEA TYPE
     ENDCASE
     CLEAR
     SET SAFETY ON

  * print inventory update forms for a single SID number or for a complete
  * district if choice 2 is chosen
  CASE REPTPICK = "2"
     DO \PAVEDB\EDITUPDT\TABLFILE
     close all
     IF .NOT. FILE('\PAVEDB\FILES\STRCSEC2.VUE')
```

```
          CLEAR
          ? "STRUCTURAL SECTION VIEW FILE (STRCSEC2.VUE) not found. Please
check . . ."
          WAIT
          RETURN TO MASTER
      ENDIF
      SET SAFETY OFF
      CLEAR
      REPTCOLL = " "
      DO WHILE .NOT. (REPTCOLL $ '123')
        @  4, 22  SAY "TEXAS FLEXIBLE PAVEMENT DATABASE       2.2"
        @  5, 34  SAY "REPORTS"
        @  6, 27  SAY "Inventory Update Forms"
        @ 10, 16  SAY "1 - By District"
        @ 11, 16  SAY "2 - By SID Number"
        @ 12, 16  SAY "3 - ALL SID Numbers"
        @ 15, 41  SAY "OPTION ====>"
        @ 15, 55  GET REPTCOLL
        @  3,  9  TO 17, 65    DOUBLE
        READ
        IF READKEY() = 12
          EXIT
        ENDIF
      ENDDO
      DO CASE
        CASE REPTCOLL = "1"
          DO \PAVEDB\REPORTS\SUM2DIST
          CLEA TYPE
        CASE REPTCOLL = "2"
          DO \PAVEDB\REPORTS\SUMM2SID
          CLEA TYPE
        CASE REPTCOLL = "3"
          DO \PAVEDB\REPORTS\SUM2FILE
          CLEA TYPE
      ENDCASE
      SET SAFETY ON

  * If choice 3 is chosen, print one of the 8 reports for the Inventory data
  CASE REPTPICK = "3"
    MREPEAT = .T.
    DO WHILE MREPEAT
      CLEAR
      SET PRINT TO &MPORT2
      REPTCOLL = " "
      DO WHILE .NOT. (REPTCOLL $ '12345678')
        @  3, 21  SAY "TEXAS FLEXIBLE PAVEMENT DATABASE       2.3"
        @  4, 34  SAY "REPORTS"
        @  5, 30  SAY "Inventory Data"
        @  8, 16  SAY "1 - Location"
        @  9, 16  SAY "2 - Location Section"
        @ 10, 16  SAY "3 - Layer ID"
        @ 11, 16  SAY "4 - Geometric & Shoulder"
        @ 12, 16  SAY "5 - Surface"
        @ 13, 16  SAY "6 - Subgrade"
```

```
  @ 14, 16  SAY "7 - Layer Thickness Across the Road"
  @ 15, 16  SAY "8 - Double Surface Treatment"
  @ 17, 37  SAY "OPTION ======>"
  @ 17, 51  GET REPTCOLL
  @  2,  9  TO 19, 65    DOUBLE
  READ
  IF READKEY() = 12
     MREPEAT = .F.
     EXIT
  ENDIF
ENDDO
IF MREPEAT
 SET ESCAPE ON
 DO CASE
  CASE REPTCOLL = "1"
     CLEAR
     USE \PAVEDB\FILES\LOCATION INDEX \PAVEDB\INDEXES\LOCSID
     REPORT FORM \PAVEDB\REPORTS\SEC_LOCN TO PRINT
  CASE REPTCOLL = "2"
     CLEAR
     SET PRINT TO &MPORT1
     @ 10, 5 SAY "Please set the printer to Condensed print"
     WAIT
     CLEAR
     USE \PAVEDB\FILES\LOCATION INDEX \PAVEDB\INDEXES\LOCSID
     REPORT FORM \PAVEDB\REPORTS\LOCATION TO PRINT
  CASE REPTCOLL = "3"
     CLEAR
     USE \PAVEDB\FILES\LAYER INDEX \PAVEDB\INDEXES\LAYNDX
     REPORT FORM \PAVEDB\REPORTS\LAYER TO PRINT
  CASE REPTCOLL = "4"
     CLEAR
     USE \PAVEDB\FILES\GEOSHO INDEX \PAVEDB\INDEXES\GEONDX
     REPORT FORM \PAVEDB\REPORTS\GEOSHO TO PRINT
  CASE REPTCOLL = "5"
     CLEAR
     USE \PAVEDB\FILES\SURFACE INDEX \PAVEDB\INDEXES\SURFNDX
     REPORT FORM \PAVEDB\REPORTS\SURFACE TO PRINT
  CASE REPTCOLL = "6"
     CLEAR
     USE \PAVEDB\FILES\SUBGRADE INDEX \PAVEDB\INDEXES\SUBGNDX
     REPORT FORM \PAVEDB\REPORTS\SUBGRADE TO PRINT
  CASE REPTCOLL = "7"
     CLEAR
     USE \PAVEDB\FILES\LAYTHICK INDEX \PAVEDB\INDEXES\LAYINDX
     REPORT FORM \PAVEDB\REPORTS\LAYTHICK TO PRINT
  CASE REPTCOLL = "8"
     CLEAR
     USE \PAVEDB\FILES\LAYER INDEX \PAVEDB\INDEXES\LAYNDX
     REPORT FORM \PAVEDB\REPORTS\DSTREPT FOR LAYMATCL = 6 TO PRINT
 ENDCASE
 SET ESCAPE OFF
 CLEA TYPE
 CLEAR
```

```
    ENDIF
   ENDDO

  * if choice 4 is chosen, print one of the 5 reports for the monitoring data
  CASE REPTPICK = "4"
    MREPEAT = .T.
    DO WHILE MREPEAT
     CLEAR
     SET PRINT TO &MPORT2
     REPTCOLL = " "
     DO WHILE .NOT. (REPTCOLL $ '12345')
       @  4, 22  SAY "TEXAS FLEXIBLE PAVEMENT DATABASE       2.4"
       @  5, 34  SAY "REPORTS"
       @  6, 30  SAY "Monitoring Data"
       @  9, 16  SAY "1 - Visual"
       @ 10, 16  SAY "2 - Serviceability Index"
       @ 11, 16  SAY "3 - Falling Weight"
       @ 12, 16  SAY "4 - Dynaflect"
       @ 13, 16  SAY "5 - Skid"
       @ 16, 38  SAY "OPTION ======>"
       @ 16, 52  GET REPTCOLL
       @  3,  9  TO 18, 65    DOUBLE
       READ
       IF READKEY() = 12
         MREPEAT = .F.
         EXIT
       ENDIF
     ENDDO
     IF MREPEAT
      SET ESCAPE ON
      DO CASE
       CASE REPTCOLL = "1"
          CLEAR
          SET PRINT TO &MPORT1
          @ 10, 5 SAY "Please set the printer to Condensed print to print
this report"
          WAIT
          CLEAR
          USE \PAVEDB\FILES\VISUAL INDEX \PAVEDB\INDEXES\VISUAL
          REPORT FORM \PAVEDB\REPORTS\VISUAL TO FILE VISUAL1.RPT
       CASE REPTCOLL = "2"
          CLEAR
          USE \PAVEDB\FILES\SI INDEX \PAVEDB\INDEXES\SI
          REPORT FORM \PAVEDB\REPORTS\SI TO PRINT
       CASE REPTCOLL = "3"
          CLEAR
          SET PRINT TO &MPORT1
          @ 10, 5 SAY "Please set the printer to Condensed print to print
this report"
          @ 11, 5 SAY "    Note that this report is in 2 parts and that the
first"
          @ 12, 5 SAY "    part will be printed completely before the next
part is printed."
          WAIT
```

```
        CLEAR
        USE \PAVEDB\FILES\FALLWGHT INDEX \PAVEDB\INDEXES\FALLWGHT
        REPORT FORM \PAVEDB\REPORTS\FALLWGHT TO PRINT
        GOTO TOP
        REPORT FORM \PAVEDB\REPORTS\FALLWGH2 TO PRINT
      CASE REPTCOLL = "4"
        CLEAR
        USE \PAVEDB\FILES\DYNAFLLD INDEX \PAVEDB\INDEXES\DYNAFLLD
        REPORT FORM \PAVEDB\REPORTS\DYNAFLLD TO PRINT
      CASE REPTCOLL = "5"
        CLEAR
        USE \PAVEDB\FILES\SKID INDEX \PAVEDB\INDEXES\SKID
        REPORT FORM \PAVEDB\REPORTS\SKID TO PRINT
     ENDCASE
     SET ESCAPE OFF
     CLEA TYPE
     CLEAR
    ENDIF
   ENDDO

* print traffic data if 5 is chosen
CASE REPTPICK = "5"
    CLEAR
    MRESPONSE = " "
    @ 10,10 SAY "You are about to print the Traffic report."
    @ 11,11 SAY "Want to continue (Y/N)? " GET MRESPONSE
    READ
    IF MRESPONSE = "Y"
      SET ESCAPE ON
      SET PRINT TO &MPORT2
      USE \PAVEDB\FILES\TRAFFIC INDEX \PAVEDB\INDEXES\TRAFFIC
      REPORT FORM \PAVEDB\REPORTS\TRAFFIC TO PRINT
      SET ESCAPE OFF
    ENDIF

* print out weather or environment data if choice 6 is chosen
CASE REPTPICK = "6"
   MREPEAT = .T.
   DO WHILE MREPEAT
    CLEAR
    SET PRINT TO &MPORT2
    REPTCOLL = " "
    DO WHILE .NOT. (REPTCOLL $ '12')
      @  5, 22  SAY "TEXAS FLEXIBLE PAVEMENT DATABASE       2.6"
      @  6, 34  SAY "REPORTS"
      @  7, 29  SAY "Environmental Data"
      @ 10, 15  SAY "1 - Environment"
      @ 11, 15  SAY "2 - Weather"
      @ 13, 40  SAY "OPTION ===>"
      @ 13, 55  GET REPTCOLL
      @  4,  9  TO 15, 65    DOUBLE
      READ
      IF READKEY() = 12
        MREPEAT = .F.
```

```
        EXIT
       ENDIF
      ENDDO
      IF MREPEAT
       SET ESCAPE ON
       DO CASE
        CASE REPTCOLL = "1"
           CLEAR
           USE \PAVEDB\FILES\ENV INDEX \PAVEDB\INDEXES\ENV
           REPORT FORM \PAVEDB\REPORTS\ENV TO PRINT
        CASE REPTCOLL = "2"
           CLEAR
           SET PRINT TO &MPORT1
           @ 10, 5 SAY "Please set the printer to Condensed print to print
this report"
           WAIT
           CLEAR
           USE \PAVEDB\FILES\WEATHER INDEX \PAVEDB\INDEXES\WEATHER
           REPORT FORM \PAVEDB\REPORTS\WEATHER TO PRINT
       ENDCASE
       SET ESCAPE OFF
       CLEAR
      ENDIF
     ENDDO

   * print out one of the tables if choice 7 is picked
   CASE REPTPICK = "7"
     MREPEAT = .T.
     DO WHILE MREPEAT
      CLEAR
      REPTCOLL = " "
      DO WHILE .NOT. (REPTCOLL $ '1234567')
        @  3, 21  SAY "TEXAS FLEXIBLE PAVEMENT DATABASE        2.7"
        @  4, 34  SAY "REPORTS"
        @  5, 34  SAY "Tables"
        @  8, 16  SAY "1 - County Name"
        @  9, 16  SAY "2 - Material Type"
        @ 10, 16  SAY "3 - Type of Pavement"
        @ 11, 16  SAY "4 - District Temperature Constant"
        @ 12, 16  SAY "5 - Widening Flag"
        @ 13, 16  SAY "6 - Layer Description"
        @ 14, 16  SAY "7 - Functional Classification"
        @ 17, 42  SAY "OPTION ======>"
        @ 17, 56  GET REPTCOLL
        @  2,  9  TO 19, 65    DOUBLE
       READ
       IF READKEY() = 12
         MREPEAT = .F.
         EXIT
       ENDIF
      ENDDO
      IF MREPEAT
       SET ESCAPE ON
       DO CASE
```

```
        CASE REPTCOLL = "1"
           CLEAR
           USE \PAVEDB\FILES\CNTYTBL INDEX \PAVEDB\INDEXES\CTYTBLNO
           REPORT FORM \PAVEDB\REPORTS\CNTYTBL TO PRINT
        CASE REPTCOLL = "2"
           CLEAR
           USE \PAVEDB\FILES\MATLTBL
           REPORT FORM \PAVEDB\REPORTS\MATLTBL TO PRINT
        CASE REPTCOLL = "3"
           CLEAR
           USE \PAVEDB\FILES\PAVETYPE
           REPORT FORM \PAVEDB\REPORTS\PAVETYPE TO PRINT
        CASE REPTCOLL = "4"
           CLEAR
           USE \PAVEDB\FILES\DISTTEMP
           REPORT FORM \PAVEDB\REPORTS\DISTTEMP TO PRINT
        CASE REPTCOLL = "5"
           CLEAR
           USE \PAVEDB\FILES\WIDENFLG
           REPORT FORM \PAVEDB\REPORTS\WIDENFLG TO PRINT
        CASE REPTCOLL = "6"
           CLEAR
           USE \PAVEDB\FILES\LAYERTBL
           REPORT FORM \PAVEDB\REPORTS\LAYERTBL TO PRINT
        CASE REPTCOLL = "7"
           CLEAR
           USE \PAVEDB\FILES\FUNCLTBL
           REPORT FORM \PAVEDB\REPORTS\FUNCLTBL TO PRINT
       ENDCASE
       SET ESCAPE OFF
       CLEAR
      ENDIF
     ENDDO
   ENDCASE
   MREPEAT2 = .F.
  ENDIF
  MREPEAT2 = .T.
ENDDO
RETURN
```

```
* SUBSYSTEM:      PRINT SUMMARY REPORT
* PROGRAM NAME:   SUMMSID.PRG        01/21/88
*                 REVISED ON:        05/26/88
* PROJECT 2456 - TEXAS FLEXIBLE PAVEMENT DATABASE CONVERSION
* TAMU/TTI
* AUTHOR:         TREVOR X. PEREIRA
* PURPOSE:        TO PRINT OUT A SUMMARY REPORT FOR INDIVIDUAL SID NUMBERS IN
*                 THE LOCATION FILE.  THE FOLLOWING PROGRAMS ARE CALLED:
*                              - GETPAVCN.PRG
*                              - GETENVIR.PRG
*                              - GETLOCAT.PRG
*                              - PRNTREPT.PRG
*                              - GETSKID.PRG
*
* THE FOLLOWING FILES ARE USED IN THIS PROGRAM:
*     CNTYENV.DBF          CNTYENV.NDX
*     CNTYTBL.DBF
*     CNTYWEAT.DBF         WEATMNTH.NDX
*                          WEATCNTY.NDX
*                          CTYTBLNO.NDX
*     DISTTEMP.DBF
*     DYNAFLLD.DBF         DYNAFLLD.NDX
*     FALLWGHT.DBF         FALLWGHT.NDX
*     GEOSHO.DBF           GEOSHO.NDX
*     LAYER.DBF            LAYNDX.NDX
*     LAYERTBL.DBF
*     LAYTHICK.DBF         LAYTNDX.NDX
*     LOCATION.DBF         LOCSID.NDX
*     MATLTBL.DBF
*     PAVETYPE.DBF
*     SI.DBF               SI.NDX
*     SIMPSKID.DBF         SIMPSKID.NDX
*     SUBGRADE.DBF         SUBGRADE.NDX
*     SURFACE.DBF          SURFNDX.NDX
*     TRAFFIC.DBF          TRAFFIC.NDX
*     VISUAL.DBF           VISUAL.NDX
*

* the program first gets all the data and stores it in separate files
* using memory variables.  When the report is printed out, the appropriate
* files with the variables are opened and deleted when done with them.

* set all parameters
SET TALK OFF
SET ECHO OFF
CLOSE DATABASES
CLEAR
SET PRINTER TO &MPORT2
PUBLIC MSID_NO, VALID, MCOUNTY, MDISTRICT
```

```
STORE " " TO MQUIT
STORE .F. TO VALID
STORE 0 TO MSID_NO

* get the sid number
DO WHILE .NOT. VALID
   CLEAR
   @ 10, 5 SAY " "
   ACCEPT "Please enter Section Identification (SID) number: " TO SID_NO
   IF READKEY() = 12
      CLEAR
      @ 19, 5 SAY "ARE YOU SURE YOU WANT TO RETURN TO PREVIOUS MENU?  (Y/N) "
GET MQUIT
      READ
      IF MQUIT = "Y" .OR. MQUIT = "y"
         EXIT
      ENDIF
   ENDIF
   MSID_NO = VAL(SID_NO)

   * calculates the correct Sid Number
   STORE 0 TO VAR1, VAR2, VAR3, COMPARE
   VAR1 = INT(MSID_NO/1000)
   VAR2 = MOD(INT(MSID_NO/100),10)
   VAR2 = VAR2 * 2
   VAR3 = MOD(INT(MSID_NO/10),10)
   VAR3 = VAR3 * 3
   VAR4 = MOD(VAR1,10)+VAR2+VAR3
   COMPARE = MOD(VAR4,10)

   * compares calculated Sid number with Sid number entered
   IF COMPARE = MOD(MSID_NO,10)
       STORE .T. TO VALID
   ELSE
       @ 19, 10 SAY "Invalid SID Number"
       WAIT
       @ 18, 10 CLEAR
   ENDIF
   * verifies that Sid number has been entered
   IF MSID_NO = 0
       STORE .F. TO VALID
       @ 19, 10 SAY "Invalid SID Number"
       WAIT
       @ 18, 10 CLEAR
   ENDIF
ENDDO

* get all the data and print the report
IF VALID
   CLEAR
   DO \PAVEDB\REPORTS\GETLOCAT
   DO \PAVEDB\REPORTS\GETPAVCN
   DO \PAVEDB\REPORTS\GETENVIR
   DO \PAVEDB\REPORTS\GETSKID
```

```
   DO \PAVEDB\REPORTS\PRNTREPT
ENDIF
RELEASE MSID_NO, VALID, MCOUNTY, MDISTRICT
RETURN
```

```
*
* SUBSYSTEM:      PRINT SUMMARY REPORT
* PROGRAM NAME:   SUMMDIST.PRG        02/04/88
*                 REVISED ON:         05/26/88
* PROJECT 2456 - TEXAS FLEXIBLE PAVEMENT DATABASE CONVERSION
* TAMU/TTI
* AUTHOR:         TREVOR X. PEREIRA
* PURPOSE:        TO PRINT OUT A SUMMARY REPORT FOR ALL SID NUMBERS IN A
*                 PARTICULAR DISTRICT.  THE FOLLOWING PROGRAMS ARE CALLED:
*                              - GETPAVCN.PRG
*                              - GETENVIR.PRG
*                              - GETLOCAT.PRG
*                              - PRNTREPT.PRG
*                              - GETSKID.PRG
*
* THE FOLLOWING FILES ARE USED IN THIS PROGRAM:
*    CNTYENV.DBF          CNTYENV.NDX
*    CNTYTBL.DBF
*    CNTYWEAT.DBF         WEATMNTH.NDX
*                         WEATCNTY.NDX
*                         CTYTBLNO.NDX
*    DISTTEMP.DBF
*    DYNAFLLD.DBF         DYNAFLLD.NDX
*    FALLWGHT.DBF         FALLWGHT.NDX
*    GEOSHO.DBF           GEOSHO.NDX
*    LAYER.DBF            LAYNDX.NDX
*    LAYERTBL.DBF
*    LAYTHICK.DBF         LAYTNDX.NDX
*    LOCATION.DBF         LOCSID.NDX
*    MATLTBL.DBF
*    PAVETYPE.DBF
*    SI.DBF               SI.NDX
*    SIMPSKID.DBF         SIMPSKID.NDX
*    SUBGRADE.DBF         SUBGRADE.NDX
*    SURFACE.DBF          SURFNDX.NDX
*    TRAFFIC.DBF          TRAFFIC.NDX
*    VISUAL.DBF           VISUAL.NDX
*    SIDSTORE.DBF - This file is used to store the SID numbers for a
*                   particular district.
*

* the program first gets all the SID numbers for the particular district and
* stores them in a temporary file.  Then data for each SID number is stored
* in memory variable files.  When the report is printed out, the appropriate
* files with the variables are opened and deleted when done with them.  This
* process continues for each SID number in the temporary file.

* set all parameters
CLOSE DATABASES
CLEAR
SET PRINTER TO &MPORT2
```

```
PUBLIC MSID_NO, VALID, MCOUNTY, MDISTRICT
STORE " " TO MQUIT, MDISTT
STORE .F. TO VALID
STORE 0 TO MSID_NO, MDISTT

* get the sid number
DO WHILE .NOT. VALID
   CLEAR
   CLEAR TYPEAHEAD
   @ 10, 5 SAY " "
   ACCEPT "Please enter District number: " TO MDISTT
   IF READKEY() = 12
      CLEAR
      @ 19, 5 SAY "ARE YOU SURE YOU WANT TO RETURN TO PREVIOUS MENU?  (Y/N) "
GET MQUIT
      READ
      IF MQUIT = "Y" .OR. MQUIT = "y"
         EXIT
      ENDIF
   ENDIF
   MDIST = VAL(MDISTT)
   @ 10, 0 CLEAR


   * Get all SID_NO's for the required district and store them to a temporary
   * dBASE file
   SELECT 1
      USE \PAVEDB\FILES\LOCATION index \PAVEDB\INDEXES\locsid
   SELECT 2
      IF .NOT. FILE('\PAVEDB\FILES\SIDSTORE.DBF')
          CLEAR
          ? "Temporary SID storage file (SIDSTORE.DBF) not found. Please
check . . ."
          WAIT
          RETURN TO MASTER
      ENDIF
      USE \PAVEDB\FILES\SIDSTORE
      DELE ALL
      PACK

   SELECT 1
   LOCATE FOR HWYDIST = MDIST
   IF .NOT. FOUND()
     @ 12, 5 SAY "District not found.  Please try again."
     WAIT
   ENDIF
   DO WHILE FOUND()
     MSID_NO = A->SID_NO
     SELE 2
     APPEND BLANK
     REPLACE B->SID_NO WITH MSID_NO
     SELE 1
     VALID = .T.
     CONTINUE
```

```
   ENDD
ENDDO
SELE 1
   USE
SELE 2
   APPEN BLANK
   REPLACE B->SID_NO WITH 0
   USE

* Print out the reports
USE \PAVEDB\FILES\SIDSTORE
MREC = 1
IF SID_NO = 0
  STORE .F. TO GOON
ELSE
  STORE .T. TO GOON
ENDIF
DO WHILE GOON
  MSID_NO = SID_NO
  DO \PAVEDB\REPORTS\GETLOCAT
  DO \PAVEDB\REPORTS\GETPAVCN
  DO \PAVEDB\REPORTS\GETENVIR
  DO \PAVEDB\REPORTS\GETSKID
  DO \PAVEDB\REPORTS\PRNTREPT
  MREC = MREC + 1
  USE \PAVEDB\FILES\SIDSTORE
  GOTO MREC
  IF SID_NO = 0
    GOON = .F.
  ENDIF
  CLEAR
ENDDO
SELE 3
  USE
SELE 2
  USE
SELE 1
  USE

CLOSE DATABASES
SET PRINTER TO LPT1
RELE MSID_NO, VALID, MCOUNTY, MDISTRICT
RETURN
```

```
*
* SUBSYSTEM:      PRINT SUMMARY REPORT
* PROGRAM NAME:   SUMMALL.PRG        01/21/88
*                 REVISED ON:        05/26/88
* PROJECT 2456 - TEXAS FLEXIBLE PAVEMENT DATABASE CONVERSION
* TAMU/TTI
* AUTHOR:         TREVOR X. PEREIRA
* PURPOSE:        TO PRINT OUT A SUMMARY REPORT FOR ALL THE SID NUMBERS
*                 IN THE LOCATION FILE.  THE FOLLOWING PROGRAMS
*                 ARE CALLED:  - GETPAVCN.PRG
*                              - GETENVIR.PRG
*                              - GETLOCAT.PRG
*                              - PRNTREPT.PRG
*                              - GETSKID.PRG
*
* THE FOLLOWING FILES ARE USED IN THIS PROGRAM:
*     CNTYENV.DBF          CNTYENV.NDX
*     CNTYTBL.DBF
*     CNTYWEAT.DBF         WEATMNTH.NDX
*                          WEATCNTY.NDX
*                          CTYTBLNO.NDX
*     DISTTEMP.DBF
*     DYNAFLLD.DBF         DYNAFLLD.NDX
*     FALLWGHT.DBF         FALLWGHT.NDX
*     GEOSHO.DBF           GEOSHO.NDX
*     LAYER.DBF            LAYNDX.NDX
*     LAYERTBL.DBF
*     LAYTHICK.DBF         LAYTNDX.NDX
*     LOCATION.DBF         LOCSID.NDX
*     MATLTBL.DBF
*     PAVETYPE.DBF
*     SI.DBF               SI.NDX
*     SIMPSKID.DBF         SIMPSKID.NDX
*     SUBGRADE.DBF         SUBGRADE.NDX
*     SURFACE.DBF          SURFNDX.NDX
*     TRAFFIC.DBF          TRAFFIC.NDX
*     VISUAL.DBF           VISUAL.NDX
*

* the program first gets all the data and stores it in separate files
* using memory variables.  When the report is printed out, the appropriate
* files with the variables are opened and closed when done with them.

* set all parameters
SET TALK OFF
SET ECHO OFF
CLOSE DATABASES
CLEAR
SET PRINTER TO &MPORT2
PUBLIC MSID_NO, MCOUNTY, MDISTRICT
STORE " " TO MQUIT
```

```
STORE 0 TO MSID_NO
STORE 1 TO MREC
USE \PAVEDB\FILES\LOCATION
MRECNUM = RECCOUNT() + 1

DO WHILE MREC < MRECNUM
   USE \PAVEDB\FILES\LOCATION
   GOTO MREC
   MSID_NO = SID_NO
   USE
   IF READKEY() = 12
      CLEAR
      @ 19, 5 SAY "ARE YOU SURE YOU WANT TO QUIT?  (Y/N) "
      @ 19, 43 GET MQUIT
      READ
      IF MQUIT = "Y" .OR. MQUIT = "y"
         EXIT
      ENDIF
   ENDIF
   CLEAR

   * get all the data and print the report
   DO \PAVEDB\REPORTS\GETLOCAT
   DO \PAVEDB\REPORTS\GETPAVCN
   DO \PAVEDB\REPORTS\GETENVIR
   DO \PAVEDB\REPORTS\GETSKID
   DO \PAVEDB\REPORTS\PRNTREPT
   MREC = MREC + 1
ENDDO

RELEASE MSID_NO, MCOUNTY, MDISTRICT
RETURN
```

```
* SUBSYSTEM:      PRINT SUMMARY REPORT
* PROGRAM NAME:  GETLOCAT.PRG     01/26/88
* PROJECT 2456 - TEXAS FLEXIBLE PAVEMENT DATABASE CONVERSION
* TAMU/TTI
* AUTHOR:         TREVOR X. PEREIRA
* PURPOSE:        TO GET DATA FROM THE FOLLOWING FILES TO PRINT THE SUMMARY
*                 REPORT:       1) LOCATION dBASE FILE
*                               2) SERVICEABILITY dBASE FILE
*                               3) TRAFFIC dBASE FILE
*                               4) LAYER IDENTIFICATION dBASE FILE
*                               5) SUBGRADE LAYER dBASE FILE
*                               6) SURFACE LAYER dBASE FILE
*                               7) GEOMETRIC & SHOULDER INFO LAYER dBASE FILE

*************************
* GET LOCATION PART OF THE REPORT AND STORE THE DATA IN A FILE
*************************

@ 5, 5 SAY "Getting data for the Location section of the report ...."

* assign databases to different work areas
SELECT 1
use \PAVEDB\FILES\location INDEX \PAVEDB\INDEXES\LOCSID

SELECT 2
USE \PAVEDB\FILES\CNTYTBL INDEX \PAVEDB\INDEXES\CTYTBLNO

* locate for the requested sid number
SELECT 1
seek msid_no
if .not. found()
   okay = .F.
   clear
   @ 19, 10 say "SID number not found. "
   WAIT
   @ 18, 10 CLEAR
   RETURN
else
   okay = .T.
endif
STORE 0 TO MCOUNTY
STORE 0 TO MDISTRICT

* if the sid number is found, get all the required data
if Okay
   mlhwydst = LTRIM(STR(hwydist,2))
   MDISTRICT = HWYDIST
   MCOUNTY = CNTYNUM
   mlcntynm = LTRIM(STR(cntynum,3))
   SELECT 2
```

```
   SEEK A->CNTYNUM
   IF FOUND()
     MLCNTYNAM = LTRIM(CNTYNAME)
   ENDIF
   SELECT 1
   mcontrl = LTRIM(STR(control,4))
   msectn  = LTRIM(STR(section,2))
   MLCONTSEC = MCONTRL + "-" + MSECTN
   mprefx  = hwyprefx
   mhwynum = LTRIM(STR(hwynum,4))
   msuffx  = hwysuffx
   MLHWY = MPREFX + " " + LTRIM(MHWYNUM) + " " + MSUFFX
   MBEGP = LTRIM(STR(BEGMPST,3))
   MBEGD =
RIGHT((STUFF(STR(BMPSTDIS,2),1,(2-LEN(LTRIM(STR(BMPSTDIS,2)))),"0")),2)
   MPOSTB = MBEGP+BDISSIGN+MBEGD
   MENDD =
RIGHT((STUFF(STR(EMPSTDIS,2),1,(2-LEN(LTRIM(STR(EMPSTDIS,2)))),"0")),2)
   MENDP = LTRIM(STR(ENDMPST,3))
   MPOSTE = MENDP+EDISSIGN+MENDD
   MLPOST = MPOSTB+" TO "+MPOSTE
   mllaneid = laneid
   IF PREVSID = 0
     MLPREVSD = "-"
   ELSE
     mlprevsd = LTRIM(STR(prevsid,4))
   ENDIF
   IF NEXTSID = 0
     MLNEXTSD = "-"
   ELSE
     mlnextsd = LTRIM(STR(nextsid,4))
   ENDIF
   MLFUNCLS = LTRIM(STR(FUNCLAS,2))
endif



* get geometric and shoulder information layer data
SELECT 1
use \PAVEDB\FILES\geosho index \PAVEDB\INDEXES\geondx

SELECT 2
USE \PAVEDB\FILES\PAVETYPE

SELECT 1
seek str(msid_no,4)
if found()
   DO CASE
     CASE WIDENFLG = 0
       MLWIDEN = "NO WIDENING"
     CASE WIDENFLG = 1
       MLWIDEN = "WIDENING"
     CASE WIDENFLG = 2
       MLWIDEN = "UNUSUAL WIDENING"
```

```
   ENDCASE
   MPAVEM = PAVETYP
   SELECT 2
   LOCATE FOR PAVECODE = MPAVEM
   IF FOUND()
     MLBASETYP = LTRIM(BASETYPE)
     MLBASETHK = LTRIM(BASETHK)
     MLBASESEAL = LTRIM(BASESEAL)
   ELSE
     MLBASETYP = " "
     MLBASETHK = " "
     MLBASESEAL = " "
   ENDIF
ELSE
   MLWIDEN = " "
   MLBASETYP = " "
   MLBASETHK = " "
   MLBASESEAL = " "
ENDIF

* save all location variables to a file
SAVE ALL LIKE ML* TO GETLOCAT
RELEASE ALL LIKE ML*
sele 1
use
sele 2
use

************************
* GET SERVICEABILITY FILE DATA AND STORE TO A FILE
************************

@ 6, 5  SAY "Getting data for Serviceability Index Section ...."
X = 1
use \PAVEDB\FILES\si index \PAVEDB\INDEXES\si
DO WHILE X < 7
   S = STR(X,1,0)
   msyear&S  = "  "
   msmean&S  = "    "
   msstdd&S  = "    "
   mscont&S  = "  "
   mscoef&S  = "    "
   mslowv&S  = "    "
   mshigh&S  = "    "
   X = X + 1
ENDDO

seek str(msid_no,4)

* if the sid number is found, get the required data and store it in variables
if found()
   * move the file pointer to the last record of the sid_no in order to get
   * the most current year first
   DO WHILE .NOT. EOF() .AND. SID_NO = MSID_NO
```

```
      SKIP
   ENDDO
   SKIP -1
   X = 1
   do while .NOT. EOF() .AND. SID_NO = MSID_NO
      IF X > 6
         EXIT
      ENDIF
      S = STR(X,1,0)
      msyear&S  = STR(actyear,2)
      msmean&S  = STR(simean,4,2)
      msstdd&S  = STR(sisd,4,3)
      mscont&S  = STR(sicount,2)
      mscoef&S  = STR(((sisd/simean)*100),4,1)
      mslowv&S  = STR(silowval,4,2)
      mshigh&S  = STR(sihival,4,2)
      X = X + 1
      skip -1
      IF BOF()
         SKIP 35
      ENDIF
   ENDDO
ENDIF

USE

* save all variables to file
save all like ms* to getservc
RELEASE ALL LIKE MS*



************************
* GET TRAFFIC FILE DATA AND STORE IT IN VARIABLES
************************

@ 7, 5 SAY  "Getting data for the Traffic Section ...."
USE \PAVEDB\FILES\TRAFFIC INDEX \PAVEDB\INDEXES\TRAFFIC
SEEK STR(MSID_NO,4)
* if the sid number is found, get the required data
IF FOUND()
   MTBYEAR = STR(YEAR,4)
   MVEHADD = AADT1WAY * 365.25
   MVEHICLE = MVEHADD
   M18KEAL = A18KEAL
   SKIP
   DO WHILE .NOT. EOF() .AND. TRAFFIC->SID_NO = MSID_NO
     MVEHADD = AADT1WAY * 365.25
     MVEHICLE = MVEHICLE + MVEHADD
     M18KEAL = M18KEAL + A18KEAL
     SKIP
   ENDDO
   MT18KEAL = STR(M18KEAL,9)
   MTVEHIC  = STR(MVEHICLE,10)
```

```
   SKIP -1
   MTAADT = STR(AADT1WAY,6)
   MTPTRK = STR(PCTTRK,4,1)
   MTEYEAR = STR(YEAR,4)
ELSE
   STORE "    " TO MTBYEAR, MTEYEAR, MTPTRK
   STORE "      " TO MTAADT
   STORE "         " TO MT18KEAL
   STORE "          " TO MTVEHIC
ENDIF
* save the variables to file
SAVE ALL LIKE MT* TO GETTRAFF
RELEASE ALL LIKE MT*
CLOSE ALL
use

*******************
* GET STRUCTURAL SECTION OF THE REPORT AND SAVE THE DATA IN VARIABLES
*******************

* variable mcount is to find out how many layers there are for the SID #.
* This will help determine the layer the subgrade goes on.  This is needed
* if layer # 1 is the top most layer and not the subgrade.


@ 8, 5  SAY  "Getting data for the Structural Section ...."
STORE " " TO MTPLAST, MTLIQLIM, MTTXTRL
MCOUNT = 1

set path to \pavedb\files,\pavedb\indexes
set view to \pavedb\files\strucsec
seek str(msid_no,4)

* if the sid number is found, get the required data
if found()
  do while .NOT. EOF() .AND. sid_no = msid_no
     skip
  enddo
  skip -1
  X = 1
  do while .NOT. EOF() .AND. sid_no = msid_no
      IF X < 10
         S = STR(X,1)
      ELSE
         S = STR(X,2)
      ENDIF
      mtstruc&S  = str(strucnum,2)
      mtlay&S    = str(laynum,2)
      mlaydes&S = laydesc
      MTDATE&S  = str(jobcmpmo,2)+"/"+ str(jobcmpyr,2)
      mlaymat&S = laymatcl
      IF CENTIHK = 0
         MTCENTHK&S = " "
      ELSE
```

```
         mtcenthk&S = str(centthk,5,2)
      ENDIF
      IF AGAPPLRT = 0
         MTAGGRAT&S = " "
      ELSE
         mtaggrat&S = str(agapplrt,3)
      ENDIF
      mtadmxtp&S = admxtyp
      IF ADMXPER = 0
         MTADMXPR&S = " "
      ELSE
         mtadmxpr&S = str(admxper,5,2)
      ENDIF
      IF ASAPPLRT = 0
         MTAPPLRT&S = " "
      ELSE
         mtapplrt&S = str(asapplrt,4,2)
      ENDIF
      SAVE ALL LIKE MT* TO GETSTC&S
      RELEASE ALL LIKE MT*
      X = X + 1
      MCOUNT = MCOUNT + 1
      skip -1
      IF BOF()
         SKIP 35
      ENDIF
   enddo
   X = MCOUNT
   MCOUNT = MCOUNT - 1
   * initialize all variables
   DO WHILE  X < 14
      IF X < 10
         S = STR(X,1)
      ELSE
         S = STR(X,2)
      ENDIF
      STORE " " TO MTSTRUC&S, MTLAY&S, MTJOBMO&S, MTJOBYR&S, MTCENTHK&S,
MTAGGRAT&S, MTADMXTP&S, MTADMXPR&S
      STORE " " TO MTAPPLRT&S, MTDATE&S, MLAYDES&S, MLAYMAT&S
      SAVE ALL LIKE MT* TO GETSTC&S
      RELEASE ALL LIKE MT*
      X = X + 1
   ENDDO

   * get the material type and the layer names from the tables
   SELECT 6
     USE \PAVEDB\FILES\MATLTBL
   SELECT 7
     USE \PAVEDB\FILES\LAYERTBL
   SELECT 6
   MZCOUNT = MCOUNT
   X = 1
   DO WHILE X < 14
      IF X < 10
```

```
        S = STR(X,1)
      ELSE
        S = STR(X,2)
      ENDIF
      SELECT 6
      TYPE1 = MLAYMAT&S
      TYPE2 = "TYPE1"
      TYPE3 = TYPE(TYPE2)
      IF TYPE3 = "N"
        LOCATE FOR MATCODE = MLAYMAT&S
        IF FOUND()
          MZMATTYP&S = F->MATDESC
          MZLAYDA&S  = F->LAYRDES
        ELSE
          MZMATTYP&S = " "
          MZLAYDA&S  = " "
        ENDIF
      ELSE
          MZMATTYP&S = " "
          MZLAYDA&S  = " "
      ENDIF
      SELECT 7
      TYPE1 = MLAYDES&S
      TYPE2 = "TYPE1"
      TYPE3 = TYPE(TYPE2)
      IF TYPE3 = "N"
        LOCATE FOR CODE = MLAYDES&S
        IF FOUND()
          MZLAYDB&S = G->CODE_DESC
        ELSE
          MZLAYDB&S = " "
        ENDIF
      ELSE
          MZLAYDB&S = " "
      ENDIF
      X = X + 1
    ENDDO
  ELSE
    X = 1
    DO WHILE  X < 14
      IF X < 10
        S = STR(X,1)
      ELSE
        S = STR(X,2)
      ENDIF
      STORE " " TO MISTRUC&S, MILAY&S, MICENTHK&S, MIAGGRAT&S, MIADMXTP&S,
  MIADMXPR&S
      STORE " " TO MIAPPLRT&S, MIDATE&S, MZLAYDA&S, MZLAYDB&S, MZMATTYP&S
      SAVE ALL LIKE MI* TO GETSTC&S
      RELEASE ALL LIKE MI*
      X = X + 1
    ENDDO
    STORE 0 TO MZCOUNT
  endif
```

```
MCOUNT = MCOUNT - 1
close DATABASES
set path to

* get the subgrade file data
use \PAVEDB\FILES\subgrade index \PAVEDB\INDEXES\subgndx
seek str(msid_no,4)
if found()
   MZplast  = str(plastix,4,1)
   MZliqlim = str(liqlim,4,1)
   MZtxtrl  = str(txtriaxl,3,1)
ELSE
   MZplast  = "    "
   MZliqlim = "    "
   MZtxtrl  = "   "
endif

close all

* save all the variables to file
save all like MZ* to getstruc
RELEASE ALL LIKE MZ*
return
```

PROGRAM LISTING

```
*
* SUBSYSTEM:      PRINT SUMMARY REPORT
* PROGRAM NAME:   GETENVIR.PRG     01/26/88
* PROJECT 2456 - TEXAS FLEXIBLE PAVEMENT DATABASE CONVERSION
* TAMU/TTI
* AUTHOR:         TREVOR X. PEREIRA
* PURPOSE:        TO GET DATA FOR THE ENVIRONMENT SECTION OF THE REPORT FROM
*                 THE FOLLOWING FILES: 1)  WEATHER dBASE FILE
*                                      2)  ENVIRONMENT dBASE FILE
*

@ 10, 5  SAY "Getting Environment Summary data ...."

* get data from the environment file
USE \PAVEDB\FILES\ENV INDEX \PAVEDB\INDEXES\ENV

SEEK MCOUNTY
IF FOUND()
     METHORN = STR(THORNMN,4,1)
ELSE
     METHORN = "    "
ENDIF

* get data from the weather file
USE \PAVEDB\FILES\WEATHER INDEX \PAVEDB\INDEXES\WEATHER
X = 1
* initialize variables
DO WHILE X < 13
  IF X < 10
     S = STR(X,1,0)
  ELSE
     S = STR(X,2,0)
  ENDIF
  MEMEAN&S = "   "
  MEPREC&S = "   "
  MEWFTC&S = "   "
  METFTC&S = "   "
  X = X + 1
ENDDO
MEMTAVER = "    "
MEPRECIP = "    "
MEWEITFTC = "   "
METOTFTC = "    "

* get file data
SEEK str(MCOUNTY,3)
IF FOUND()
   X = 1
   STORE 0 TO MMTAVER, MPRECIP, MWEITFTC, MTOTFTC
   DO WHILE .NOT. EOF() .AND. A->CNTYNUM = MCOUNTY
      IF X > 12
```

```
            EXIT
         ENDIF
         IF X < 10
           S = STR(X,1,0)
         ELSE
           S = STR(X,2,0)
         ENDIF
         MEMEAN&S = STR(ATMPMN,3,0)
         MEPREC&S = STR(PRECMN,3,1)
         MEWFTC&S = STR(WFTCMN,3,0)
         METFTC&S = STR(TFTCMN,3,0)
         MMTAVER = MMTAVER + ATMPMN
         MPRECIP = MPRECIP + PRECMN
         MWETFTC = MWETFTC + WFTCMN
         MTOTFTC = MTOTFTC + TFTCMN
         X = X + 1
         SKIP
      ENDDO
      MTEMP = MMTAVER/12
      MEMTAVER = STR(MTEMP,4,1)
      MEPRECIP = STR(MPRECIP,4,1)
      MEWETFTC = STR(MWETFTC,3,0)
      METOTFTC = STR(MTOTFTC,4,1)
ENDIF

USE \PAVEDB\FILES\DISTTEMP
LOCATE FOR DISTTEMP->DISTRICT = MDISTRICT
IF FOUND()
   METEMPCN = STR(TEMPCONS,2)
ELSE
   METEMPCN = "  "
ENDIF

* save all variables to file
SAVE ALL LIKE ME* TO GETENVIR
CLOSE ALL

RETURN
```

```
*
* SUBSYSTEM:      PRINT SUMMARY REPORT
* PROGRAM NAME:  GETPAVCN.PRG      01/26/88
* PROJECT 2456 - TEXAS FLEXIBLE PAVEMENT DATABASE CONVERSION
* TAMU/TTI
* AUTHOR:         TREVOR X. PEREIRA
* PURPOSE:        TO GET DATA FROM THE VISUAL dBASE FILE TO PRINT OUT THE
*                 SUMMARY REPORT.
*

X = 1
* initialize all variables
DO WHILE X < 7
   IF X < 10
      S = STR(X,1,0)
   ELSE
      S = STR(X,2,0)
   ENDIF
   mpactyr&S = "  "
   mpprs&S   = "   "
   MPRUTTN&S = " "
   MPRUTTA&S = "  "
   MPBLCKN&S = " "
   MPBLCKA&S = "  "
   MPALLGN&S = " "
   MPALLGA&S = "  "
   MPLONGN&S = " "
   MPLONGA&S = "  "
   MPTRANN&S = " "
   MPTRANA&S = "  "
   mpcrack&S  = " "
   MPPATCN&S = " "
   MPPATCA&S = " "
   mpfailml&S = " "
   X = X + 1
ENDDO

@ 9, 5  SAY "Getting Pavement Condition Survey data ...."
use \PAVEDB\FILES\visual index \PAVEDB\INDEXES\visual
seek str(msid_no,4)
if found()
  * go to the last record of the sid number in order to get the most current
  * sid number first
  DO WHILE .NOT. EOF() .AND. MSID_NO = SID_NO
     SKIP
  ENDDO
  SKIP -1

  X = 1
  * get the required data and store it in variables
  do while .NOT. EOF() .AND. msid_no = sid_no
     IF X < 10
        S = STR(X,1,0)
     ELSE
```

```
   S = STR(X,2,0)
ENDIF
mpactyr&S = str(actyear,2)
mpprs&S   = str(prs,3)
DO CASE
   CASE RUTTSV<>0
      MPRUTIN&S = STR(RUTTSV,1)
      MPRUTTA&S = "SE"
   CASE RUTTSL<>0
      MPRUTIN&S = STR(RUTTSL,1)
      MPRUTTA&S = "SL"
   CASE RUTTMD<>0
      MPRUTIN&S = STR(RUTTMD,1)
      MPRUTTA&S = "MO"
   OTHERWISE
      MPRUTIN&S = " "
      MPRUTTA&S = "  "
ENDCASE
DO CASE
   CASE BLKCRKMD<>0
      mpblckN&S = STR(blkcrkMD,1)
      MPBLCKA&S = "MO"
   CASE BLKCRKSL<>0
      mpblckN&S = STR(blkcrkSL,1)
      MPBLCKA&S = "SL"
   CASE BLKCRKSV<>0
      mpblckN&S = STR(blkcrksv,1)
      MPBLCKA&S = "SE"
   OTHERWISE
      MPBLCKN&S = " "
      MPBLCKA&S = "  "
ENDCASE
DO CASE
   CASE ALLGCRMD<>0
      mpALLGN&S = STR(ALLGCRMD,1)
      MPALLGA&S = "MO"
   CASE ALLGCRSL<>0
      mpALLGN&S = STR(ALLGCRSL,1)
      MPALLGA&S = "SL"
   CASE ALLGCRSV<>0
      mpALLGN&S = STR(ALLGCRsv,1)
      MPALLGA&S = "SE"
   OTHERWISE
      MPALLGN&S = " "
      MPALLGA&S = "  "
ENDCASE
DO CASE
   CASE LONGCRMD<>0
      mpLONGN&S = STR(LONGCRMD,1)
      MPLONGA&S = "MO"
   CASE LONGCRSL<>0
      mpLONGN&S = STR(LONGCRSL,1)
      MPLONGA&S = "SL"
   CASE LONGCRSV<>0
```

```
            mpLONGN&S = STR(LONGCRsv,1)
            MPLONGA&S = "SE"
         OTHERWISE
            MPLONGN&S = " "
            MPLONGA&S = "  "
      ENDCASE
      DO CASE
         CASE TRANCRMD<>0
            mpTRANN&S = STR(TRANCRMD,1)
            MPTRANA&S = "MO"
         CASE TRANCRSL<>0
            mpTRANN&S = STR(TRANCRSL,1)
            MPTRANA&S = "SL"
         CASE TRANCRSV<>0
            mpTRANN&S = STR(TRANCRSV,1)
            MPTRANA&S = "SE"
         OTHERWISE
            MPTRANN&S = " "
            MPTRANA&S = "  "
      ENDCASE

      DO CASE
         CASE SEALCRCD = 0
           mpcrack&S  = " "
         CASE SEALCRCD = 1
           mpcrack&S  = " S"
         CASE SEALCRCD = 2
           MPCRACK&S  = "PS"
         CASE SEALCRCD = 3
           MPCRACK&S  = "NS"
      ENDCASE

      DO CASE
         CASE PATCHGD<>0
            mpPATCN&S = STR(PATCHGD,1)
            MPPATCA&S = "G"
         CASE PATCHFR<>0
            mpPATCN&S = STR(PATCHFR,1)
            MPPATCA&S = "F"
         CASE PATCHPR<>0
            mpPATCN&S = STR(PATCHPR,1)
            MPPATCA&S = "P"
         OTHERWISE
            MPPATCN&S = " "
            MPPATCA&S = " "
      ENDCASE

      mpfailml&S = str(failmile,1)
      X = X + 1
      skip -1
      IF BOF()
         SKIP 35
      ENDIF
   enddo
```

```
endif

use
* save the variables to file
save all like mp* to getpavcn
return
```

```
*
* SUBSYSTEM:      PRINT SUMMARY REPORT
* PROGRAM NAME:   GETSKID.PRG    01/26/88
* PROJECT 2456 - TEXAS FLEXIBLE PAVEMENT DATABASE CONVERSION
* TAMU/TTI
* AUTHOR:         TREVOR X. PEREIRA
* PURPOSE:        TO GET DATA FROM THE FOLLOWING FILES:
*                    1)  SKID dBASE FILE
*                    2)  DYNAFLECT dBASE FILE
*                    3)  FALLING WEIGHT dBASE FILE
*

****************
* GET SKID DATA
****************

@ 11, 5  SAY "Getting the Skid data ...."
X = 1
DO WHILE X < 7
     S = STR(X,1)
     MKDATE&S = "     "
     MKMEAN&S = "  "
     MKHIGH&S = "  "
     MKLOW&S  = "  "
     X = X + 1
ENDDO

USE \PAVEDB\FILES\SKID INDEX \PAVEDB\INDEXES\SKID

SEEK str(MSID_NO,4)
IF FOUND()
   DO WHILE .NOT. EOF() .AND. SID_NO = MSID_NO
     SKIP
   ENDDO
   SKIP -1
   X = 1
   DO WHILE .NOT. EOF() .AND. SID_NO = MSID_NO
     IF X > 6
        EXIT
     ENDIF
     S = STR(X,1)
     MYEAR = STR(YEAR,2)
     MMONTH = STR(MONTH,2)
     MKDATE&S = MMONTH +"/"+ MYEAR
     MKMEAN&S = STR(SKIDNUMM,2)
     MKHIGH&S = STR(SKIDNUMH,2)
     MKLOW&S  = STR(SKIDNUML,2)
     X = X + 1
     SKIP -1
     IF BOF()
       SKIP 18
```

```
      EXIT
    ENDIF
  ENDDO
ENDIF
SAVE ALL LIKE MK* TO GETSKID


*************
* GET DYNAFLECT DATA
*************

@ 12, 5  SAY  "Getting the Dynaflect data ...."
USE \PAVEDB\FILES\DYNAFLLD INDEX \PAVEDB\INDEXES\DYNAFLLD
SEEK STR(MSID_NO,4)
IF FOUND()
  MDAY   = LTRIM(STR(DAY,2))
  MMONTH = LTRIM(STR(MONTH,2))
  MYEAR  = STR(YEAR,2)
  MDDATE = MMONTH + "/" + MDAY + "/" + MYEAR
  MW1 = 0
  MW2 = 0
  MW3 = 0
  MW4 = 0
  MW5 = 0
  DO WHILE .NOT. EOF() .AND. SID_NO = MSID_NO
    MW1 = MW1 + SENS1RD
    MW2 = MW2 + SENS2RD
    MW3 = MW3 + SENS3RD
    MW4 = MW4 + SENS4RD
    MW5 = MW5 + SENS5RD
    SKIP
  ENDDO
  MW1 = MW1/14
  MW2 = MW2/14
  MW3 = MW3/14
  MW4 = MW4/14
  MW5 = MW5/14
  MDD  = "D"
  MDW1 = STR(MW1,5,2)
  MDW2 = STR(MW2,5,2)
  MDW3 = STR(MW3,5,2)
  MDW4 = STR(MW4,5,2)
  MDW5 = STR(MW5,5,2)
ELSE
  MDW1 = "     "
  MDW2 = "     "
  MDW3 = "     "
  MDW4 = "     "
  MDW5 = "     "
  MDD  = " "
  MDDATE = "        "
ENDIF
SAVE ALL LIKE MD* TO GETDYNA
```

```
*****************
* GET FALLING WEIGHT DATA
*****************
@ 13, 5  SAY "Getting Falling Weight data ...."
USE \PAVEDB\FILES\FALLWGHT INDEX \PAVEDB\INDEXES\FALLWGHT
SEEK STR(MSID_NO,4)
IF FOUND()
   MDAY   = LTRIM(STR(DAY,2))
   MMONTH = LTRIM(STR(MONTH,2))
   MYEAR  = STR(YEAR,2)
   MFDATE = MMONTH + "/" + MDAY + "/" + MYEAR
   MW1 = SSIGP11+SSIGP21+SSIGP31+SSIGP41+SSIGP51
   MW2 = SSIGP12+SSIGP22+SSIGP32+SSIGP42+SSIGP52
   MW3 = SSIGP13+SSIGP23+SSIGP33+SSIGP43+SSIGP53
   MW4 = SSIGP14+SSIGP24+SSIGP34+SSIGP44+SSIGP54
   MW5 = SSIGP15+SSIGP25+SSIGP35+SSIGP45+SSIGP55
   MW6 = SSIGP16+SSIGP26+SSIGP36+SSIGP46+SSIGP56
   MW7 = SSIGP17+SSIGP27+SSIGP37+SSIGP47+SSIGP57
   MW1 = MW1/5
   MW2 = MW2/5
   MW3 = MW3/5
   MW4 = MW4/5
   MW5 = MW5/5
   MW6 = MW6/5
   MW7 = MW7/5
   MFW1 = STR(MW1,5,2)
   MFW2 = STR(MW2,5,2)
   MFW3 = STR(MW3,5,2)
   MFW4 = STR(MW4,5,2)
   MFW5 = STR(MW5,5,2)
   MFW6 = STR(MW6,5,2)
   MFW7 = STR(MW7,5,2)
   MFD  = "F"
ELSE
   MFW1 = "      "
   MFW2 = "      "
   MFW3 = "      "
   MFW4 = "      "
   MFW5 = "      "
   MFW6 = "      "
   MFW7 = "      "
   MFD  = " "
   MFDATE = "         "
ENDIF

SAVE ALL LIKE MF* TO GETFALL
CLOSE DATABASES
RETURN
```

```
*
* SUBSYSTEM:      PRINT SUMMARY REPORT
* PROGRAM NAME:   PRNTREPT.PRG        02/02/88
* MODIFIED ON:    09/08/88
* PROJECT 2456 - TEXAS FLEXIBLE PAVEMENT DATABASE CONVERSION
* TAMU/TTI
* AUTHOR:         TREVOR X. PEREIRA
* PURPOSE:        TO PRINT OUT THE SUMMARY REPORT USING THE DATA STORED
*                 IN MEMORY VARIABLE FILES.
*

STORE "N" TO MANSWER
CLEAR
ON ERROR DO \PAVEDB\REPORTS\ERR_PRNT
SET PRINT TO &MPORT1
ON ERROR DO \PAVEDB\ERROR
@ 10, 10  SAY "Printing report for SID " + STR(MSID_NO,4)
mdate = dtoc(date())
SET DEVICE TO PRINT
@ 1,115 say "Date: " + mdate
@ 2, 0    say " _____________________________________________________
_____________________________________________-
_____________________________________"
@ 3, 0    say "|LOCATION                                                | |ENVIRONMENT -
20 YEAR SUMMARY  (1955-1974)                                             |"

************
* PRINT LOCATION SECTION
************
RESTORE FROM GETLOCAT ADDITIVE
@ 4, 0    SAY "|    SECTION ID NO:     "+LTRIM(STR(MSID_NO,4))
@ 4, 47   SAY "|  |"
@ 4, 131 SAY "|"
@ 5, 0    SAY "|    DISTRICT NO:       "+MLHWYDST
RELEASE MLHWYDST
@ 5, 47   SAY "|  |"
@ 5, 72   SAY "JAN FEB MAR APR MAY JUN JUL AUG SEP OCT NOV DEC    ANN      |"
@ 6, 0    SAY "|    COUNTY NO/NAME:    "+MLCNTYNM+"/"+MLCNTYNAM

******************
* PRINT ENVIRONMENT SECTION
******************
RESTORE FROM GETENVIR ADDITIVE
@ 6, 47   SAY "|  |  THORNTHWAITE INDEX:  -   -   -   -   -   -   -   -   -   -
 -   -"
@ 6, 122 SAY METHORN
@ 6, 131 SAY "|"
@ 7, 0    SAY "|    CONTROL-SECTION:   "+MLCONTSEC
@ 7, 47   SAY "|  |  MEAN TEMPERATURE:    "+MEMEAN1+" "+MEMEAN2+" "+MEMEAN3+"
"+MEMEAN4+" "+MEMEAN5+" "+MEMEAN6+" "+MEMEAN7
@ 7, 100 SAY MEMEAN8+" "+MEMEAN9+" "+MEMEAN10+" "+MEMEAN11+" "+MEMEAN12+"
```

```
"+MEMTAVER
@ 7, 131 SAY "|"
@ 8, 0   SAY "|    HIGHWAY:           "+MLHWY
@ 8, 47  SAY "|  |  PRECIPITATION:"
@ 8, 72  SAY MEPREC1+" "+MEPREC2+" "+MEPREC3+" "+MEPREC4+" "+MEPREC5+"
"+MEPREC6+" "+MEPREC7+" "+MEPREC8+" "+MEPREC9
@ 8, 108 SAY MEPREC10+" "+MEPREC11+" "+MEPREC12+"    "+MEPRECIP
@ 8, 131 SAY "|"
@ 9, 0   SAY "|    MILE POSTS:        "+MLPOST
@ 9, 47  SAY "|  |  WET F-T CYCLES:"
@ 9, 72  SAY MEWFTC1+" "+MEWFTC2+" "+MEWFTC3+" "+MEWFTC4+" "+MEWFTC5+"
"+MEWFTC6+" "+MEWFTC7+" "+MEWFTC8+" "+MEWFTC9
@ 9, 108 SAY MEWFTC10+" "+MEWFTC11+" "+MEWFTC12+"     "+MEWETFTC
@ 9, 131 SAY "|"
@ 10, 0   SAY "|    LANE:              "+MLLANEID
@ 10, 47  SAY "|  |  TOTAL F-T CYCLES:"
@ 10, 72  SAY METFTC1+" "+METFTC2+" "+METFTC3+" "+METFTC4+" "+METFTC5+"
"+METFTC6+" "+METFTC7+" "+METFTC8+" "+METFTC9
@ 10, 108 SAY METFTC10+" "+METFTC11+" "+METFTC12+"    "+METOTFTC
@ 10, 131 SAY "|"
@ 11, 0   SAY "|    PREVIOUS SID:      "+MLPREVSD
@ 11, 47  SAY "|  |  DIST TEMP CONSTANT: - - - - - - - - - -
- - -       "+METEMPCN

RELEASE ALL LIKE ME*
@ 11, 131 SAY "|"
@ 12, 0   SAY "|    NEXT SID:         "+MLNEXTSD
@ 12, 47  SAY "|  |"
@ 12, 131 SAY "|"
@ 13, 0   SAY "|    FUNCTIONAL CLASS: "+MLFUNCLS
@ 13, 47  SAY "|  |_____________________________________________-
_____________________________|"
@ 14, 0   SAY "|    TYPE OF PAVEMENT: "+MLBASETYP
@ 14, 47  SAY "|"
@ 15, 0   SAY "|                      "+MLBASETHK
@ 15, 47  SAY "|  ___________________________________
                                    "
@ 16, 0   SAY "|"
@ 16, 22  SAY "    "+MLBASESEAL
@ 16, 47  SAY "|  |TRAFFIC"
@ 16, 85  SAY "|  |SERVICEABILITY INDEX"
@ 16, 131 SAY "|"
@ 17, 0   SAY "|"
@ 17, 47  SAY "|  |"
@ 17, 22  SAY MLWIDEN
@ 17, 85  SAY "|  |"
@ 17, 131 SAY "|"
RELEASE ALL LIKE ML*

************
* PRINT TRAFFIC SECTION
************
RESTORE FROM GETTRAFFIC ADDITIVE
@ 18,  0  SAY "|"
```

```
@ 18, 47  SAY "|  |"
@ 18, 49  SAY "|   "+MTEYEAR+" ADT:"
@ 18, 78  SAY MTAADT
@ 18, 85  SAY "|  |   YR  MEAN  STD DEV N    CV    LOW   HIGH  |"
@ 19,  0  SAY "|"
@ 19, 47  SAY "|  |   "+MTEYEAR
@ 19, 57  SAY "PERCENT TRUCKS:"
@ 19, 80  SAY MTPTRK

************
* PRINT SERVICEABILITY SECTION
************
RESTORE FROM GETSERVC ADDITIVE
@ 19, 85  SAY "|  |   "+MSYEAR1+"   "+MSMEAN1+"    "+MSSTDD1+"    "+MSCONT1+"
"+MSCOEF1+"    "+MSLOWV1+"   "+MSHIGH1+"   |"
@ 20,  0  SAY "|"
@ 20, 47  SAY "|  |"
@ 20, 49  SAY "|"
@ 20, 52  SAY MTBYEAR+"-"+MTEYEAR
@ 20, 63  SAY "VEHICLES:  "+MTVEHIC
@ 20, 85  SAY "|  |   "+MSYEAR2+"   "+MSMEAN2+"    "+MSSTDD2+"    "+MSCONT2+"
"+MSCOEF2+"    "+MSLOWV2+"   "+MSHIGH2+"   |"
RELEASE ALL LIKE MS????2, MS????1

************
* PRINT PAVEMENT CONDITION SURVEY SECTION
************
RESTORE FROM GETPAVCN ADDITIVE
@ 21, 0   SAY "|_____________________________________________|  |"
@ 21, 52  SAY MTBYEAR+"-"+MTEYEAR
@ 21, 63  SAY "18K AXLES:  "+MT18KEAL
RELEASE ALL LIKE MT*
@ 21, 85  SAY "|  |   "+MSYEAR3+"   "+MSMEAN3+"    "+MSSTDD3+"    "+MSCONT3+"
"+MSCOEF3+"    "+MSLOWV3+"   "+MSHIGH3+"   |"
@ 22,  0  SAY "                                             "
@ 22, 49  SAY "|"
@ 22, 85  SAY "|  |   "+MSYEAR4+"   "+MSMEAN4+"    "+MSSTDD4+"    "+MSCONT4+"
"+MSCOEF4+"    "+MSLOWV4+"   "+MSHIGH4+"   |"
@ 23,  0  SAY "|PAVEMENT CONDITION SURVEY"
@ 23, 47  SAY "|  |"
@ 23, 85  SAY "|  |   "+MSYEAR5+"   "+MSMEAN5+"    "+MSSTDD5+"    "+MSCONT5+"
"+MSCOEF5+"    "+MSLOWV5+"   "+MSHIGH5+"   |"
@ 24,  0  SAY "|  PVMT RATING  "+MPACTYR1+"    "+MPACTYR2+"    "+MPACTYR3+"
"+MPACTYR4+"    "+MPACTYR5+"    "+MPACTYR6+"     |  |"
@ 24, 85  SAY "|  |   "+MSYEAR6+"   "+MSMEAN6+"    "+MSSTDD6+"    "+MSCONT6+"
"+MSCOEF6+"    "+MSLOWV6+"   "+MSHIGH6+"   |"
@ 25,  0  SAY "|    PRS"
@ 25, 15  SAY MPPRS1+"  "+MPPRS2+"  "+MPPRS3+"  "+MPPRS4+"  "+MPPRS5+"
"+MPPRS6
@ 25, 47  SAY "|  |_____________________________________|"
@ 25, 87  SAY "|_____________________________________________|"
RELEASE ALL LIKE MS*
@ 26,  0  SAY "|    RUTT"
@ 26, 16  SAY MPRUTIN1+MPRUTTA1+"  "+MPRUTIN2+MPRUTTA2+"
```

```
"+MPRUTIN3+MPRUTTA3+"  "+MPRUTIN4+MPRUTTA4+"  "+MPRUTIN5+MPRUTTA5+"  "
@ 26, 41  SAY MPRUTIN6+MPRUTTA6+"  |"
@ 27,  0  SAY "|    BLOCK CR   "+MPBLCKN1+MPBLCKA1+"  "+MPBLCKN2+MPBLCKA2+"
"+MPBLCKN3+MPBLCKA3+"  "+MPBLCKN4+MPBLCKA4+"  "
@ 27, 36  SAY MPBLCKN5+MPBLCKA5+"  "+MPBLCKN6+MPBLCKA6+"  |"
@ 27, 47  SAY "  ____________________________"
@ 27, 79  SAY "                                                              "
@ 28,  0  SAY "|    ALLG CR    "+MPALLGN1+MPALLGA1+"  "+MPALLGN2+MPALLGA2+"
"+MPALLGN3+MPALLGA3+"  "+MPALLGN4+MPALLGA4+"  "
@ 28, 36  SAY MPALLGN5+MPALLGA5+"  "+MPALLGN6+MPALLGA6+"  |"
RELEASE ALL LIKE MPPRS?, MPRUTIN?, MPBLCKN?, MPACTYR?
@ 28, 47  SAY "| |SKID NUMBER"
@ 28, 76  SAY "| |DEFLECTION (MEAN VARIABLES)"
@ 28, 131 SAY "|"
@ 29,  0  SAY "|    LONG CR    "+MPLONGN1+MPLONGA1+"  "+MPLONGN2+MPLONGA2+"
"+MPLONGN3+MPLONGA3+"  "+MPLONGN4+MPLONGA4+"  "
@ 29, 36  SAY MPLONGN5+MPLONGA5+"  "+MPLONGN6+MPLONGA6+"  | |"
@ 29, 52  SAY "DATE    AVG  LOW  HIGH  | |"
@ 29, 131 SAY "|"
@ 30,  0  SAY "|    TRANS CR   "+MPTRANN1+MPTRANA1+"  "+MPTRANN2+MPTRANA2+"
"+MPTRANN3+MPTRANA3+"  "+MPTRANN4+MPTRANA4+"  "
@ 30, 36  SAY MPTRANN5+MPTRANA5+"  "+MPTRANN6+MPTRANA6+"  | |"
RELEASE ALL LIKE MPALLG*, MPLONG*, MPTRAN*

*****************
* PRINT SKID DATA
*****************
RESTORE FROM GETSKID ADDITIVE
@ 30, 51  SAY MKDATE1
@ 30, 60  SAY MKMEAN1+"  "+MKLOW1+"  "+MKHIGH1
@ 30, 76  SAY "| | DATE    D   W1    W2    W3    W4    W5    W6    W7 |"
@ 31,  0  SAY "|    CRACKS     "+MPCRACK1+"  "+MPCRACK2+"  "+MPCRACK3+"
"+MPCRACK4+"  "+MPCRACK5+"  "+MPCRACK6
@ 31, 47  SAY "| |"
@ 31, 51  SAY MKDATE2
@ 31, 60  SAY MKMEAN2+"  "+MKLOW2+"  "+MKHIGH2
@ 31, 76  SAY "| |"

*****************
*  PUT IN DEFLECTION DATA
*****************
RESTORE FROM GETDYNA ADDITIVE
@ 31, 79  SAY MDDATE
@ 31, 88  SAY MDD+" "+MDW1+" "+MDW2+" "+MDW3+" "+MDW4+" "+MDW5
RELEASE ALL LIKE MD*
@ 31, 131 SAY "|"
@ 32,  0  SAY "|    PATCHING   "+MPPATCN1+MPPATCA1+"  "+MPPATCN2+MPPATCA2+"
 "+MPPATCN3+MPPATCA3+"  "+MPPATCN4+MPPATCA4+"  "
@ 32, 36  SAY MPPATCN5+MPPATCA5+"  "+MPPATCN6+MPPATCA6+"  | |"
@ 32, 51  SAY MKDATE3
@ 32, 60  SAY MKMEAN3+"  "+MKLOW3+"  "+MKHIGH3
@ 32, 76  SAY "| |"
RESTORE FROM GETFALL ADDITIVE
@ 32, 79  SAY MFDATE
```

```
@ 32, 88  SAY MFD+" "+MFW1+" "+MFW2+" "+MFW3+" "+MFW4+" "+MFW5+" "+MFW6+"
"+MFW7
RELEASE ALL LIKE MF*
@ 32, 131 SAY "|"
@ 33,  0  SAY "|    FAIL/MI     "+MPFAILML1+"    "+MPFAILML2+"
"+MPFAILML3+"    "+MPFAILML4+"    "+MPFAILML5+"    "+MPFAILML6+"    | |"
RELEASE ALL LIKE MP*
@ 33, 51  SAY MKDATE4
@ 33, 60  SAY MKMEAN4+"   "+MKLOW4+"   "+MKHIGH4
@ 33, 76  SAY "| |"
@ 33, 131 SAY "|"
@ 34,  0  SAY "|"
@ 34, 47  SAY "| |"
@ 34, 51  SAY MKDATE5
@ 34, 60  SAY MKMEAN5+"   "+MKLOW5+"   "+MKHIGH5
@ 34, 76  SAY "| |"
@ 34, 131 SAY "|"
@ 35,  0  SAY "|"
@ 35, 47  SAY "| |"
@ 35, 51  SAY MKDATE6
@ 35, 60  SAY MKMEAN6+"   "+MKLOW6+"   "+MKHIGH6
RELEASE ALL LIKE MK*
@ 35, 76  SAY "| |"
@ 35, 131 SAY "|"
@ 36,  0  SAY "|"
@ 36, 47  SAY "| |"
@ 36, 76  SAY "| |"
@ 36, 131 SAY "|"
@ 37,  0  SAY "|_____________________________________________________|
|_____________|"
@ 37, 78  SAY "|_____________________________________________________|"
@ 39,  0  SAY "_____________________________________________________-
_____________________________________________________"
@ 40,  0  SAY "|STRUCTURAL SECTION"
@ 40, 60  SAY "AGG. ====ADMIXTURE====  APPL  THICK"
@ 40, 131 SAY "|"
@ 41,  0  SAY "|  LAYER STRUCTURE DESCRIPTION DATE     MATERIAL TYPE
RATE  TYPE        PCNT   RATE   CENT    TTC     LL    PI"
@ 41, 131 SAY "|"

***********
* PRINT STRUCTURAL SECTION
***********
RESTORE FROM GETSTRUC ADDITIVE
X = 1
MROW = 42
DO WHILE X < 14
     IF X < 10
        S = STR(X,1)
     ELSE
        S = STR(X,2)
     ENDIF
     RESTORE FROM GETSTC&S ADDI
     @ MROW,  0  SAY "|   "+MTLAY&S
```

```
     @ MROW, 12  SAY MTSTRUC&S
     @ MROW, 21  SAY MZLAYDA&S
     @ MROW, 24  SAY MZLAYDB&S
     @ MROW, 31  SAY MTDATE&S
     @ MROW, 38  SAY MZMATTYP&S
     @ MROW, 60  SAY MTAGGRAT&S
     @ MROW, 65  SAY MTADMXTP&S
     @ MROW, 78  SAY MTADMXPR&S
     @ MROW, 85  SAY MTAPPLRT&S
     @ MROW, 91  SAY MTCENTHK&S
     IF MZCOUNT = X
          @ MROW, 101  SAY MZTXTRL
          @ MROW, 108  SAY MZLIQLIM
          @ MROW, 114 SAY MZPLAST
     ENDIF
     RELE ALL LIKE MT*
     MFILE = "GETSTC" + S + ".MEM"
     DELE FILE &MFILE
     @ MROW, 131 SAY "|"
     X = X + 1
     MROW = MROW + 1
ENDDO
RELEASE ALL LIKE MZ*

@ 55, 0   SAY "|"
@ 55, 131 SAY "|"
@ 56, 0   SAY
"|_____________________________________________________________________________-
_____________________________________________________________|"

* AT END DELETE ALL MEM FILES

CLEAR
SET PRINT OFF
SET DEVICE TO SCREEN
DELETE FILE GETENVIR.MEM
DELETE FILE GETLOCAT.MEM
DELETE FILE GETSKID.MEM
DELETE FILE GETPAVCN.MEM
DELETE FILE GETSERVC.MEM
DELETE FILE GETTRAFF.MEM
DELETE FILE GETSTRUC.MEM
DELETE FILE GETDYNA.MEM
DELETE FILE GETFALL.MEM

@ 15, 20 SAY "DONE ....."
RETURN
```

```
*
* SUBSYSTEM:      PRINT SUMMARY REPORT
* PROGRAM NAME:   SUM2DIST.PRG        02/04/88
* PROJECT 2456 - TEXAS FLEXIBLE PAVEMENT DATABASE CONVERSION
* TAMU/TTI
* AUTHOR:         TREVOR X. PEREIRA
* PURPOSE:        TO PRINT OUT THE INVENTORY UPDATE REPORT.  THE FOLLOWING
*                 PROGRAMS ARE CALLED:
*                              - GETLOCT2.PRG
*                              - PRTREPT2.PRG
*                              - PRTREPT3.PRG
*

* the program first gets all the SID numbers for the particular district and
* stores them in a temporary file.  Then data for each SID number is stored
* in memory variable files.  When the report is printed out, the appropriate
* files with the variables are opened and deleted when done with them.  This
* process continues for each SID number in the temporary file.

* set all parameters
CLOSE DATABASES
CLEAR
SET PRINTER TO &MPORT2
PUBLIC MSID_NO, VALID, MCOUNTY, MDISTRICT
DEVICEON = "SET DEVICE TO PRINT"
DEVICEOFF = "SET DEVICE TO SCREEN"
STORE " " TO MQUIT, MDISTT
STORE .F. TO VALID
STORE 0 TO MSID_NO, MDISTT

* get the sid number
DO WHILE .NOT. VALID
   CLEAR
   CLEAR TYPEAHEAD
   @ 10, 5 SAY " "
   ACCEPT "Please enter District number: " TO MDISTT
   IF READKEY() = 12
      CLEAR
      RETURN
   ENDIF
   MDIST = VAL(MDISTT)
   @ 10, 0 CLEAR

   * Get all SID_NO's for the required district and store them to a temporary
   * dBASE file
   SELECT 1
      USE \PAVEDB\FILES\LOCATION index \PAVEDB\INDEXES\locsid
   SELECT 2
      IF .NOT. FILE('\PAVEDB\FILES\SIDSTOR2.DBF')
          CLEAR
          ? "Temporary SID storage file (SIDSTOR2.DBF) not found. Please
```

```
check . . ."
          WAIT
          RETURN TO MASTER
       ENDIF
       USE \PAVEDB\FILES\SIDSTOR2
       DELE ALL
       PACK

    SELECT 1
    LOCATE FOR HWYDIST = MDIST
    IF .NOT. FOUND()
      @ 12, 5 SAY "Not found.  Please try again."
      WAIT
    ENDIF
    DO WHILE FOUND()
      MSID_NO = A->SID_NO
      SELE 2
      APPEND BLANK
      REPLACE B->SID_NO WITH MSID_NO
      SELE 1
      VALID = .T.
      CONTINUE
    ENDD
ENDDO
SELE 1
   USE
SELE 2
   APPEN BLANK
   REPLACE B->SID_NO WITH 0
   USE

* Print out the reports
USE \PAVEDB\FILES\SIDSTOR2
MREC = 1
IF SID_NO = 0
  STORE .F. TO GOON
ELSE
  STORE .T. TO GOON
ENDIF
DO WHILE GOON
  MSID_NO = SID_NO
  DO \PAVEDB\REPORTS\GETLOCT2
  DO \PAVEDB\REPORTS\PRTREPT2
  DO \PAVEDB\REPORTS\PRTREPT3
  MREC = MREC + 1
  USE \PAVEDB\FILES\SIDSTOR2
  GOTO MREC
  IF SID_NO = 0
    GOON = .F.
  ENDIF
  CLEAR
ENDDO
SELE 1
  USE
```

```
SELE 2
  USE
SELE 3
  USE

CLOSE DATABASES
SET PRINTER TO LPT1
RELE MSID_NO, VALID, MCOUNTY, MDISTRICT
RETURN
```

```
*
* SUBSYSTEM:      PRINT INVENTORY UPDATE REPORT
* PROGRAM NAME:   SUMM2SID.PRG        01/21/88
* PROJECT 2456 - TEXAS FLEXIBLE PAVEMENT DATABASE CONVERSION
* TAMU/TTI
* AUTHOR:         TREVOR X. PEREIRA
* PURPOSE:        TO PRINT OUT THE INVENTORY UPDATE REPORT.  THE FOLLOWING
*                 PROGRAMS ARE CALLED:
*                              - GETLOCT2.PRG
*                              - PRTREPT2.PRG
*                              - PRTREPT3.PRG
*

* the program first gets all the data and stores it in separate files
* using memory variables.  When the report is printed out, the appropriate
* files with the variables are opened and closed when done with them.

* set all parameters
SET TALK OFF
SET ECHO OFF
CLOSE DATABASES
CLEAR
DEVICEON = "SET DEVICE TO PRINT"
DEVICEOFF = "SET DEVICE TO SCREEN"
SET PRINTER TO &MPORT2
PUBLIC MSID_NO, VALID, MCOUNTY, MDISTRICT
STORE " " TO MQUIT
STORE .F. TO VALID
STORE 0 TO MSID_NO

* get the sid number
DO WHILE .NOT. VALID
   CLEAR
   @ 10, 5 SAY " "
   ACCEPT "Please enter Section Identification (SID) number: " TO SID_NO
   IF READKEY() = 12
      CLEAR
      RETURN
   ENDIF
   @ 0, 0  CLEAR
  MSID_NO = VAL(SID_NO)

   * check the sid number
   STORE 0 TO VAR1, VAR2, VAR3, COMPARE

   * calculates the correct Sid Number
   VAR1 = INT(MSID_NO/1000)
   VAR2 = MOD(INT(MSID_NO/100),10)
   VAR2 = VAR2 * 2
   VAR3 = MOD(INT(MSID_NO/10),10)
   VAR3 = VAR3 * 3
```

```
   VAR4 = MOD(VAR1,10)+VAR2+VAR3
   COMPARE = MOD(VAR4,10)

   * compares calculated Sid number with Sid number entered
   IF COMPARE = MOD(MSID_NO,10)
       STORE .T. TO VALID
   ELSE
       @ 19, 10 SAY "Invalid SID Number"
       WAIT
       @ 18, 10 CLEAR
   ENDIF

   * verifies that Sid number has been entered
   IF MSID_NO = 0
       STORE .F. TO VALID
       @ 19, 10 SAY "Invalid SID Number"
       WAIT
       @ 18, 10 CLEAR
   ENDIF
ENDDO

* get all the data and print the report
IF VALID
   CLEAR
   DO \PAVEDB\REPORTS\GETLOCT2
   DO \PAVEDB\REPORTS\PRTREPT2
   DO \PAVEDB\REPORTS\PRTREPT3
ENDIF

RELEASE MSID_NO, VALID, MCOUNTY, MDISTRICT
CLOSE DATABASES
RETURN
```

```
    MSID_NO = SID_NO
    DO \PAVEDB\REPORTS\GETLOCT2
    DO \PAVEDB\REPORTS\PRTREPT2
    DO \PAVEDB\REPORTS\PRTREPT3
    MREC = MREC + 1
    USE \PAVEDB\FILES\LOCATN2
    GOTO MREC
    IF SID_NO = 0
      GOON = .F.
    ENDIF
    CLEAR
ENDDO
SELE 1
  USE
SELE 2
  USE
SELE 3
  USE
CLOSE ALTE
RELEASE MSID_NO, MCOUNTY, MDISTRICT
CLOSE DATABASES
RETURN
```

```
*
* SUBSYSTEM:      PRINT INVENTORY UPDATE REPORT
* PROGRAM NAME:   SUM2FILE.PRG       06/14/88
* PROJECT 2456 - TEXAS FLEXIBLE PAVEMENT DATABASE CONVERSION
* TAMU/TTI
* AUTHOR:         TREVOR X. PEREIRA
* PURPOSE:        TO PRINT OUT THE INVENTORY UPDATE REPORT FOR ALL THE SID
*                 NUMBERS.  THE FOLLOWING PROGRAMS ARE CALLED:
*                              - GETLOCT2.PRG
*                              - PRTREPT2.PRG
*                              - PRTREPT3.PRG
*

* the program first gets all the data and stores it in separate files
* using memory variables.  When the report is printed out, the appropriate
* files with the variables are opened and closed when done with them.
* THE REPORT FOR EACH SID IS PRINTED OUT TO A FILE.

* set all parameters
SET TALK OFF
SET ECHO OFF
CLOSE DATABASES
CLEAR
SET PRINTER TO &MPORT2
DEVICEON = "SET DEVICE TO PRINT"
DEVICEOFF = "SET DEVICE TO SCREEN"
PUBLIC MSID_NO, MCOUNTY, MDISTRICT
STORE " " TO MQUIT
STORE 0 TO MSID_NO

* get the sid number
IF FILE('\PAVEDB\FILES\LOCATN2.DBF')
     DELE FILE \PAVEDB\FILES\LOCATN2.DBF
ENDIF
COPY FILE \PAVEDB\FILES\LOCATION.DBF TO \PAVEDB\FILES\LOCATN2.DBF
USE \PAVEDB\FILES\LOCATN2
APPEND BLANK
REPLACE SID_NO WITH 0
GOTO TOP
MREC = 1
IF SID_NO = 0
  STORE .F. TO GOON
ELSE
  STORE .T. TO GOON
ENDIF
CLEAR
DO WHILE GOON
   IF READKEY() = 12
      CLEAR
      RETURN
   ENDIF
```

```
* SUBSYSTEM:      PRINT INVENTORY UPDATE REPORT
* PROGRAM NAME:   GETLOCT2.PRG          02/04/88
* PROJECT 2456 - TEXAS FLEXIBLE PAVEMENT DATABASE CONVERSION
* TAMU/TTI
* AUTHOR:         TREVOR X. PEREIRA
* PURPOSE:        TO GET DATA FROM THE FOLLOWING FILES TO PRINT THE SUMMARY
*                 REPORT:      1) LOCATION dBASE FILE
*                              2) LAYER IDENTIFICATION dBASE FILE
*                              3) SUBGRADE LAYER dBASE FILE
*                              4) SURFACE LAYER dBASE FILE
*                              5) GEOMETRIC & SHOULDER INFO LAYER dBASE FILE
*                              6) LAYER THICKNESS ACROSS THE ROAD
*

************************
* GET LOCATION PART OF THE REPORT AND STORE THE DATA IN A FILE
************************

@ 2,14 SAY "PROCESSING SID NUMBER " + STR(MSID_NO,4)
@ 5, 5 SAY "Getting data for the Location section of the report ...."

close all

* assign databases to different work areas
SELECT 1
use \PAVEDB\FILES\location INDEX \PAVEDB\INDEXES\LOCSID

SELECT 2
USE \PAVEDB\FILES\CNTYTBL INDEX \PAVEDB\INDEXES\CTYTBLNO

* locate for the requested sid number
SELECT 1
seek msid_no
if .not. found()
   okay = .F.
   clear
   @ 19, 10 say "SID number not found.  Please try again ..."
   WAIT
   @ 18, 10 CLEAR
   RETURN
else
   okay = .T.
endif
STORE 0 TO MCOUNTY
STORE 0 TO MDISTRICT

* if the sid number is found, get all the required data
if Okay
   mlhwydst = LTRIM(STR(hwydist,2))
   MDISTRICT = HWYDIST
```

```
    MCOUNTY = CNTYNUM
    mlcntynm = LTRIM(STR(cntynum,3))
    SELECT 2
    SEEK A->CNTYNUM
    IF FOUND()
      MLCNTYNAM = LTRIM(CNTYNAME)
    ENDIF
    SELECT 1
    mcontrl = LTRIM(STR(control,4))
    msectn  = LTRIM(STR(section,2))
    MLCONTSEC = MCONTRL + "-" + MSECTN
    MLNUMLAN = LTRIM(STR(NUMLANES,2))
    mprefx  = hwyprefx
    mhwynum = LTRIM(STR(hwynum,4))
    msuffx  = hwysuffx
    MLHWY = MPREFX + " " + LTRIM(MHWYNUM) + " " + MSUFFX
    MBEGP = LTRIM(STR(BEGMPST,3))
    MBEGD =
RIGHT((STUFF(STR(BMPSTDIS,2),1,(2-LEN(LTRIM(STR(BMPSTDIS,2)))),"0")),2)
    MPOSTB = MBEGP+BDISSIGN+MBEGD
    MENDD =
RIGHT((STUFF(STR(EMPSTDIS,2),1,(2-LEN(LTRIM(STR(EMPSTDIS,2)))),"0")),2)
    MENDP = LTRIM(STR(ENDMPST,3))
    MPOSTE = MENDP+EDISSIGN+MENDD
    MLPOST = MPOSTB+" TO "+MPOSTE
    IF ACTVFLAG
      MLINACTV = "NO"
    ELSE
      MLINACTV = "YES"
    ENDIF
    mlbegmpn = LTRIM(STR(begmpnt,6,3))
    mlendmpn = LTRIM(STR(endmpnt,6,3))
    mllaneid = laneid
    IF PREVSID = 0
      MLPREVSD = "-"
    ELSE
      mlprevsd = LTRIM(STR(prevsid,4))
    ENDIF
    IF NEXTSID = 0
      MLNEXTSD = "-"
    ELSE
      mlnextsd = LTRIM(STR(nextsid,4))
    ENDIF
    MLFUNCLS = LTRIM(STR(FUNCLAS,2))
endif
SELECT 1
USE
SELECT 2
USE

*****************
* get PAVEMENT INFORMATION FROM geometric and shoulder LAYER FILE
*****************
```

```
SELECT 1
use \PAVEDB\FILES\geosho index \PAVEDB\INDEXES\geondx

SELECT 2
USE \PAVEDB\FILES\PAVETYPE

SELECT 1
LOCATE FOR SID_NO = msid_no .AND. STRUCNUM = 1
if found()
   DO CASE
     CASE WIDENFLG = 0
       MLWIDEN = "NO WIDENING"
     CASE WIDENFLG = 1
       MLWIDEN = "WIDENING"
     CASE WIDENFLG = 2
       MLWIDEN = "UNUSUAL WIDENING"
   ENDCASE
   MPAVEM = PAVETYP
   SELECT 2
   LOCATE FOR PAVECODE = MPAVEM
   IF FOUND()
     MLBASETYP = LTRIM(BASETYPE)
     MLBASETHK = LTRIM(BASETHK)
     MLBASESEAL = LTRIM(BASESEAL)
   ELSE
     MLBASETYP = " "
     MLBASETHK = " "
     MLBASESEAL = " "
   ENDIF
ELSE
   MLWIDEN = " "
   MLBASETYP = " "
   MLBASETHK = " "
   MLBASESEAL = " "
ENDIF
* save all location variables to a file
SAVE ALL LIKE ML* TO GETLOCT2
RELEASE ALL
SELE 1
USE
SELE 2
USE


**********************
* GET LAYER THICKNESS ACROSS THE ROAD INFORMATION
**********************
USE \PAVEDB\FILES\LAYTHICK INDEX \PAVEDB\INDEXES\LAYINDX
SEEK STR(MSID_NO,4)
IF FOUND()
     DO WHILE .NOT. EOF() .AND. SID_NO = MSID_NO
          SKIP
     ENDDO
     SKIP -1
```

```
     DO CASE
        CASE FC3DIS <> 0
          MDTHICK = STR(FC3DIS,4,1)
        CASE FC2DIS <> 0
          MDTHICK = STR(FC2DIS,4,1)
        CASE FC1DIS <> 0
          MDTHICK = STR(FC1DIS,4,1)
        OTHERWISE
          MDTHICK = "  "
     ENDCASE
ELSE
     MDTHICK = "  "
ENDIF
SAVE ALL LIKE MD* TO GETDISTN
RELEASE ALL LIKE MD*


**********************
* GET OTHER GEOMETRIC AND SHOULDER INFORMATION
**********************
use \PAVEDB\FILES\geosho index \PAVEDB\INDEXES\geondx
SEEK STR(msid_no,4)
X = 0
DO WHILE X < 16
     IF X < 10
        S = STR(X,1)
     ELSE
        S = STR(X,2)
     ENDIF
     MGLANEWD&S = "    "
     MGOUTSHD&S = "    "
     MGSHDSUR&S = " "
     MGSHDBSE&S = "  "
     MGSHDSFT&S = "     "
     MGSHDBST&S = "     "
     MGNUMLAN&S = " "
     MGSTRUC&S  = "  "
     MGPAVE&S   = "  "
     MGWIDEN&S  = " "
     X = X + 1
ENDDO

if found()
   X = 0
   DO WHILE .NOT. EOF() .AND. SID_NO = MSID_NO
     IF X > 15
        EXIT
     ENDIF
     IF X < 10
        S = STR(X,1)
     ELSE
        S = STR(X,2)
     ENDIF
     MGLANEWD&S = STR(LANEWID,4,1)
```

```
      MGOUTSHD&S = STR(OUTSHOWD,4,1)
      MGSHDSUR&S = STR(SHOSFTYP,1)
      MGSHDBSE&S = STR(SHOBSTYP,2)
      MGSHDSFT&S = STR(SHOSFTHK,5,2)
      MGSHDBST&S = STR(SHOBSTHK,5,2)
      MGSTRUC&S  = STR(STRUCNUM,2)
      MGPAVE&S   = STR(PAVETYP,2)
      MGWIDEN&S  = STR(WIDENFLG,1)
      X = X + 1
      SKIP
   ENDDO
ENDIF
SAVE ALL LIKE MG* TO GETGEOSH
RELEASE ALL LIKE MG*
USE

*******************
* GET STRUCTURAL SECTION OF THE REPORT AND SAVE THE DATA IN VARIABLES
*******************

* variable mcount is to find out how many layers there are for the SID #.
* This will help determine the layer the subgrade goes on.  This is needed
* if layer # 1 is the top most layer and not the subgrade.

@ 8, 5  SAY  "Getting data for the Structural Section ...."
STORE " " TO MTPLAST, MTLIQLIM, MTTXTRL
MCOUNT = 1

set path to \pavedb\files,\pavedb\indexes
set view to \pavedb\files\strcsec2
seek str(msid_no,4)

* if the sid number is found, get the required data
if found()
  do while .NOT. EOF() .AND. sid_no = msid_no
     skip
  enddo
  skip -1
  X = 1
  do while .NOT. EOF() .AND. sid_no = msid_no
      IF X < 10
         S = STR(X,1)
      ELSE
         S = STR(X,2)
      ENDIF
      mtstruc&S  = str(strucnum,2)
      mtlay&S    = str(laynum,2)
      mlaydes&S = laydesc
      MTDATE&S  = str(jobcmpmo,2)+"/"+ str(jobcmpyr,2)
      mlaymat&S = laymatcl
      IF WIDENLMO = 0 .OR. WIDENLYR = 0
          MTWIDAT&S = "     "
      ELSE
          MWIDEYR = STR(WIDENLYR,2)
```

```
            MWIDEMO = LTRIM(STR(WIDENLMO,2))
            MIWIDAT&S = MWIDEMO+"/"+MWIDEYR
         ENDIF
         IF CENTTHK = 0
            MTCENTHK&S = " "
         ELSE
            mtcenthk&S = str(centthk,5,2)
         ENDIF
         IF AGAPPLRT = 0
            MTAGGRAT&S = " "
         ELSE
            mtaggrat&S = str(agapplrt,3)
         ENDIF
         mtadmxtp&S = admxtyp
         IF ADMXPER = 0
            MTADMXPR&S = " "
         ELSE
            mtadmxpr&S = str(admxper,5,2)
         ENDIF
         IF ASAPPLRT = 0
            MTAPPLRT&S = " "
         ELSE
            mtapplrt&S = str(asapplrt,4,2)
         ENDIF
         SAVE ALL LIKE MT* TO GETSTC&S
         RELEASE ALL LIKE MT*
         X = X + 1
         MCOUNT = MCOUNT + 1
         skip -1
         IF BOF()
            SKIP 35
         ENDIF
   enddo
   SELE 1
   USE
   SELE 2
   USE
   SELE 3
   USE
   X = MCOUNT
   MCOUNT = MCOUNT - 1
   * initialize rest of variables
   DO WHILE  X < 14
      IF X < 10
         S = STR(X,1)
      ELSE
         S = STR(X,2)
      ENDIF
      STORE " " TO MISTRUC&S, MTLAY&S, MTJOBMO&S, MTJOBYR&S, MTCENTHK&S,
MTAGGRAT&S, MTADMXPR&S
      STORE " " TO MTAPPLRT&S, MTDATE&S, MLAYDES&S, MLAYMAT&S, MIWIDAT&S
      SAVE ALL LIKE MT* TO GETSTC&S
      RELEASE ALL LIKE MT*
      X = X + 1
```

```
  ENDDO

  * get the material type and the layer names from the tables
  SELECT 6
    USE \PAVEDB\FILES\MATLITBL
  SELECT 7
    USE \PAVEDB\FILES\LAYERTBL
  SELECT 6
  MZCOUNT = MCOUNT
  X = 1
  DO WHILE X < 14
     IF X < 10
        S = STR(X,1)
     ELSE
        S = STR(X,2)
     ENDIF
     SELECT 6
     TYPE1 = MLAYMAT&S
     TYPE2 = "TYPE1"
     TYPE3 = TYPE(TYPE2)
     IF TYPE3 = "N"
        LOCATE FOR MATCODE = MLAYMAT&S
        IF FOUND()
          MZMATTYP&S = F->MATDESC
          MZLAYDA&S  = F->LAYRDES
        ELSE
          MZMATTYP&S = " "
          MZLAYDA&S  = " "
        ENDIF
     ELSE
          MZMATTYP&S = " "
          MZLAYDA&S  = " "
     ENDIF
     SELECT 7
     TYPE1 = MLAYDES&S
     TYPE2 = "TYPE1"
     TYPE3 = TYPE(TYPE2)
     IF TYPE3 = "N"
       LOCATE FOR CODE = MLAYDES&S
       IF FOUND()
          MZLAYDB&S = G->CODE_DESC
       ELSE
          MZLAYDB&S = " "
       ENDIF
     ELSE
          MZLAYDB&S = " "
     ENDIF
     X = X + 1
  ENDDO
ELSE
  * initialize all variables
  X = 1
  DO WHILE  X < 14
     IF X < 10
```

```
        S = STR(X,1)
     ELSE
        S = STR(X,2)
     ENDIF
     STORE " " TO MTSTRUC&S, MTLAY&S, MTCENTHK&S, MTAGGRAT&S, MTADMXPR&S,
MZMATTYP&S
     STORE " " TO MTAPPLRT&S, MTDATE&S, MLAYDES&S, MLAYMAT&S, MTWIDAT&S,
MZLAYDA&S, MZLAYDB&S
     SAVE ALL LIKE MT* TO GETSTC&S
     RELEASE ALL LIKE MT*
     X = X + 1
  ENDDO
  STORE 0 TO MZCOUNT
endif
MCOUNT = MCOUNT - 1
close DATABASES
set path to

**********************
* get the subgrade file data
**********************
use \PAVEDB\FILES\subgrade index \PAVEDB\INDEXES\subgndx
seek str(msid_no,4)
if found()
   MZplast  = str(plastix,4,1)
   MZliqlim = str(liqlim,4,1)
   MZtxtrl  = str(txtriaxl,3,1)
   MZPRPASS = STR(PPSV200,4,1)
   MZPERM   = STR(PERMIX,5,2)
ELSE
   MZplast  = "    "
   MZliqlim = "    "
   MZtxtrl  = "   "
   MZPRPASS = "    "
   MZPERM   = "     "
endif

close all

* save all the variables to file
save all like MZ* to getstruc
RELEASE ALL LIKE MZ*
return
```

```
*
* SUBSYSTEM:      PRINT SUMMARY REPORT
* PROGRAM NAME:   PRTREPT2.PRG        02/04/88
* PROJECT 2456 - TEXAS FLEXIBLE PAVEMENT DATABASE CONVERSION
* TAMU/TTI
* AUTHOR:         TREVOR X. PEREIRA
* PURPOSE:        TO PRINT OUT THE INVENTORY UPDATE REPORT USING THE DATA
STORED
*                 IN MEMORY VARIABLE FILES.
*
CLEAR
@ 10, 5  SAY "Printing report One (Inventory Update Sheet) for "+str(msid_no)
ON ERROR DO \PAVEDB\REPORTS\ERR_PRNT
SET PRINT TO &MPORT2
ON ERROR DO \PAVEDB\ERROR
&DEVICEON
@ 0, 0  Say "^[&l1O"
mdate = dtoc(date())
SET ALTE ON
@ 0, 145 say "Date: " + mdate
@ 1, 0 say " _____________________________________________________________
_____________________________________________________________________________-
_____________________"
@ 2, 0 say "|LOCATION                                                        |
|GEOMETRIC AND SHOULDER INFORMATION
                          |"
************
* PRINT LOCATION SECTION
************
RESTORE FROM GETLOCT2 ADDITIVE
@ 3, 0   SAY "|    SECTION ID NO:     "+LTRIM(STR(MSID_NO,4))
@ 3, 57  SAY "| |                TYPE              OUTSIDE     SHOULDER
SHOULDER   SHOULDER    SHOULDER"
@ 3, 161 SAY "|"
@ 4, 0   SAY "|    DISTRICT NO:       "+MLHWYDST
@ 4, 57  SAY "| | STRUCTURE      OF      LANE     SHOULDER   SURFACE      BASE
   SURFACE        BASE      WIDENING"
@ 4, 161 SAY "|"
RELEASE MLHWYDST
@ 5, 0   SAY "|    COUNTY NO/NAME:    "+MLCNTYNM+"/"+MLCNTYNAM
@ 5, 57  SAY "| |  NUMBER     PAVEMENT   WIDTH    WIDTH       TYPE       TYPE
   THICKNESS   THICKNESS     FLAG"
@ 5, 161 SAY "|"
@ 6, 0   SAY "|    CONTROL-SECTION:  "+MLCONTISEC
@ 6, 57  SAY "| |  ~~~~~~~~~   ~~~~~~~~   ~~~~~   ~~~~~~~~   ~~~~~~~~
~~~~~~~~   ~~~~~~~~~   ~~~~~~~~~   ~~~~~~~~"
@ 6, 161 SAY "|"
@ 7, 0   SAY "|    HIGHWAY:           "+MLHWY
RESTORE FROM GETGEOSH ADDITIVE
@ 7, 57  SAY "| |    "+MGSTRUC0+"          "+MGPAVE0+"       "+MGLANEWD0+"
"+MGOUTSHD0+"         "+MGSHDSUR0+"         "+MGSHDBSE0+"
```

```
"+MGSHDSFT0+"        "+MGSHDBST0+"          "+MGWIDEN0
@ 7, 161 SAY "|"
@ 8, 0   SAY "|    MILE POINTS:       "+MLBEGMPN+"-"+MLENDMPN
@ 8, 57  SAY "| |    "+MGSTRUC1+"          "+MGPAVE1+"       "+MGLANEWD1+"
"+MGOUTSHD1+"          "+MGSHDSUR1+"          "+MGSHDBSE1+"
"+MGSHDSFT1+"        "+MGSHDBST1+"          "+MGWIDEN1
@ 8, 161 SAY "|"

******************
* PRINT GEOMETRIC & SHOULDER SECTION
******************
@ 9, 0   SAY "|    LANE:               "+MLLANEID
@ 9, 57  SAY "| |    "+MGSTRUC2+"          "+MGPAVE2+"       "+MGLANEWD2+"
"+MGOUTSHD2+"          "+MGSHDSUR2+"          "+MGSHDBSE2+"
"+MGSHDSFT2+"        "+MGSHDBST2+"          "+MGWIDEN2
@ 9, 161 SAY "|"
@ 10, 0   SAY "|    MILE POST:          "+MLPOST
@ 10, 57  SAY "| |    "+MGSTRUC3+"          "+MGPAVE3+"       "+MGLANEWD3+"
 "+MGOUTSHD3+"          "+MGSHDSUR3+"          "+MGSHDBSE3+"
"+MGSHDSFT3+"        "+MGSHDBST3+"          "+MGWIDEN3
@ 10, 161 SAY "|"
@ 11, 0   SAY "|    PREVIOUS SID:      "+MLPREVSD
@ 11, 57  SAY "| |    "+MGSTRUC4+"          "+MGPAVE4+"       "+MGLANEWD4+"
 "+MGOUTSHD4+"          "+MGSHDSUR4+"          "+MGSHDBSE4+"
"+MGSHDSFT4+"        "+MGSHDBST4+"          "+MGWIDEN4
@ 11, 161 SAY "|"
@ 12, 0   SAY "|    NEXT SID:          "+MLNEXTSD
@ 12, 57  SAY "| |    "+MGSTRUC5+"          "+MGPAVE5+"       "+MGLANEWD5+"
 "+MGOUTSHD5+"          "+MGSHDSUR5+"          "+MGSHDBSE5+"
"+MGSHDSFT5+"        "+MGSHDBST5+"          "+MGWIDEN5
@ 12, 161 SAY "|"
@ 13, 0   SAY "|    FUNCTIONAL CLASS: "+MLFUNCLS
@ 13, 57  SAY "| |    "+MGSTRUC6+"          "+MGPAVE6+"       "+MGLANEWD6+"
 "+MGOUTSHD6+"          "+MGSHDSUR6+"          "+MGSHDBSE6+"
"+MGSHDSFT6+"        "+MGSHDBST6+"          "+MGWIDEN6
@ 13, 161 SAY "|"
@ 14, 0   SAY "|    TYPE OF PAVEMENT: "+MLBASETYP
@ 14, 57  SAY "| |    "+MGSTRUC7+"          "+MGPAVE7+"       "+MGLANEWD7+"
 "+MGOUTSHD7+"          "+MGSHDSUR7+"          "+MGSHDBSE7+"
"+MGSHDSFT7+"        "+MGSHDBST7+"          "+MGWIDEN7
@ 14, 161 SAY "|"
@ 15, 0   SAY "|                          "+MLBASETHK
@ 15, 57   SAY "| |    "+MGSTRUC8+"          "+MGPAVE8+"       "+MGLANEWD8+"
  "+MGOUTSHD8+"          "+MGSHDSUR8+"          "+MGSHDBSE8+"
"+MGSHDSFT8+"        "+MGSHDBST8+"         "+MGWIDEN8
@ 15, 161 SAY "|"
@ 16, 0   SAY "|"
@ 16, 22  SAY "    "+MLBASESEAL
@ 16, 57  SAY "| |    "+MGSTRUC9+"          "+MGPAVE9+"       "+MGLANEWD9+"
 "+MGOUTSHD9+"          "+MGSHDSUR9+"          "+MGSHDBSE9+"
"+MGSHDSFT9+"        "+MGSHDBST9+"          "+MGWIDEN9
@ 16, 161 SAY "|"
@ 17, 0   SAY "|"
@ 17, 22  SAY MLWIDEN
```

```
@ 17, 57  SAY "| |     "+MGSTRUC10+"           "+MGPAVE10+"       "+MGLANEWD10+"
   "+MGOUTSHD10+"          "+MGSHDSUR10+"          "+MGSHDBSE10+"
"+MGSHDSFT10+"       "+MGSHDBST10+"          "+MGWIDEN10
@ 17, 161 SAY "|"
@ 18, 0   SAY "|    INACTIVE SID:      "+MLINACTV
@ 18,  0  SAY "|"
@ 18, 57  SAY "| |     "+MGSTRUC11+"           "+MGPAVE11+"       "+MGLANEWD11+"
   "+MGOUTSHD11+"          "+MGSHDSUR11+"          "+MGSHDBSE11+"
"+MGSHDSFT11+"       "+MGSHDBST11+"          "+MGWIDEN11
@ 18, 161 SAY "|"
@ 19,  0  SAY "|    NUMBER OF LANES:  "+MLNUMLAN
@ 19, 57  SAY "| |     "+MGSTRUC12+"           "+MGPAVE12+"       "+MGLANEWD12+"
   "+MGOUTSHD12+"          "+MGSHDSUR12+"          "+MGSHDBSE12+"
"+MGSHDSFT12+"       "+MGSHDBST12+"          "+MGWIDEN12
@ 19, 161 SAY "|"
@ 20,  0  SAY "|"
@ 20, 57  SAY "| |     "+MGSTRUC13+"           "+MGPAVE13+"       "+MGLANEWD13+"
   "+MGOUTSHD13+"          "+MGSHDSUR13+"          "+MGSHDBSE13+"
"+MGSHDSFT13+"       "+MGSHDBST13+"          "+MGWIDEN13
@ 20, 161 SAY "|"
RELEASE ALL LIKE ML*
@ 21,  0  SAY "|"
@ 21, 57  SAY "| |     "+MGSTRUC14+"           "+MGPAVE14+"       "+MGLANEWD14+"
   "+MGOUTSHD14+"          "+MGSHDSUR14+"          "+MGSHDBSE14+"
"+MGSHDSFT14+"       "+MGSHDBST14+"          "+MGWIDEN14
@ 21, 161 SAY "|"
@ 22,  0  SAY "|"
@ 22, 57  SAY "| |     "+MGSTRUC15+"           "+MGPAVE15+"       "+MGLANEWD15+"
   "+MGOUTSHD15+"          "+MGSHDSUR15+"          "+MGSHDBSE15+"
"+MGSHDSFT15+"       "+MGSHDBST15+"          "+MGWIDEN15
@ 22, 161 SAY "|"
RELEASE ALL LIKE MG*
@ 23, 0   SAY "|_____________________________________________________________|
|_____________________________________________________________________________
_____________________|"
@ 24, 0 SAY
"_____________________________________________________________________________
_____________________________________________________________________________
_____________________"
@ 25, 0   SAY "|STRUCTURAL SECTION              THICK
    WIDENING AGG.  ===ADMIXTURE===  APPL % PASSING                    PERM."
@ 25, 161 SAY "|"
@ 26,  0  SAY "|  STRUCTURE LAYER DESCRIPTION CENT     MATERIAL TYPE
DATE     DATE   RATE  TYPE          PCNT  RATE 200 SIEVE TTC   LL    PI
INDEX"
@ 26, 161 SAY "|"
@ 27,  0  SAY "|  ~~~~~~~~~ ~~~~~ ~~~~~~~~~~ ~~~~~  ~~~~~~~~~~~~~~~~~~~
~~~~~  ~~~~~~~~ ~~~~  ~~~~~~~~~~~ ~~~~  ~~~~ ~~~~~~~~~ ~~~  ~~~~  ~~~~
~~~~~"
@ 27, 161 SAY "|"

***********
* PRINT STRUCTURAL SECTION
***********
```

```
RESTORE FROM GETSTRUC ADDITIVE
X = 1
MROW = 28
DO WHILE X < 14
     IF X < 10
        S = STR(X,1)
     ELSE
        S = STR(X,2)
     ENDIF
     RESTORE FROM GETSTC&S ADDI
     @ MROW,  0  SAY "|"
     @ MROW,  6  SAY MISTRUC&S
     @ MROW, 14  SAY MILAY&S
     @ MROW, 21  SAY MZLAYDA&S
     @ MROW, 24  SAY MZLAYDB&S
     @ MROW, 31  SAY MICENTHK&S
     @ MROW, 38  SAY MZMATTYP&S
     @ MROW, 60  SAY MIDATE&S
     @ MROW, 70  SAY MIWIDAT&S
     @ MROW, 76  SAY MIAGGRAT&S
     @ MROW, 94  SAY MIADMXPR&S
     @ MROW, 101 SAY MIAPPLRT&S
     IF MZCOUNT = X
          @ MROW, 108  SAY MZPRPASS
          @ MROW, 116  SAY MZTXTRL
          @ MROW, 121  SAY MZLIQLIM
          @ MROW, 127  SAY MZPLAST
          @ MROW, 132  SAY MZPERM
     ENDIF
     RELE ALL LIKE MI*
     MFILE = "GETSTC" + S + ".MEM"
     DELE FILE &MFILE
     @ MROW, 161 SAY "|"
     X = X + 1
     MROW = MROW + 1
ENDDO
RELE ALL LIKE MZ*
X = 1
MROW = 41
DO WHILE X < 4
     IF X < 10
        S = STR(X,1)
     ELSE
        S = STR(X,2)
     ENDIF
     @ MROW,  0  SAY "|"
     @ MROW, 161 SAY "|"
     X = X + 1
     MROW = MROW + 1
ENDDO
RELEASE ALL LIKE MI*
@ 44, 0   SAY
"|_____________________________________________________-
_____________________________________________________-
```

```
_____|"
* AT END DELETE ALL MEM FILES
SET ALTE OFF
SET PRINT OFF
&DEVICEOFF
DELETE FILE GETGEOSH.MEM
DELETE FILE GETSTRUC.MEM
RETURN
```

```
*
* SUBSYSTEM:      PRINT SUMMARY REPORT
* PROGRAM NAME:   PRTREPT3.PRG        02/04/88
* PROJECT 2456 - TEXAS FLEXIBLE PAVEMENT DATABASE CONVERSION
* TAMU/TTI
* AUTHOR:         TREVOR X. PEREIRA
* PURPOSE:        TO PRINT OUT THE SECOND PAGE OF THE INVENTORY UPDATE REPORT
*

@ 15, 10  SAY "Printing report Two (Blank) ...."
ON ERROR DO \PAVEDB\REPORTS\ERR_PRNT
SET PRINT TO &MPORT2
ON ERROR DO \PAVEDB\ERROR
&DEVICEON
@ 0, 0  Say "^[&l1O"
SET ALTE ON
mdate = dtoc(date())
@ 0, 145 say "Date: " + mdate
@ 1, 0   say " _____________________________________________________
_____________________________________________________________________-
_____________________"
@ 2, 0   say "|LOCATION                                                  |
|GEOMEITRIC AND SHOULDER INFORMATION
                        |"

************
* PRINT LOCATION SECTION
************
RESTORE FROM GETLOCT2 ADDITIVE
@ 3, 0   SAY "|   SECTION ID NO:    "+LTRIM(STR(MSID_NO,4))
@ 3, 57  SAY "| |                TYPE              OUTSIDE    SHOULDER
SHOULDER   SHOULDER    SHOULDER"
@ 3, 161 SAY "|"
@ 4, 0   SAY "|   DISTRICT NO:      "+MLHWYDST
@ 4, 57  SAY "| | STRUCTURE      OF      LANE    SHOULDER   SURFACE      BASE
   SURFACE       BASE      WIDENING"
@ 4, 161 SAY "|"
RELEASE MLHWYDST
@ 5, 0   SAY "|   COUNTY NO/NAME:   "+MLCNTYNM+"/"+MLCNTYNAM
@ 5, 57  SAY "| | NUMBER     PAVEMENT   WIDTH    WIDTH       TYPE       TYPE
   THICKNESS   THICKNESS     FLAG"
@ 5, 161 SAY "|"
@ 6, 0   SAY "|   CONTROL-SECTION:  "+MLCONTSEC
@ 6, 57  SAY "| | ~~~~~~~~~  ~~~~~~~~  ~~~~~  ~~~~~~~~  ~~~~~~~~
~~~~~~~  ~~~~~~~~  ~~~~~~~~  ~~~~~~~~"
@ 6, 161 SAY "|"
@ 7, 0   SAY "|   HIGHWAY:          "+MLHWY
@ 7, 57  SAY "| | _______  _____  _____  _____  _____
_____  _____  _____  _____"
@ 7, 161 SAY "|"
@ 8, 0   SAY "|   MILE POINTS:      "+MLBEGMPN+"-"+MLENDMPN
```

```
@ 8, 57  SAY "|  |"
@ 8, 161 SAY "|"

******************
* PRINT GEOMETRIC & SHOULDER SECTION
******************
@ 9, 0   SAY "|    LANE:              "+MLLANEID
@ 9, 57  SAY "|  | _______    _______  _____    _____     _____
_____ _____ |"
@ 9, 161 SAY "|"
@ 10, 0   SAY "|    MILE POST:         "+MLPOST
@ 10, 57  SAY "|  |"
@ 10, 161 SAY "|"
@ 11, 0   SAY "|    PREVIOUS SID:      "+MLPREVSD
@ 11, 57  SAY "|  | _______    _______  _____    _____     _____
_____ _____ |"
@ 11, 161 SAY "|"
@ 12, 0   SAY "|    NEXT SID:          "+MLNEXTSD
@ 12, 57  SAY "|  |"
@ 12, 161 SAY "|"
@ 13, 0   SAY "|    FUNCTIONAL CLASS: "+MLFUNCLS
@ 13, 57  SAY "|  | _______    _______  _____    _____     _____
_____ _____ |"
@ 13, 161 SAY "|"
@ 14, 0   SAY "|    TYPE OF PAVEMENT: "+MLBASETYP
@ 14, 57  SAY "|  |_____-
_____|"
@ 15, 0   SAY "|                       "+MLBASETHK
@ 15, 57  SAY "|
_____-
                    "
_____
@ 16, 0   SAY "|"
@ 16, 22  SAY "    "+MLBASESEAL
RESTORE FROM GETDISTN ADDITIVE
@ 16, 57  SAY "|  | LAYER THICKNESS ACROSS THE ROAD"
@ 16, 132 SAY "CURRENT SURFACE WIDTH " + MDTHICK
@ 16, 161 SAY "|"
@ 17, 0   SAY "|"
@ 17, 22  SAY MLWIDEN
@ 17, 57  SAY "|  |                          THICKNESS FROM CENTER
THICKNESS       DISTANCE FROM CENTER"
@ 17, 161 SAY "|"
@ 18, 0   SAY "|    INACTIVE SID:      "+MLINACTV
@ 18,  0  SAY "|"
@ 18, 57  SAY "|  | STRUCTURE   LAYER    ----------------------        AT
      ------------------------"
@ 18, 161 SAY "|"
@ 19,  0  SAY "|    NUMBER OF LANES:  "+MLNUMLAN
@ 19, 57  SAY "|  |  NUMBER     NUMBER   3rd POS  2nd POS  1st POS      CENTER
     3rd POS  2nd POS  1st POS"
@ 19, 161 SAY "|"
@ 20,  0  SAY "|"
@ 20, 57  SAY "|  | ~~~~~~~~~   ~~~~~~   ~~~~~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~  ~~~~~~~~~~~~~~~~~~~~~~~~"
```

```
@ 20, 161 SAY "|"
RELEASE ALL LIKE M1*
@ 21,  0  SAY "|"
@ 21, 57  SAY "|  |  _____      ____  "  _______  _______  _______
_____  _____  _____  _____"
@ 21, 161 SAY "|"
@ 22,  0  SAY "|"
@ 22, 57  SAY "|  |"
@ 22, 161 SAY "|"
@ 23,  0  SAY "|"
@ 23, 57  SAY "|  |  _____      ____  "  _______  _______  _______
_____  _____  _____  _____"
@ 23, 161 SAY "|"
@ 24,  0  SAY "|"
@ 24, 57  SAY "|  |"
@ 24, 161 SAY "|"
@ 25,  0  SAY "|"
@ 25, 57  SAY "|  |  _____      ____  "  _______  _______  _______
_____  _____  _____  _____"
@ 25, 161 SAY "|"
@ 26,  0  SAY "|"
@ 26, 57  SAY "|  |"
@ 26, 161 SAY "|"
@ 27,  0  SAY "|"
@ 27, 57  SAY "|  |  _____      ____  "  _______  _______  _______
_____  _____  _____  _____"
@ 27, 161 SAY "|"
@ 28, 0   SAY "|_____________________________________________________________
|_____________________________________________-
_____________________________________|"
@ 29, 0   SAY " _____________________________________________________-
_____________________________________________________-
_____"
@ 30, 0   SAY "|STRUCTURAL SECTION            THICK
WIDENING  AGG.   ===ADMIXTURE===  APPL  % PASSING
PERM."
@ 30, 161 SAY "|"
@ 31,  0  SAY "|  STRUCTURE LAYER DESCRIPTION CENT   MATERIAL TYPE  DATE
DATE    RATE   TYPE          PCNT  RATE  200 SIEVE  TTC     LL      PI
INDEX"
@ 31, 161 SAY "|"
@ 32,  0  SAY "|  ~~~~~~~~~ ~~~~~ ~~~~~~~~~~ ~~~~~  ~~~~~~~~~~~~  ~~~~~
~~~~~~~~ ~~~~~~  ~~~~~~~~~~~~ ~~~~  ~~~~  ~~~~~~~~~  ~~~   ~~~~~~   ~~~~
~~~~~"
@ 32, 161 SAY "|"

***********
* PRINT STRUCTURAL SECTION
***********
X = 1
MROW = 33
DO WHILE X < 12
     IF X < 10
        S = STR(X,1)
```

```
      ELSE
         S = STR(X,2)
      ENDIF
      @ MROW,  0  SAY "|"
      @ MROW,  3  SAY "_______"          && STRUCTURE NUMBER
      @ MROW, 13  SAY "_____"            && LAYER NUMBER
      @ MROW, 21  SAY "________"         && DESCRIPTION
      @ MROW, 31  SAY "_______"          && CENTER THICKNESS
      @ MROW, 40  SAY "__________"       && MATERIAL TYPE
      @ MROW, 53  SAY "_____"            && DATE
      @ MROW, 61  SAY "_______"          && WIDEN DATE
      @ MROW, 69  SAY "_______"          && AGGREGATE RATE
      @ MROW, 77  SAY "__________"       && ADMIXTURE TYPE
      @ MROW, 88  SAY "_______"          && ADMIXTURE PERCENT
      @ MROW, 95  SAY "_______"          && APPLICATION RATE
      @ MROW, 103 SAY "________"         && % PASSING SIEVE
      @ MROW, 112 SAY "_____"            && TEXAS TRIAXLE CLASS
      @ MROW, 119 SAY "_______"          && LIQUID LIMIT
      @ MROW, 127 SAY "_______"          && PLASTICITY INDEX
      @ MROW, 135 SAY "_____"            && PERMIABILITY INDEX
      @ MROW, 161 SAY "|"
      X = X + 2
      MROW = MROW + 1
      @ MROW,  0  SAY "|"
      @ MROW, 161 SAY "|"
      MROW = MROW + 1
ENDDO

@ 44, 0   SAY "|_____________________________________________________-
_____________________________________________________________________-
_____________________________________|"

* AT END DELETE ALL MEM FILES
SET ALTE OFF
CLEAR
SET PRINT OFF
&DEVICEOFF
DELETE FILE GETLOCT2.MEM
DELETE FILE GETDISIN.MEM

@ 15, 20 SAY "DONE ....."
RETURN
```

CHAPTER 4

EDIT AND UPDATE SUBSYSTEM

GENERAL NARRATIVE

The Edit & Update Menu System allows the user to edit or update the data in the Master files and the 7 tables. This portion is divided into 4 sections. The sections are:

- Edit and Update Driver
- Monitoring Data Update
- Inventory Data Update
- Traffic Date Update.

In addition to driving the edit and update process, EDITUPDT.PRG also updates the table files. The user must be an experienced dBASE programmer because the master tables are modified directly (without edit checks) using the dBASE BROWSE command. The table files can be found in the subdirectory \PAVEDB\FILES while the program EDITUPDT.PRG is in \PAVEDB\EDITUPDT.

The PES data is copied from the Original PES Tape to a disk file by a stand alone batch file (i.e. the batch file is not included in the Flexible Pavement Menu System). The batch file (READPES.BAT) is documented in Section 2 of this Chapter. This data is then converted to 6 comma delimited data files by the program PESMOD.PAS. These delimited files are further converted to dBASE files by PESUP.PRG and are used to update the monitoring files (Skid, Visual, Serviceability Index and Falling Weight). The PES programs can be found in the subdirectory \PAVEDB\EDITUPDT\PES.

Updating the Inventory files is divided into two parts - adding inventory data and changing inventory data. The data is added manually by the user through the program ENTRLAYR.PRG while the data is modified through CHNGLAYR.PRG. The Inventory files modified are Location, Layer Identification, Layer Thickness Across The Road, Geometric and Shoulder Information, Surface and Subgrade. The Add and Change programs can be found in the subdirectory \PAVEDB\EDITUPDT.

The traffic data is copied from the Roadway Inventory (RIFILE) Tape File to a disk file by a stand alone batch program (READTRAF.BAT) The batch file is documented in section 4 of this Chapter. This data is then converted to a comma delimited format to be added to a dBASE File. This dBASE file is then used to update the Traffic File. Other than copying the tape file to disk, TRAFUPD.PRG controls the traffice update process. The traffic programs can be found in the subdirectory \PAVEDB\EDITUPDT\TRAFFIC.

Each Sub-section has the following information:

- Program Narrative
- Program Flow Diagram
- Program Specification
- Menu Screens
- Sample Reports
- Program Listings

Section 1: Edit and Update Driver

Edit and Update Driver
Program Narrative

The Edit & Update program (EDITUPDT.PRG) produces the main menu screen for the Edit & Update Subsystem. It also produces the menu screen to edit or change the tables. The screens generated are displayed in this section.

The Edit & Update program also calls the programs PESMOD.PAS and PESUP.PRG to add PES data to the monitoring files. To add data to the Traffic file, the program TRAFUPD.PRG is called which obtains traffic data from the Roadway Inventory File. EDITUPDT.PRG also calls the programs to update the data in the Inventory Files. To add inventory data, ENTRLAYR.PRG is called and to change inventory data, CHNGLAYR.PRG is called. To update the tables, EDITUPDT.PRG uses the browse command to let the user change or add data.

Edit and Update Subsystem
Program Flow Diagram

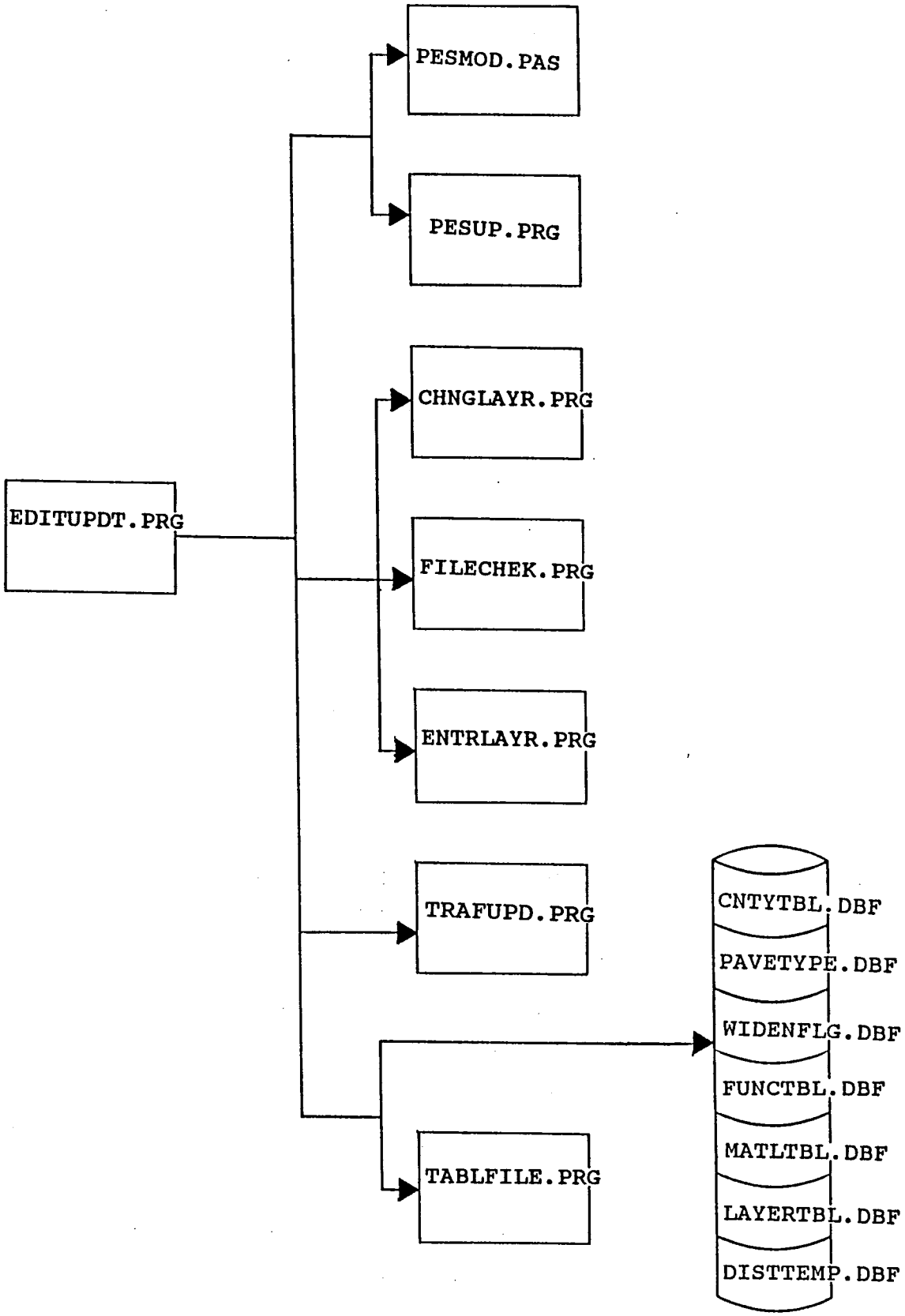
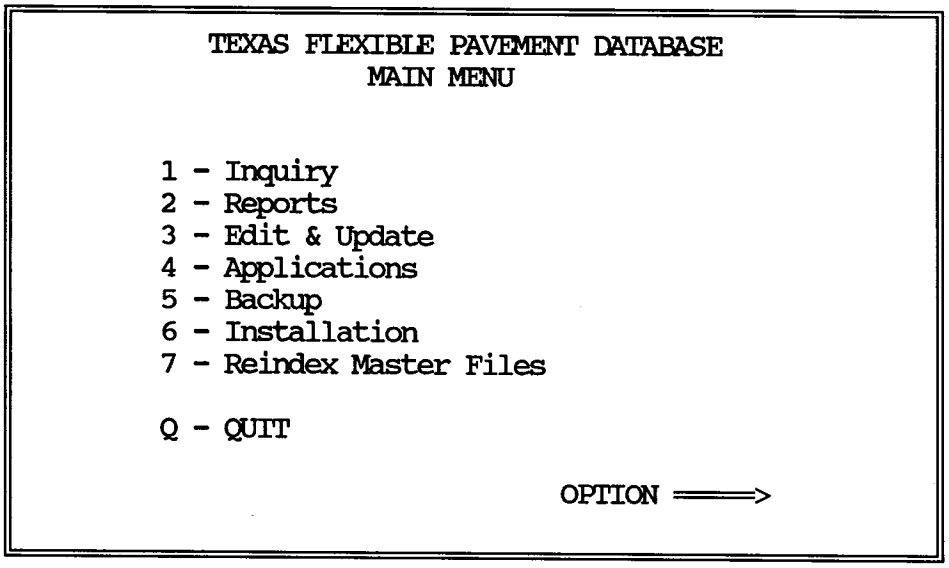


FIGURE 10

EDIT AND UPDATE SCREENS

The following screens are produced by EDITUPDT.PRG except for the Main Menu.



Above is the Main Menu of the Texas Flexible Pavement Database System. When the Edit & Update option is chosen, then screen 3.0 is displayed.

EDIT AND UPDATE
MENU SCREENS (cont.)

```
        TEXAS FLEXIBLE PAVEMENT DATABASE          3.0
                  Edit & Update


        1 - Pavement Condition Data
        2 - Inventory Data
        3 - Traffic Data
        4 - Tables

                                  OPTION ===>
```

Choice 1 — Pavement Condition Data runs the PESMOD.PAS and PESUP.PRG programs which display their own screens and are shown in Section 2 of this chapter.

Choice 2 — Inventory Data runs the ENTRLAYR.PRG program if ADD is desired or CHNGLAYR.PRG if CHANGE is desired. These programs have their own screens which are displayed in Section 3 of this chapter.

Choice 3 — Traffic Data runs the TRAFUPD.PRG program which has its own screens and are shown in Section 4 of this chapter.

Choice 4 — Tables displays screen 3.5 and executes the dBASE BROWSE command for the requested option.

```
        TEXAS FLEXIBLE PAVEMENT DATABASE          3.5
                  EDIT & UPDATE
                     Tables

        1 - County Name
        2 - Material Type
        3 - Type of Pavement
        4 - District Temperature Constant
        5 - Widening Flag
        6 - Layer Description
        7 - Functional Classification

                                  OPTION ===>
```

WARNING ! ! ! - These files should only be modified by an experienced dBASE user. To abandon process, press the Esc key. Otherwise, enter option.

PROGRAM SPECIFICATION

Program Name: EDITUPDT.PRG

Purpose: To update the data in the master files and tables.

Procedures\Edits:

Refer to the appropriate sub-section for each program.

1) Convert PES data from tape to comma delimited ASCII files using the program PESMOD.PAS.
2) Add PES data to the Monitoring Files using the program PESUP.PRG.
3) Add data to the Inventory Files using the program ENTRLAYR.PRG.
4) Change data in the Inventory Files using the program CHNGLAYR.PRG.
5) Update the Traffic File with data from the Roadway Inventory File using the program TRAFUPD.PRG.
6) Add or Change the data in the Table files using the program EDITUPDT.PRG.

Input\Output Files:

PES Data -	refer to the PESMOD.PAS sub-section for PES Modification files and the PESUP.PRG sub-section for PES Update files.
Inventory Data -	refer to the CHNGLAYR.PRG sub-section for change files and the ENTRLAYR.PRG sub-section for add files.
Traffic Data -	Refer to the TRAFUPD.PRG sub-section for Traffic files.
Tables -	County Table (CNTYTBL.DBF) Material Type Table (MATLTBL.DBF) Pavement Type Table (PAVETYPE.DBF) District Temperature (DISTTEMP.DBF) Widening Table (WIDENFLG.DBF) Layer Description (LAYERTBL.DBF) Functional Classification (FUNCLTBL.DBF)

Programs Called (See Program Flow Diagram):

PES - PESMOD.PAS
- NOPESSID.PRG
- NO_LOCN.PRG
- PESUP.PRG
- PES_MRM.PRG
- PES_SKID.PRG
- PES_SSI.PRG
- PES_VISL.PRG

INVENTORY - FILECHEK.PRG
Add - ENTRLAYR.PRG
- ADDLAYR.PRG
- INV_BKUP.PRG
- INV_UPDT.PRG
- LOCNCHEK.PRG
- LAYRCHEK.PRG
- LAYTCHEK.PRG
- GEOSCHEK.PRG
- SURFCHEK.PRG
- SUBGCHEK.PRG

Change - CHNGLAYR.PRG
- CHNGBKUP.PRG
- CHEKLOCN.PRG
- CHEKLAYR.PRG
- CHEKLAYT.PRG
- CHEKGEOS.PRG
- CHEKSURF.PRG
- CHEKSUBG.PRG
- COPYLAYR.PRG

TRAFFIC - TRAFUPD.PRG
- LOGTRAF.PRG
- NEWTRAF.PRG
- SIDTLOG.PRG
- STLOG.PAS

TABLES - TABLFILE.PRG

```
*
* SUBSYSTEM:        EDIT & UPDATE MAIN MENU
* PROGRAM NAME:     EDITUPDT.PRG              06/03/88
* MODIFIED ON:      10/27/88
* PROJECT 2456 -    TEXAS FLEXIBLE PAVEMENT DATABASE CONVERSION
* TAMU/TTI
* AUTHOR:           TREVOR X. PEREIRA
* PURPOSE:          TO EDIT AND UPDATE DATA FOR THE FOLLOWING:
*                         - Monitoring data (PES)
*                         - Inventory data
*                         - Traffic data
*                         - Create Graph file
*                         - Create Model file
*                         - Tables
*

* set parameters and initialize variables
STORE 0 TO MSID_NO
CLOSE DATABASES
* SET PROCEDURE TO \PAVEDB\EDITUPDT\EDITCOLL
MREPEAT2 = .T.
SET STAT OFF
SET ECHO OFF
SET TALK OFF
SET CONFIRM ON

DO WHILE MREPEAT2
 STORE " " TO EDITPICK
 * set up the EDIT & UPDATE menu screen and do the loop until EDITPICK is 1-4
 DO WHILE .NOT. (EDITPICK $ '1234')
  @  0,  0  CLEAR
  @  4, 20  SAY "TEXAS FLEXIBLE PAVEMENT DATABASE        3.0"
  @  5, 29  SAY "Edit & Update"
  @ 10, 16  SAY "1 - Pavement Condition Data"
  @ 11, 16  SAY "2 - Inventory Data"
  @ 12, 16  SAY "3 - Traffic Data"
  @ 13, 16  SAY "4 - Tables"
  @ 16, 41  SAY "OPTION ======> "  GET EDITPICK
  @  3,  9  TO 18, 65    DOUBLE
  READ
  IF READKEY() = 12
     CLEAR
     CLOSE DATABASE
     SET PROC TO
     MREPEAT2 = .F.
     EXIT
  ENDIF
 ENDDO
 IF MREPEAT2
```

```
  * according to the response received from the EDIT & UPDATE menu, the
appropriate
  * commands are executed
  DO CASE
   * if choice 1 is chosen, run the PES program to add data to the
   * monitoring files
   CASE EDITPICK = "1"
      CLEAR
      MRUN = " "
      @ 5, 5 SAY "This program is going to update the monitoring database"
      @ 6, 5 SAY "  Files.  It will take the approximately 20 HOURS to run."
      @ 8, 5 SAY "To run this program, you need to have the PES Data file "
      @ 9, 5 SAY "  that is obtained from tape in the subdirectory "
      @ 10, 5 SAY "  \PAVEDB\EDITUPDT\PES "
      @ 12, 5 SAY "The data file should be called PES87.DAT for year 1987. "
      @ 13, 5 SAY "  If the file is for year 1988, the data file should be "
      @ 14, 5 SAY "  called PES88.DAT"
      @ 17, 5 SAY "Do you want to continue (Y/N) ? " GET MRUN
      READ
      IF MRUN = "Y"
         RUN \PAVEDB\EDITUPDT\PES\PESMOD
         DO \PAVEDB\EDITUPDT\PES\PESUP
         CLEA TYPE
      ENDIF
      RELEASE MRUN

   * If choice 2 is chosen
   CASE EDITPICK = "2"
     CLEAR
     DO \PAVEDB\EDITUPDT\FILECHEK
     GETREPLY = " "
     DO WHILE .NOT. (GETREPLY $ 'AC')
       @ 10, 15  SAY "Do you want to ADD data to the inventory files or"
       @ 11, 15  SAY "         CHANGE the data in the inventory files."
       @ 14, 15  SAY "    Enter 'A' to ADD, 'C' to CHANGE or Esc to exit"
       @ 16, 15  SAY "          OPTION ====> " GET GETREPLY PICTURE "!"
       READ
       IF READKEY() = 12
          MCONTINUE = .F.
          EDITCOLL  = " "
          GETREPLY = " "
          EXIT
       ENDIF
     ENDDO
     IF GETREPLY = "C"
       DO \PAVEDB\EDITUPDT\CHNGLAYR
     ENDIF
     IF GETREPLY = "A"
       DO \PAVEDB\EDITUPDT\ENTRLAYR
     ENDIF
     CLEAR
     SET PROCEDURE TO
```

```
   * if choice 3 is chosen, run the pascal traffic program
   CASE EDITPICK = "3"
     MYEAR = 0
     * run the program to update traffic file
     @ 15, 30  SAY "DATA YEAR (19XX) ====> " GET MYEAR PICT "9999"
     READ
     * Add the data to the traffic file
     IF FILE('\PAVEDB\TLOG.DAT')
          DO \PAVEDB\EDITUPDT\TRAFFIC\UPDATE
     ELSE
          CLEAR
          @ 10,10 SAY " "
          ? "      Cannot find TLOG.DAT file.  Please check and place the"
          ? "         file in subdirectory \PAVEDB "
          ? " "
          ? " "
          WAIT
     ENDIF
     CLEA TYPE
     CLEAR

   * if choice 4 is chosen, display the tables menu
   CASE EDITPICK = "4"
     MCONTINUE = .T.
     EDITCOLL = " "
     DO \PAVEDB\EDITUPDT\TABLFILE
     DO WHILE MCONTINUE
       CLEAR
       EDITCOLL = " "
       DO WHILE .NOT. (EDITCOLL $ '1234567')
        @  3, 21  SAY "TEXAS FLEXIBLE PAVEMENT DATABASE      3.5"
        @  4, 31  SAY "EDIT & UPDATE"
        @  5, 34  SAY "Tables"
        @  8, 16  SAY "1 - County Name"
        @  9, 16  SAY "2 - Material Type"
        @ 10, 16  SAY "3 - Type of Pavement"
        @ 11, 16  SAY "4 - District Temperature Constant"
        @ 12, 16  SAY "5 - Widening Flag"
        @ 13, 16  SAY "6 - Layer Description"
        @ 14, 16  SAY "7 - Functional Classification"
        @ 17, 42  SAY "OPTION ====> " GET EDITCOLL
        @  2,  9  TO 18, 65    DOUBLE
        * Give warning
        @ 19, 10 SAY "WARNING ! ! !  - These files should only be modified
by"
        @ 20, 10 SAY "        an experienced dBASE user.  To abandon
process,"
        @ 21, 10 SAY "        press the Esc key.  Otherwise, enter option."
        CLEA TYPE
        READ

        IF READKEY() = 12
          MCONTINUE = .F.
```

```
            CLEAR
            EXIT
          ENDIF
        ENDDO
        IF MCONTINUE
         DO CASE
           CASE EDITCOLL = "1"
              USE \PAVEDB\FILES\CNTYTBL INDEX \PAVEDB\INDEXES\CTYTBLNO
              BROWSE
           CASE EDITCOLL = "2"
              USE \PAVEDB\FILES\MATLTBL
              BROWSE
           CASE EDITCOLL = "3"
              USE \PAVEDB\FILES\PAVETYPE
              BROWSE
           CASE EDITCOLL = "4"
              USE \PAVEDB\FILES\DISTTEMP
              BROWSE
           CASE EDITCOLL = "5"
              USE \PAVEDB\FILES\WIDENFLG
              BROWSE
           CASE EDITCOLL = "6"
              USE \PAVEDB\FILES\LAYERTBL
              BROWSE
           CASE EDITCOLL = "7"
              USE \PAVEDB\FILES\FUNCLTBL
              BROWSE
         ENDCASE
        ENDIF
      ENDDO
    ENDCASE
  ENDIF
ENDDO

SET PROCEDURE TO
CLOSE DATABASES
CLEAR
RETURN
```

```
*
* SUBSYSTEM:      EDIT & UPDATE
* PROGRAM NAME:   TABLFILE.PRG                07/03/88
* CALLED FROM:    EDITUPDT.PRG
* PROJECT 2456 - TEXAS FLEXIBLE PAVEMENT DATABASE CONVERSION
* TAMU/TTI
* AUTHOR:         TREVOR X. PEREIRA
* PURPOSE:        CHECKS TO SEE IF THE TABLE FILES ARE PRESENT
*

IF .NOT. FILE('\PAVEDB\FILES\CNTYTBL.DBF')
   ? "COUNTY TABLE FILE (CNTYTBL.DBF) not found.  Please Check . . ."
   WAIT
   RETURN TO MASTER
ENDIF
IF .NOT. FILE('\PAVEDB\INDEXES\CTYTBLNO.NDX')
   ? "COUNTY TABLE INDEX (CTYTBLNO.NDX) not found.  Please Check . . ."
   WAIT
   RETURN TO MASTER
ENDIF
IF .NOT. FILE('\PAVEDB\FILES\DISTTEMP.DBF')
   ? "DISTRICT TEMPERATURE FILE (DISTTEMP.DBF) not found.  Please Check . .
."
   WAIT
   RETURN TO MASTER
ENDIF
IF .NOT. FILE('\PAVEDB\FILES\MATLTBL.DBF')
   ? "MATERIAL TABLE FILE (MATLTBL.DBF) not found.  Please Check . . ."
   WAIT
   RETURN TO MASTER
ENDIF
IF .NOT. FILE('\PAVEDB\FILES\PAVETYPE.DBF')
   ? "PAVEMENT TYPE FILE (PAVETYPE.DBF) not found.  Please Check . . ."
   WAIT
   RETURN TO MASTER
ENDIF
IF .NOT. FILE('\PAVEDB\FILES\WIDENFLG.DBF')
   ? "WIDENING FLAG FILE (WIDENFLG.DBF) not found.  Please Check . . ."
   WAIT
   RETURN TO MASTER
ENDIF
IF .NOT. FILE('\PAVEDB\FILES\LAYERTBL.DBF')
   ? "LAYER TABLE FILE (LAYERTBL.DBF) not found.  Please Check . . ."
   WAIT
   RETURN TO MASTER
ENDIF
IF .NOT. FILE('\PAVEDB\FILES\FUNCLTBL.DBF')
   ? "FUNCTIONAL CLASSIFICATION TABLE FILE (FUNCLTBL.DBF) not found.  "
   ? "Please Check . . ."
```

```
   WAIT
   RETURN TO MASTER
ENDIF
RETURN
```

Section 2: Monitoring Data Update

PROGRAM NARRATIVE

This program is for use with a 6250 BPI Cipher GRC Cache Tape tape drive systems manufactured by Overland Data Incorporated (ODI). It utilize the ODI DEPOT2 program to copy a tape file to a disk file. The DEPOT2 program is documented in Chapter 4 of the ODI manual. PESMOD.PAS uses the disk file as an input file. The diskfile (PES.DAT) is stored in the subdirectory \PAVEDB.

PROGRAM SPECIFICATION

Program Name: READPES.BAT

Purpose: To copy the PES data from a tape to a disk file using the ODI DEPOT2 program.

Input File: Original PES Tape File obtained from the Texas State Department of Highways and Public Transportation. The user can request only those records flagged for the Texas Flexible Pavement Database; thereby, decreasing the amount of disk space needed and decreasing processing time.

Output File: PES.DAT - Original PES Disk File

Programs Called:

DEPOT2
READPES.CMP - command file for DEPOT2

PROGRAM LISTING

```
rem readpes.bat issues the depot2 command which calls
rem the cmd file to read the original pes tape obtained from
rem SDHPT
DEPOT2 /c READPES.CMD
```

PROGRAM LISTING

```
; READPES.CMD, a command file for the ODI program DEPOT2.
; Use as:  DEPOT2 /c READPES.CMD
; Instuctions on making out this command file are in Chapter 4
;    of the ODI manual pages 4-26 to 4-28.
; Change D:\PAVEDB\PES.DAT to the name of your output disk file,
;    if needed.
; Change "/r nn" record length if needed.
; Change "/s nnnn" blocksize if needed.
; "/m 1" skips past the tape label.
; "/tvM" translates from EBCDIC, Verbose explanations, reads till End-
;    Of-File mark at the end of the data.
/n D:\PAVEDB\PES.DAT /r 1389 /s 1389 /m 1 /tvM
```

PES Modification Program Narrative

The Pavement Evaluation System (PES) record contains monitoring information for several lanes within a segment. The PES Data Modification program, which is written in TURBO Pascal splits an original PES record into separate monitoring files which contain a separate record for each visual evaluation lane. The original data comes from the PES file which is copied from a tape using READPES.BAT. The output of this program is six comma-delimited files containing different information (general, visual, maysmeter, skid, structural strength index and score information). The key fields are the same for all the output files. The output data is used to update the LOCATION, VISUAL, SI, SKID AND FALLWGHT dBASE files in the Texas Flexible Pavement System. The general program flow is shown in Figure 11. The programs and comma delimited files are stored in the subdirectory \PAVEDB\EDITUPDT\PES.

PES MODIFICATION PROGRAM FLOW DIAGRAM

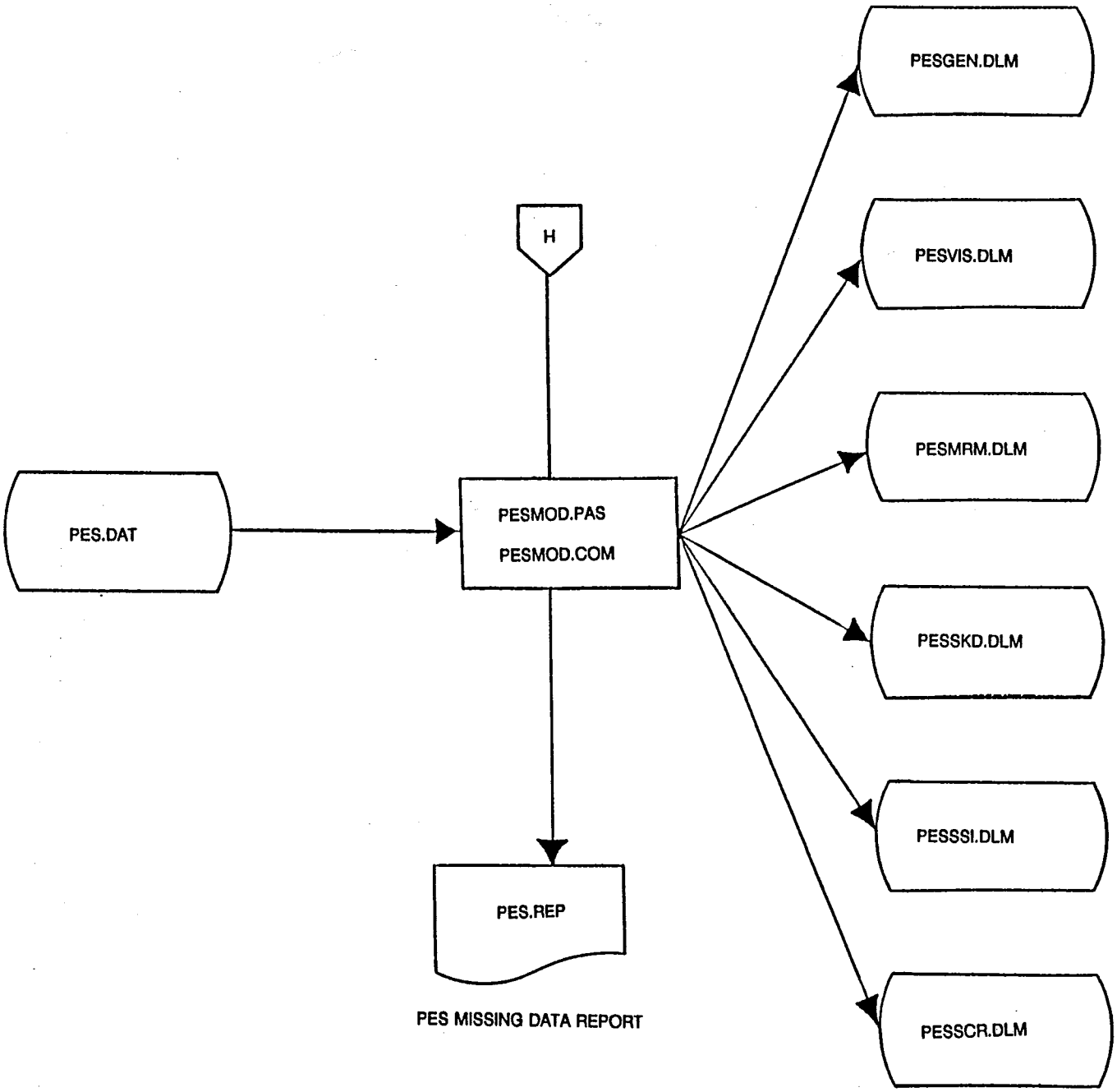


FIGURE 11

PES Modification Program

Program Specification

Program Name: PESMOD.PAS (written in Turbo PASCAL)

Purpose: To split the original PES record into separate records for each Visual Evaluation Lane.

Input File: Original PES File (PES.DAT). The PES Tape File is copied to the PES.DAT disk file using a stand alone routine.

Procedures/Edit:
1. See the Procedure page.
2. See the Original PES File Record Layout and the Modified PES File Record Layouts.

Output Files: Modified PES Files - comma delimited files.
- general information files (PESGEN.DLM)
- visual information file (PESVIS.DLM)
- maysmeter information file (PESMRM.DLM)
- skid information file (PESSKD.DLM)
- structural strength index information file (PESSSI.DLM)
- score information file (PESSCR.DLM)

Report: Missing Data - See the Missing Data Report Layout.

Note: Following the Procedure page, the record layouts are provided in the order they are mentioned above with the Original PES File record layout being first.

Program Specification

PROCEDURE

The original PES record contains general information, visual rating, maysmeter, skid, SSI, and pavement socre information for all roadways within a segment. The PESMOD.PAS program reads the PES file and creates six files each of which contain a separate record for each roadway withing the segemtn. A separate file is created for general information, maysmeter information, visual rating information, skid information, SSI information, and for pavement scores, exclusively. Identifying information for the roadway (also called laneset) is the same in all six files.

The program creates a separate record in each output file for each lane found under original Pes record Item 18 - Visual Evaluation. The lanes for the visual evaluation are in columns 80, 112, 144, 176, 208, and 240 of the original PES record. For the lane set (i.e. R-V, L-P, A-C, etc.), which includes the visual lane being processed, the program selects the appropriate MRM, Skid, SSI, and Score data. For the applicable lane set, the program identifies the high and low MRM reading, calculates the average MRM reading and the standard deviation for the MRM, and stores the total number of MRM readings (maximum of 15). Also it provides the high, the low, the mean, the standard deviation, and the count of observations for the skid data for the appropriate lane set. The program also determines if there is visual evaluation data without corresponding MRM, SSI, or Score data and vice versa. Records which have missing data are reported. Only records with an 'A' in the Mandatory Sample Selection Flag field (Item 25A) and an 'E', 'F', 'G', 'H', 'I', 'N', 'O', 'P', 'Q' or 'R' in the Other Sample Selection Flag field (Item 25B) processed.

ORIGINAL PES FILE RECORD LAYOUT

ITEM NO.	FROM THRU	SIZE	DEC. POS.	FIELD TYPE	ITEM NAME	DESCRIPTION
1	01-02	2		N	DISTRICT NO.	+ VALUES 01-21 & 23-25.
2	03-04	2		N	MAINTENANCE SECTION NO.	+ ALIAS FOREMAN NUMBER.
3	05-07	3		N	COUNTY NO.	+ VALUES 001-254.
4.A	08-09	2		A	HIGHWAY - SYSTEM	+ ALIAS HIGHWAY PREFIX. VALUES IH,US,SH,FM,PR (RE,RM,RR=FM).
.B	10-13	4		N	- NUMBER	+ EXCEPT.. OSR=SH00SR , NASA1=SHNASA .
.C	14	1		A	- SUFFIX	+ VALUES S,L,A,N,E,W, BLANK. FOR SYS=PR, BLANK AND A,B,...,Z.
5.A	15-17	3		N	SEGMENT BEGIN - MILEPOST NO.	+ FOR NON-IH, EVEN NOS. (0,2,4,...) ONLY. FOR IH, ODD & EVEN.
.B	18	1		A	- DISPLACEMENT SIGN	+ VALUES + AND -. ALWAYS + WHEN DISPLACEMENT VALUE = 0.0.
.C	19-20	2	1	N	- DISPLACEMENT VALUE	+ ALIAS DISTANCE-FROM-POST. VALUES 0.0-3.0.
6.A	21-23	3		N	SEGMENT END - MILEPOST NO.	+ SEE ITEM 5 DESCRIPTION.
.B	24	1		A	- DISPLACEMENT SIGN	
.C	25-26	2	1	N	- DISPLACEMENT VALUE	
7.A.1	27-29	3		A/N	INTERMEDIATE POST - 1ST POST NO.	+ ESPECIALLY ON IH, SEGMENTATION CAN REMOVE POSTS FROM VIEW
.2	30-32	3		A/N	- 2ND POST NO.	(IH SEG. FROM POST 110+0.0 TO 112+0.2 COVERS-UP POST 111).
.B.1	33-34	2	1	A/N	- DISTANCE TO 1ST	MAX. OF 2 COVERUPS. BLANK WHEN NO COVERUP EXISTS. DISTANCE
.2	35-36	2	1	A/N	- DISTANCE TO 2ND	IS THAT MILEAGE FROM SEGMENT BEGIN TO THE COVERED-UP POST.
8.A	37-40	4	1	N	ROUTE-THRU-COUNTY - BEGIN VALUE	+ DISTANCE FROM BEGINNING OF ROUTE IN THIS COUNTY TO THE
.B	41-44	4	1	N	- END VALUE	SEGMENT BEGIN OR END (ITEMS 5 OR 6) POSITION RESPECTIVELY.
9	45-46	2	1	N	SEGMENT LENGTH	+ CENTERLINE MILEAGE FROM SEG. BEGIN TO END. VALUES 0.1-3.0.
10	47-52	6		N	LAST UPDATE DATE	+ DATE THIS RECORD (SEGMENT) LAST CHANGED. FORMAT = YYMMDD.
11	53-61	9	2	N	SEGMENT MAINTENANCE COST	+ FOR PES RELEASE 1.0, ALWAYS 0.00.
12	62-67	6		N	AVERAGE DAILY TRAFFIC	+ ALL LANES. HIGHEST NO. IN SEGMENT.
13	68-72	5		N	18-KIP EQUIVALENCY (,000)	+ ALL LANES ONE DIRECTION IN THOUSANDS. HIGHEST NO. IN SEG.
14	73	1		N	FUNCTIONAL CLASSIFICATION	+ VALUES 1-7. (SEE FTNOTE 1).
15	74-75	2		N	NUMBER OF LANES	+ MAIN LANES ONLY. HIGHEST NO. WITHIN SEGMENT.
16	76-77	2		N	SPEED LIMIT	+ FROM VISUAL EVALUATION, HIGHEST LEGAL SPEED. DEFAULT = 55.
17	78-79	2		N	RIS SURFACE TYPE	+ USED IN SEGMENT LIMIT DETERMINATION ONLY. (SEE FTNOTE 2).
18.A.1	80	1		A	VIS EVAL.(1ST) - LANE	+ (SEE FTNOTE 3). BLANK WHEN NO EVALUATION DATA PRESENT.
.2	81-86	6		N	- DATE EVALUATED	+ FORMAT = YYMMDD
.3	87-88	2		N	- COMMENT	+ (SEE PES VIS. EVAL. FORM 1624, 1625 OR 1626)
.4	89-90	2		N	- PAVEMENT TYPE	+ VALUES 01-10. (SEE FTNOTE 4).
.5	91-93	3		N	- FAIL-ACP-B	+ (FLEX=000,100,010,001/CRC = 000 TO 200).
.6	94-96	3		N	- TRAN-B-APPJNTSP	+ (FLEX=000,100,010,001/JCP=010 TO 100).
.7	97-99	3		N	- LONG-B-PCCP	+ (FLEX=000,100,010,001/JCP=000 TO 999).
.8	100-102	3		N	- RUT-SPALL-FAILJNT	+ (FLEX=000,100,010,001,200,020,002/CRC OR JCP=000 TO 999).
.9	103-105	3		N	- ALLG-CRCKSPC-LNGCRCK	+ (FLEX=000,100,010,001/CRC=000 TO 075/JCP=000 TO 999).
.10	106-108	3		N	- BLCK-PCCP-SHATSLB	+ (FLEX=000,100,010,001/CRC=000 TO 200/JCP=000 TO 999).
.11	109-111	3		N	- PATCH-PNCH-FAILURE	+ (FLEX=000,100,010,001/CRC=000 TO 200/JCP=000 TO 999).
.B	112-143	32		A&N	(2ND) - ALL ITEMS	+ (SEE 18.A FORMAT & DESC.). LANE BLANK WHEN NO 2ND EVAL.
.C	144-175	32		A&N	(3RD) - ALL ITEMS	+ (SEE 18.A FORMAT & DESC.). LANE BLANK WHEN NO 3RD EVAL.
.D	176-207	32		A&N	(4TH) - ALL ITEMS	+ (SEE 18.A FORMAT & DESC.). LANE BLANK WHEN NO 4TH EVAL.
.E	208-239	32		A&N	(5TH) - ALL ITEMS	+ (SEE 18.A FORMAT & DESC.). LANE BLANK WHEN NO 5TH EVAL.
.F	240-271	32		A&N	(6TH) - ALL ITEMS	+ (SEE 18.A FORMAT & DESC.). LANE BLANK WHEN NO 6TH EVAL.

ORIGINAL PES FILE RECORD LAYOUT (Continued)

ITEM NO.	FROM THRU	SIZE	DEC. POS.	FIELD TYPE	ITEM NAME	DESCRIPTION
19.A	272-277	6		N	MRM GENERAL - LAST UPDATE DATE	+ MAIN LANES ONLY. FORMAT = YYMMDD.
.B	278-279	2	1	N	- AVERAGE SI VALUE	+ MAIN LANES ONLY. 0.0 WHEN NONE TO AVERAGE. VALUES 0.1-5.0.
20.A.1.A	280	1		A	MRM - LANES R-V - CELL 1 - LANE	+ MAIN LANES WITH POSTS. BLANK WHEN NO SI. (SEE FTNOTE 5).
.B	281-282	2	1	N	- SI VALUE	+ MAIN LANES WITH POSTS. 0.0 WHEN NO SI. VALUES 0.1-5.0.
.2	283-324	42	-	A&N	- CELLS 2-15 - ALL	+ MAIN LNS W/ POSTS. OTHER CELLS. (14 OCCURS OF 20.A.1 FMT).
.B	325-369	45	-	A&N	- LANES L-P - CELLS 1-15 - ALL	+ MAIN LANES AGAINST POSTS. (SEE 20.A FORMAT).
.C	370-414	45	-	A&N	- LANES A-C - CELLS 1-15 - ALL	+ FRONTAGE ROAD LANES WITH POSTS. (SEE 20.A FORMAT).
.D	415-459	45	-	A&N	- LANES X-Z - CELLS 1-15 - ALL	+ FRONTAGE ROAD LANES AGAINST POSTS. (SEE 20.A FORMAT).
21.A	460-465	6		N	SKID GENERAL - LAST UPDATE DATE	+ MAIN LANES ONLY. FORMAT = YYMMDD. (SKID INFO. IS OPTIONAL)
.B	466-467	2		N	- AVERAGE SN VALUE	+ MAIN LANES ONLY. ZERO WHEN NONE TO AVERAGE.
22.A.1.A	468	1		A	SKID - LANES R-V - CELL 1 - LANE	+ MAIN LANES WITH POSTS. BLANK WHEN NO SN. (SEE FTNOTE 5).
.B	469-470	2		N	- SN VALUE	+ MAIN LANES WITH POSTS. ZERO WHEN NO SN.
.2	471-512	42		A&N	- CELLS 2-15 - ALL	+ MAIN LNS W/ POSTS. OTHER CELLS. (14 OCCURS OF 22.A.1 FMT).
.B	513-557	45		A&N	- LANES L-P - CELLS 1-15 - ALL	+ MAIN LANES AGAINST POSTS. (SEE 22.A FORMAT).
.C	558-602	45		A&N	- LANES A-C - CELLS 1-15 - ALL	+ FRONTAGE ROAD LANES WITH POSTS. (SEE 22.A FORMAT).
.D	603-647	45		A&N	- LANES X-Z - CELLS 1-15 - ALL	+ FRONTAGE ROAD LANES AGAINST POSTS. (SEE 22.A FORMAT).
23	648-650	3		N	SURFACE WIDTH	+ MAIN LANES ONLY. MEASURED IN FEET.
24	651	1		N	HIGHWAY DESIGN TYPE	+ VALUES 1-9. HIGHEST VALUE IN SEG. USED. (SEE FTNOTE 6).
25.A	652	1		A	SAMPLE SELCTION FLAG - MANDATORY	+ WHEN SET (NONBLANK), INDICATES SEGMENT WHICH MUST BE TESTED.
.B	653	1		A	- OTHER	+ WHEN SET (NONBLANK), INDICATES SOME SPECIAL-PURPOSE SEGMENT.
26.A.1.A	654-659	6		N	SSI - LANES R-V - CELL 1 - DATE	+ FORMAT = YYMMDD.
.B	660-662	3		N	AVG. SSI	+ ZERO WHEN NO SSI DATA.
.C	663-665	3		N	TEMP.	+ ZERO WHEN NO SSI DATA.
.2	666-677	12	-	N	- LANES L-P - CELLS 1-3 - ALL	+ MAIN LANES AGAINST POSTS. (SEE 26.A.1 FORMAT)
.3	678-689	12	-	N	- LANES A-C - CELLS 1-3 - ALL	+ FRONTAGE ROAD LANES W/ POSTS. (SEE 26.A.1 FORMAT).
.4	690-701	12	-	N	- LANES X-Z - CELLS 1-3 - ALL	+ FRONTAGE ROAD LANES AGAINST POSTS. (SEE 26.A.1 FORMAT).
27.A.1.A	702	1	-	A	SSI - LANES R-V - CELL 1 - LANE	+ MAIN LNS WITH POSTS. BLANK WHEN NO SSI DATA. (SEE FTNOTE 8).
.B	703-706	4	2	N	- GPHN1	+ MAIN LANES WITH POSTS. ZERO WHEN NO SSI DATA.
.C	707-710	4	2	N	- GPHN2	+ MAIN LANES WITH POSTS. ZERO WHEN NO SSI DATA.
.D	711-714	4	2	N	- GPHN3	+ MAIN LANES WITH POSTS. ZERO WHEN NO SSI DATA.
.E	715-718	4	2	N	- GPHN4	+ MAIN LANES WITH POSTS. ZERO WHEN NO SSI DATA.
.F	719-722	4	2	N	- GPHN5	+ MAIN LANES WITH POSTS. ZERO WHEN NO SSI DATA.
.G	723-726	4	2	N	- GPHN6	+ MAIN LANES WITH POSTS. ZERO WHEN NO SSI DATA.
.H	727-730	4	2	N	- GPHN7	+ MAIN LANES WITH POSTS. ZERO WHEN NO SSI DATA.
.2	731-846	116	-	A&N	- CELLS 2-5 - ALL	+ MAIN LNS W/ POSTS. OTHER CELLS. (4 OCCURS OF 27.A.1 FORMAT).
.B	847-991	145	-	A&N	- LANES L-P - CELLS 1-5 - ALL	+ MAIN LANES AGAINST POSTS. (SEE 27.A.1 FORMAT).
.C	992-1136	145	-	A&N	- LANES A-C - CELLS 1-5 - ALL	+ FRONTAGE ROAD LANES WITH POSTS. (SEE 27.A.1 FORMAT).
.D	1137-1281	145	-	A&N	- LANES X-Z - CELLS 1-5 - ALL	+ FRONTAGE ROAD LANES AGAINST POSTS. (SEE 27.A.1 FORMAT).
28.A.1	1282-1284	3		A	SCORES - RDWAY R-V - IDENTIFIER	+ MAIN RDWAY WITH POSTS. BLANK WHEN NO SCORE.(SEE FTNOTE 7)
.2	1285-1287	3	2	N	- MRM UTILITY	+ MAIN RDWAY WITH POSTS. 0.00 WHEN UNABLE TO CALCULATE.
.3	1288-1290	3	2	N	- SKID UTILITY	+ MAIN RDWAY WITH POSTS. FOR PES RELEASE 1.0, NOT AVAILABLE.
.4	1291-1293	3	2	N	- MAINT. COST	+ MAIN RDWAY WITH POSTS. FOR PES RELEASE 1.0, NOT AVAILABLE.
.5	1294-1296	3	2	N	- UNWEIGHTED VIS.	+ MAIN RDWAY WITH POSTS. 0.00 WHEN NO CALCULATION.
.6	1297-1299	3	2	N	- ADJUSTED VIS.	+ MAIN RDWAY WITH POSTS. 0.00 WHEN NO CALCULATION.
.7	1300-1302	3	2	N	- WEIGHTED VIS.	+ MAIN RDWAY WITH POSTS. 0.00 WHEN NO CALCULATION.
.8	1303-1305	3	2	N	- UNWEIGHTED PVT.	+ MAIN RDWAY WITH POSTS. 0.00 WHEN NO CALCULATION.
.9	1306-1308	3	2	N	- PVT. SCORE	+ MAIN RDWAY WITH POSTS. 0.00 WHEN NO CALCULATION.
.B	1309-1335	27	-	A&N	- RDWAY L-P - ALL ITEMS	+ MAIN ROADWAY AGAINST POSTS. (SEE 28.A FORMAT).
.C	1336-1362	27	-	A&N	- RDWAY A-C - ALL ITEMS	+ FRONTAGE ROADWAY WITH POSTS. (SEE 28.A FORMAT).
.D	1363-1389	27	-	A&N	- RDWAY X-Z - ALL ITEMS	+ FRONTAGE ROADWAY AGAINST POSTS. (SEE 28.A FORMAT).

ORIGINAL PES FILE RECORD LAYOUT (Continued)

GENERAL INFO. -
+ SIZE INDICATES FULL LENGTH OF ITEM DESCRIBED. NUMERICS ARE EXTERNAL DECIMAL.
+ DECIMAL POSITIONS (DEC. POS.) ARE INCLUDED IN SIZE. DECIMAL POINT IS IMPLIED.
+ FIELD TYPE A IS ALPHANUMERIC (BLANK,+,-,A,B,....,Z). FIELD TYPE N IS ETERNAL DECIMAL NUMERIC (0-9).
+ SOLE USER CONTROL OVER FILE DUMP IS THRU MANIPULATION OF REPORT HEADER (MANDATORY AND OTHER SAMPLE SELECTION FLAGS). OTHERWISE, ENTIRE DISTRICT OR ENTIRE STATE IS DUMPED.
+ FOR PES RELEASE 1.0, NO UTILITY SCORES ARE COMPUTED UNLESS ALL INFO. REQUIRED FOR CALCULATION OF PAVEMENT SCORE IS AVAILABLE.
+ ALL KEY INFO. (DISTRICT, MILEPOSTS, ...) AND ALL RIS INFO. (ADT, FUNC. CLASS., ...) ARE PRESENTED AS CAPTURED AT CONVERSION TIME, NORMALLY AROUND THE BEGINNING OF EACH FISCAL YEAR.
+ FOR PES RELEASE 1.0, SKID INFORMATION IS WHOLLY OPTIONAL. INCLUSION / EXCLUSION WILL NOT ALTER CALCULATED UTILITIES AND SCORES. NOTE THAT SKID UTILITY IS NOT PRESENTLY AVAILABLE.
+ QUESTIONS SHOULD BE DIRECTED TO FILE D-18, TRAFFIC SAFETY AND MAINTENANCE OPERATIONS DIVISION.

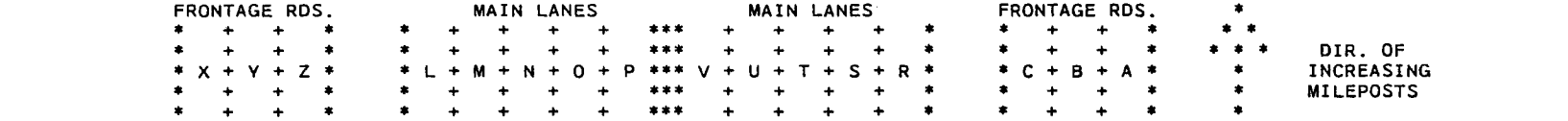
FOOTNOTE 1 - A CHANGE IN FUNCTIONAL CLASS. IS ONE CAUSE OF SEGMENTATION. DECODIFICATION IS FROM D-10 PLATE 25 (REV. 5/1/84).

1. INTERSTATE
2. OTHER URBAN FREEWAY AND EXPRESSWAY
3. RURAL PRINCIPAL ARTERIAL -OR-
3. URBAN CONNECTING LINKS OF RURAL ARTERIALS -OR-
3. OTHER URBAN PRINCIPAL ARTERIALS
4. MINOR ARTERIAL ROAD OR STREET
5. RURAL MAJOR COLLECTOR -OR-
5. URBAN COLLECTOR STREET
6. RURAL MINOR COLLECTORS
7. LOCAL ROAD OR STREET

FOOTNOTE 2 - A CHANGE IN SURFACE TYPE GROUP IS ANOTHER CAUSE OF SEGMENTATION. CODE 61 AND 62 HAVE THEIR OWN GROUPS WHILE ALL OTHER CODES FORM A THIRD GROUP. WITHIN THAT 3ND GROUP, THE HIGHEST NON-71 AND -81 CODE IS USED. THE READER SHOULD UNDERSTAND THAT THIS DATA ITEM IS USED DURING INITIAL SEGMENTATION (CONVERSION) ONLY AND IS INCORPORATED IN THE PES RECORD FRAMEWORK SOLELY FOR DOCUMENTATION PURPOSES. PAVEMENT TYPE, FOUND UNDER VISUAL EVALUATION HEADING, IS THE VITAL PAVEMENT ITEM USED IN UTILITY SCORE CALCULATIONS. SURFACE TYPE AGAIN IS DRAWN FROM D-10 PLATE 25.

51. BITUMINOUS SURFACE - TREATED (UNDER 1 IN.)
52. MIXED BIT. SURFACE (BASE & SURF. UNDER 7 IN.)
53. MIXED BIT. SURFACE (BASE & SURF. 7 IN. OR MORE)
54. BIT. PENETRATION (BASE & SURF. 7 IN. OR MORE)
55. BIT. CONCRETE OR SHEET ASPHALT (1 IN. OR MORE)
56. BIT. PENETRATION (BASE & SURF. UNDER 7 IN.)
61. CONCRETE (JCP- W/ OR W/O ASPH. SURF. UNDER 1 IN)
62. CONCRETE (CRCP-W/ OR W/O ASPH. SURF. UNDER 1 IN)
71. BRICK ROAD
81. BLOCK ROAD

FOOTNOTE 3 - LANE DESIGNATIONS FOLLOW. DESCRIPTION IS FROM OUTSIDE LANE TOWARD CENTERLINE OR MEDIAN. THUS, A TWO-LANE DESIGN TYPE 1 FACILITY WILL SHOW LANES R & L ONLY. A DESIGN TYPE 9 FACILITY WILL HAVE MANY LANES (R,S,...,L,M,....,X,A,....) BUT CANNOT HAVE LANE T, FOR INSTANCE, WITHOUT AN ADJACENT LANE S (TO THE RIGHT) WHICH IS, IN TURN, ADJACENT TO LANE R.



FOOTNOTE 4 - VISUALLY EVALUATED (OBTAINED) PAVEMENT TYPE DECODIFICATION HAS BEEN PULLED FROM FORM 1626.

1. CONTINUOUSLY REINFORCED CONCRETE
2. JOINTED REINFORCED CONCRETE
3. JOINTED PLAIN CONCRETE
4. THICK ASPHALTIC CONCRETE (OVER 5.5 IN.)
5. INTERMEDIATE ASPHALTIC CONCRETE (2.5-5.5 IN.)
6. THIN SURFACED FLEXIBLE BASE (UNDER 2.5 IN.)
7. COMPOSITE (ASPHALT SURFACED CONCRETE)
8. OVERLAID AND/OR WIDENED OLD CONCRETE
9. OVERLAID AND/OR WIDENED OLD FLEXIBLE
10. THIN SURFACED FLEX. BASE (SURF. TRT. - SEAL COAT)

FOOTNOTE 5 - 15 CELLS FOR EACH OF 4 SETS OF LANES (EACH LANE SET REPRESENTS A POTENTIAL ROADWAY) ARE RESERVED. LANE SETS WHICH, DUE TO HIGHWAY DESIGN, DO NOT EXIST, ARE VOID OF DATA. EACH CELL CONSTITUTES 0.2 MILES OF A SEGMENT (THUS, MAXIMUM SEGMENT LENGTH OF 3.0 MILES) WITH THE EXCEPTION THAT THE LAST CELL OF AN ODD MILEAGE SEGMENT (IE, LENGTH = 1.9 MILES) WILL BE UNDERSTOOD TO CONSTITUTE ONLY 0.1 MILE. CELLS OUTSIDE THE RANGE NEEDED TO EXHAUST THE SEGMENT LENGTH ARE VOID OF DATA. REGARDLESS OF DIRECTION-OF-TRAVEL (WITH OR AGAINST POSTS), CELLS WITHIN A 15-MEMBER UNIT TIE TOGETHER END-TO-END FROM LOW NUMBERED CELL TO HIGH NUMBERED CELL (1 TO 15) AND COVER THE MILEAGE TRAVERSED FROM SEGMENT BEGIN POST & DISPLACEMENT TO SEGMENT END POST & DISPLACEMENT. THUS, CELL 1 FOR EACH OF THE 4 LANE SETS CONSTITUTES THE FIRST 0.2 MILES OF A SEGMENT (ROW-LINE TO ROW-LINE).

ORIGINAL PES FILE RECORD LAYOUT (Continued)

FOOTNOTE 6 - HIGHWAY DESIGN CODES ARE PULLED FROM D-10 PLATE 25. NUMBER OF ROADWAYS HAS BEEN PARENTHETICALLY APPENDED.

1. TWO-WAY TRAFFIC (1 ROADWAY)
2. ONE-WAY TRAFFIC (1 ROADWAY)
3. BOULEVARD (2 ROADWAYS)
4. EXPRESSWAY - NO SERVICE ROADS (2 ROADWAYS)
5. EXPRESSWAY - ONE SERVICE ROAD (3 ROADWAYS)
6. EXPRESSWAY - TWO SERVICE ROADS (4 ROADWAYS)
7. FREEWAY - NO SERVICE ROADS (2 ROADWAYS)
8. FREEWAY - ONE SERVICE ROAD (3 ROADWAYS)
9. FREEWAY - TWO SERVICE ROADS (4 ROADWAYS)

FOOTNOTE 7 - AS MENTIONED IN FOOTNOTE 5, EACH LANE SET REPRESENTS A POTENTIAL ROADWAY. HOWEVER, THE FOLLOWING HANDLING RULES SHOULD BE KEPT IN MIND. UTILITY AND PAVEMENT SCORES MAY / MAY NOT BE CALCULATED FOR EXISTING ROADWAYS DEPENDING UPON THE AVAILABILITY OF REQUIRED DATA ITEMS. FOR PES RELEASE 1.0, EITHER ALL SCORES WILL BE CALCULATED OR NONE WILL BE FOUND.

1. FOR HIGHWAY DESIGN TYPES 1 & 2, ONLY1 ROADWAY EXISTS. LANE SETS R-V AND L-P DATA ARE MERGED WITH RESULTING SCORE OUTPUT (IF COMPUTATIONS POSSIBLE) AS A ROADWAY IDENTIFIED BY R-L AND OCCUPYING THE ROADWAY R-V AREA.
2. FOR DESIGN TYPES 3, 4, AND 7, THE ROADWAY R-V AND L-P AREAS RECEIVE SCORE CALCULATIONS.
3. FOR DESIGN TYPES 5 AND 8, ROADWAY AREAS R-V, L-P, AND (A-C OR X-Z) CAN HOLD SCORES. THE USER MUST DETERMINE WHICH ONE OF THE A-C OR X-Z AREAS CONTAINS SCORES (IF CALCULATED).
4. FOR DESIGN TYPES 6 AND 9, ALL ROADWAY AREAS ARE POTENTIAL RECEIVERS OF SCORE CALCULATIONS.

FOOTNOTE 8 -- 5 CELLS FOR EACH OF 4 SETS OF LANES (EACH LANE SET REPRESENTS A POTENTIAL ROADWAY) ARE RESERVED. LANE SETS WHICH, DUE TO HIGHWAY DESIGN, DO NOT EXIST, ARE VOID OF DATA. EACH CELL CONSTITUTES ONE-FIFTH OF A SEGMENT AND GEOPHONE READINGS FOR EACH OF THE 5 CELLS ARE MANDATORY.

General Information File Record Layout

Field	Size/ Type	Note	PES Item No.	Description
* HWYDIST	2N		1	Highway District Number
	2N		2	Maintenance Section Number
* CNTYNUM	3N		3	County Number
* HWYPREFX	2C		4.A	Highway System
* HWYNUM	4N(C)	2	4.B	Highway Number
* HWYSUFFX	1C		4.C	Highway Suffix
* BEGMPST	3N		5.A	Segment Begin - Milepost No
* BDISSIGN	1C		5.B	- Displacement Sign
* BMPSTDIS	3.1N	1	5.C	(9.9) - Displacement Value
*	3C	11		Laneset
ENDMPST	3N		6.A	Segment End - Milepost No.
EDISSIGN	1C		6.B	- Displacement Sign
EMPSTDIS	3.1N	1	6.C	(9.9) - Displacement Value
	3C/N	2	7.A.1	Intermediate Post- 1st Post
	3C/N	2	7.A.2	- 2nd Post
	3.1N	2	7.B.1	- Distance to 1st
	3.1N	2	7.B.2	- Distance to 2nd
	5.1N	2	8.A	Route-Thru-County - Begin Value
	5.1N	2	8.B	- End Value
	3.1N		9	Segment Length
	6N		10	Last Update Date (YYMMDD)
	10.2N		11	Segment Maintenance Cost
	6N		12	Average Daily Traffic
	5N		13	18-Kip Equivalency
FUNCLAS	1N		14	Functional Classification
NO_LANES	2N		15	Number of lanes

* Key field

General Information File Record Layout

Field	Size/ Type	Note	PES Item No.	Description
	2N		16	Speed Limit
	2N		17	RIS Surface Type
	3N		23	Surface Width
	1N		24	Highway Design Type
	1C	6	25.A	Sample Selection Flag - Mandatory
	1C	6	25.B	- Other

* Key Field

Visual Information File Record Layout

	Field	Size/ Type	Note	PES Item No.	Description
*	HWYDIST	2N		1	Highway District Number
		2N		2	Maintenance Section Number
*	CNTYNUM	3N		3	County Number
*	HWYPREFX	2C		4.A	Highway System
*	HWYNUM	4N(C)	2	4.B	Highway Number
*	HWYSUFFX	1C		4.C	Highway Suffix
*	BEGMPST	3N		5.A	Segment Begin - Milepost No
*	BDISSIGN	1C		5.B	- Displacement Sign
*	BMPSTDIS	3.1N	1	5.C	(9.9) - Displacement Value
*		3C	11		Laneset
		1C	3	18.A-F.1	Visual Evaluation - Lane
		6N	3	18.A-F.2	(YYMMDD) - Date Evaluated
		2N	3	18.A-F.3	- Comment
		2N	3	18.A-F.4	- Pavement Type
	FAILMILE	3N	3	18.A-F.5	- Failures
	TRANCRMD	3N	3	18.A-F.6	- Transverse Cracks
	LONGCRMD	3N	3	18.A-F.7	- Longitudinal Cracks
		3N	3	18.A-F.8	- Rutting
	ALLGCRMD	3N	3	18.A-F.9	- Alligator Cracks
	BLKCRKMD	3N	3	18.A-F.10	- Block Cracks
	PATCHFR	3N	3	18.A-F.11	- Maintenance Patching

* Key field

Maysmeter Information File Record Layout

Field	Size/ Type	Note	PES Item No.	Description
* HWYDIST	2N		1	Highway District Number
	2N		2	Maintenance Section Number
* CNTYNUM	3N		3	County Number
* HWYPREFX	2C		4.A	Highway System
* HWYNUM	4N(C)	2	4.B	Highway Number
* HWYSUFFX	1C		4.C	Highway Suffix
* BEGMPST	3N		5.A	Segment Begin - Milepost No
* BDISSIGN	1C		5.B	- Displacement Sign
* BMPSTDIS	3.1N	1	5.C	(9.9) - Displacement Value
*	3C	11		Laneset
	6N		19.A	MRM General - Last Update Date (YYMMDD)
	3.1N		19.B	- Average SI Value
	1C	4	20.A-D.1.A	MRM - Lane
	3.1N	4	20.A-D.1.B	- Lane SI Value
	56C&N	4	20.A-D.2	- Lane & SI Values (14 occurrences of 20.A.1)
SIMEAN	7.5N	12		MRM Mean for the Laneset
SISD	7.5N	12		MRM Std. Dev. for the Laneset
SIHI	3.1N	12		MRM High for the Laneset
SILOWVAL	3.1N	12		MRM Low for the Laneset
SICOUNT	3N	12		MRM Count of Obs. for the Laneset

* Key field

Skid Information File Record Layout

Field	Size/ Type	Note	PES Item No.	Description
* HWYDIST	2N		1	Highway District Number
	2N		2	Maintenance Section Number
* CNTYNUM	3N		3	County Number
* HWYPREFX	2C		4.A	Highway System
* HWYNUM	4N(C)	2	4.B	Highway Number
* HWYSUFFX	1C		4.C	Highway Suffix
* BEGMPST	3N		5.A	Segment Begin - Milepost No
* BDISSIGN	1C		5.B	- Displacement Sign
* BMPSTDIS	3.1N	1	5.C	(9.9) - Displacement Value
*	3C	11		Laneset
	6N		21.A	Skid General - Last Update (YYMMDD) Date
	2N		21.B	- Average SN Value
	1C	5	22.A-D.1.A	Skid - Lane
	2N	5	22.A-D.1.B	- Lane SN Values
	42 C&N	5	22.A-D.2	- Lane & SN Value (14 occurrences of 22.A.1)
SKIDNUMN	2N	13		SN Mean for the Laneset
SNSD	2N	13		SN Std. Dev. for the Laneset
SKIDNUMH	2N	13		SN High for the Laneset
SKIDSD	2N	13		SN Low for the Laneset
SNCOUNT	2N	13		SN Count of Obs. for the Laneset

* Key field

Structural Strength Index Information File Record Layout

	Field	Size/ Type	Note	PES Item No.	Description
*	HWYDIST	2N		1	Highway District Number
		2N		2	Maintenance Section Number
*	CNTYNUM	3N		3	County Number
*	HWYPREFX	2C		4.A	Highway System
*	HWYNUM	4N(C)	2	4.B	Highway Number
*	HWYSUFFX	1C		4.C	Highway Suffix
*	BEGMPST	3N		5.A	Segment Begin - Milepost No
*	BDISSIGN	1C		5.B	- Displacement Sign
*	BMPSTDIS	3.1N	1	5.C	(9.9) - Displacement Value
*		3C	11		Laneset
		6N	7	26.A.1-4.A	SSI Date (YYMMDD)
	RWSSIAVG	3N	7	26.A.1-4.B	Average SSI
	SSITEMP	3N	7	26.A-D.1.A	SSI Temperature
		1C	8	27.A-D.1.A	SSI Lane Station 1
	SSIGPH11	5.2N	8	27.A-D.1.B	Geophone Reading 1 Station 1
	SSIGPH12	5.2N	8	27.A-D.1.C	Geophone Reading 2 Station 1
	SSIGPH13	5.2N	8	27.A-D.1.D	Geophone Reading 3 Station 1
	SSIGPH14	5.2N	8	27.A-D.1.E	Geophone Reading 4 Station 1
	SSIGPH15	5.2N	8	27.A-D.1.F	Geophone Reading 5 Station 1
	SSIGPH16	5.2N	8	27.A-D.1.G	Geophone Reading 6 Station 1
	SSIGPH17	5.2N	8	27.A-D.1.H	Geophone Reading 7 Station 1
		1C	8	27.A-D.2.A	SSI Lane Station 2
	SSIGPH21	5.2N	8	27.A-D.2.B	Geophone Reading 1 Station 2
	SSIGPH22	5.2N	8	27.A-D.2.C	Geophone Reading 2 Station 2
	SSIGPH23	5.2N	8	27.A-D.2.D	Geophone Reading 3 Station 2
	SSIGPH24	5.2N	8	27.A-D.2.E	Geophone Reading 4 Station 2
	SSIGPH25	5.2N	8	27.A-D.2.F	Geophone Reading 5 Station 2
	SSIGPH26	5.2N	8	27.A-D.2.G	Geophone Reading 6 Station 2
	SSIGPH27	5.2N	8	27.A-D.2.H	Geophone Reading 7 Station 2
		1C	8	27.A-D.3.A	SSI Lane Station 3
	SSIGPH31	5.2N	8	27.A-D.3.B	Geophone Reading 1 Station 3
	SSIGPH32	5.2N	8	27.A-D.3.C	Geophone Reading 2 Station 3
	SSIGPH33	5.2N	8	27.A-D.3.D	Geophone Reading 3 Station 3
	SSIGPH34	5.2N	8	27.A-D.3.E	Geophone Reading 4 Station 3
	SSIGPH35	5.2N	8	27.A-D.3.F	Geophone Reading 5 Station 3
	SSIGPH36	5.2N	8	27.A-D.3.G	Geophone Reading 6 Station 3

* Key Field

Structural Strength Index Information File Record Layout

Field	Size/ Type	Note	PES Item No.	Description
SSIGPH37	5.2N	8	27.A-D.3.H	Geophone Reading 7 Station 3
	1C	8	27.A-D.4.A	SSI Lane Station 4
SSIGPH41	5.2N	8	27.A-D.4.B	Geophone Reading 1 Station 4
SSIGPH42	5.2N	8	27.A-D.4.C	Geophone Reading 2 Station 4
SSIGPH43	5.2N	8	27.A-D.4.D	Geophone Reading 3 Station 4
SSIGPH44	5.2N	8	27.A-D.4.E	Geophone Reading 4 Station 4
SSIGPH45	5.2N	8	27.A-D.4.F	Geophone Reading 5 Station 4
SSIGPH46	5.2N	8	27.A-D.4.G	Geophone Reading 6 Station 4
SSIGPH47	5.2N	8	27.A-D.4.H	Geophone Reading 7 Station 4
	1C	8	27.A-D.5.A	SSI Lane Station 5
SSIGPH51	5.2N	8	27.A-D.5.B	Geophone Reading 1 Station 5
SSIGPH52	5.2N	8	27.A-D.5.C	Geophone Reading 2 Station 5
SSIGPH53	5.2N	8	27.A-D.5.D	Geophone Reading 3 Station 5
SSIGPH54	5.2N	8	27.A-D.5.E	Geophone Reading 4 Station 5
SSIGPH55	5.2N	8	27.A-D.5.F	Geophone Reading 5 Station 5
SSIGPH56	5.2N	8	27.A-D.5.G	Geophone Reading 6 Station 5
SSIGPH57	5.2N	8	27.A-D.5.H	Geophone Reading 7 Station 5

Score Information File Record Layout

	Field	Size/ Type	Note	PES Item No.	Description
*	HWYDIST	2N		1	Highway District Number
		2N		2	Maintenance Section Number
*	CNTYNUM	3N		3	County Number
*	HWYPREFX	2C		4.A	Highway System
*	HWYNUM	4N(C)	2	4.B	Highway Number
*	HWYSUFFX	1C		4.C	Highway Suffix
*	BEGMPST	3N		5.A	Segment Begin - Milepost No
*	BDISSIGN	1C		5.B	- Displacement Sign
*	BMPSTDIS	3.1N	1	5.C	(9.9) - Displacement Value
*		3C	11		Laneset
		3C	9	28.A-D.1	Scores - Identifier
		4.2N	10	28.A-D.2	- MRM Utility
		4.2N	10	28.A-D.3	- Skid Utility
		4.2N	10	28.A-D.4	- Maintenance Cost
	UVURS	4.2N	10	28.A-D.5	- Unweighted Visual
		4.2N	10	28.A-D.6	- Adjusted Visual
		4.2N	10	28.A-D.7	- Weighted Visual
		4.2N	10	28.A-D.8	- Unweighted Pavement
	PESPRS	4.2N	10	28.A-D.9	- Pavement Score

* Key Field

Notes on Output Record Layouts

General Notes:

A. All decimal points are implied in the Original PES File. The Size/Type column in the output layouts include 1 extra position for the decimal which physically exists, except for the Beginning and Ending Milepoint Displacements. The Displacements have an implied decimal point in the output files also.

B. All dates are in the format of YYMMDD.

C. Refer to the original PES record layout for an explanation of each PES item.

D. The Visual Evaluation is taken on the worst lane in the roadway. There will be MRM and SSI data for all roadways in which a visual evaluation has been made. However, for a Highway Design Code of 1 or 2, the roadway would be R-L instead of R-V and L-P; hence the visual evaluation could be taken in lane 'L' and the MRM or the SSI could be taken in the 'R' lane. For Highway Design Code 1 or 2, the program only creates one record in each modified PES file (See Specific Note 3). If the Visual evaluation, MRM, SSI and the Score data are not available for all the roadways which exist for the segment (see Highway Design), the program reports the test segment and which roadway and data are missing. Currently, there will only be up to 4 visual evaluation lanes - 1 per roadway.

E. The following formula is used to calculate the standard deviation

$$\sigma = \sqrt{\frac{(x - \bar{x})^2}{n - 1}} \qquad \text{where } x = \frac{\sum_{i=1}^{n} x_i}{n}$$

F. A field name is provided when the field directly corresponds to a field in the dBASE III Texas Flexible Pavement Database. The field name is the same as that used in the database.

Notes on Output Record Layouts

Specific Notes:

The record layouts refer to these notes in the Note column of the layout.

1. Stored without the decimal place (eg. 3.0 is stored as 30)

2. See the Original PES File Layout documentation.

3. A separate record should be created for original PES items 18.A.1, 18.B.1, 18.C.1, 18.D.1, 18.E.1, and 18.F.1. See General Note D.

4. If the Visual Evaluation lane is in the R-V laneset, then the program extracts laneset R-V cells 1-15 for the MRM readings. The MRM reading are in the format of lane-SI value (eg. R01 - the SI value has an implied decimal point between the two digits). If the Visual Evaluation lane is in the L-P laneset, the program extracts the L-P laneset, MRM readings, etc. (If the Highway Design Code is 1 or 2, the SI values are in the R-V cells.)

5. If the Visual Evaluation lane is in the R-V laneset, then the program extracts the Skid cells 1-15 for the R-V laneset. If the Visual Evaluation lane is in the L-P laneset, the program extracts the L-P laneset skid values, etc. The skid values are in the format of lane-SN value (eg. L20). (If the Highway Design Code is 1 or 2, the Skid values are in the R-V cells.)

6. The program only extracts PES records with an 'A' in the Mandatory Sample Selection Flag field and an 'E', 'F', 'G', 'H', 'I', 'N', 'O', 'P', 'Q', or 'R' in the Other Sample Selection Flag field.

7. As with the MRM and Skid data, the SSI data is given for 4 possible roadways. If the Visual Evaluation Lane is in the R-V laneset, then the program extracts the R-V laneset SSI data, etc. (If the Highway Design Code is 1 or 2, the SSI values are in the R-V cells.)

8. The actual geophone reading for the lanesets are provided in item 27. There are 5 sets of 7 geophone readings per roadway. The lane in which the reading was taken as well as the 7 readings are given for each of the 5 sets of data. The program extracts the readings for the appropriate roadway based on the Visual Evaluation Lane.

9. See Footnote 7 of the Original PES record layout documentation.

Notes on Output Record Layouts

10. The program picks up the appropriate roadway scores based on the Visual Evaluation Lane and Footnote 7 of the original PES File layout.

11. The laneset field will contains the laneset in which the Visual Evaluation Lane is in (eg. 'R-V', 'R-L', 'L-P', 'A-C', 'X-Z').

12. The program calculates the MRM Mean, Standard Deviation, High Value, Low Value, and Count of Observations for the laneset.

13. The program calculates the SN (Skid) Mean, High Value, Low Value, Standard Deviation, and Count of Observations in the laneset.

PES Modification Program
Program Specification (continued)

Missing Data Report Layout

Texas Flexible Pavement Database
PES Modification Program
Missing Data Report

Hwy Dist	Hwy Sec	Cnty Num	Highway	Begin Mile pt.	Hwy Des	Laneset	---- Missing data ---- Vis	MRM	Skd	SSI	Scr
xx	xx	xxx	xxxxxxxx	xxx x xxx	x	x-x	x	x	x	x	x

Notes:
1. Highway - Highway System, Number, Suffix
2. Highway Design - Ittem 24 and Footnote 6 of the Original PES File Record Layout Documentation - Values 1-9
3. Laneset - R-V, L-P, R-L, A-C, X-C
4. Place an 'X' in the column for which data is missing.

PROGRAM LISTING

```
{*  File name:      PESMOD.PAS
 *  Program name:   pesmod
 *  Project 2456:   Texas Flexible Pavement Database Conversion
 *  TAMU/TTI
 *  Written by:     Victor Wong
 *  Written on:     Feb 2, 1988
 *  Last updated:   May 13, 1988
 *  Purpose:        This Turbo PASCAL program reads in PES data
 *                  and extract information for the project
 *                  and output it in comma-delimited format
 *                  (suitable for DBASE III + and BASIC).
}

{>>> PROGRAM PESMOD <<<}
program pesmod;

   {>>> constant declarations <<<}
   const
      comma = ',';
      decpt = '.';
      blank = ' ';
      zero  = '0';
      item_array_size = 15;

   {>>> type declarations <<<}
   type
      item_array_range = array[1..item_array_size] of real;
      characters = set of 'A'..'Z';
      file_str = string[30];

      laneset = (rv,lp,ac,xz);
      mrm_rec = record                    {      MRM}
      lane: array[1..15] of char;         {1C    - lane}
      si: array[1..15] of string[2];      {2.1N  - SI value}
                                          {      (15 sets for each laneset)}
                                          {** NOTE: lane blank if no si **}
                                          {** NOTE: SI value 0.0 if no si }
                                          {         values 0.1 - 5.0 **}
      end;

      skid_rec = record                   {      skid}
      lane:array[1..15] of char;          {1C    - lane}
      sn:array[1..15] of string[2];       {2N    - SN value}
                                          {      (15 sets for each laneset)}
                                          {** NOTE: lane blank when no SN **}
                                          {**       SN value 0.0 when no
data**}
      end;
```

```
      ssi_gen_rec = record                {      SSI general}
      date      : string[6];              {6N    - date YYMMDD}
      avg       : string[3];              {3N    - average SSI}
      temp      : string[3];              {3N    - temperature}
                                          {      (1 set for each laneset)}
                                          {** NOTE: avg, temp zero if no data}

      end;

      ssi_rec = record                    {      SSI}
      lane: array[1..5] of char;          {1C    - Lane}
      gph1:array[1..5] of string[4];      {4.2N  - gphn1}
      gph2:array[1..5] of string[4];      {4.2N  - gphn2}
      gph3:array[1..5] of string[4];      {4.2N  - gphn3}
      gph4:array[1..5] of string[4];      {4.2N  - gphn4}
      gph5:array[1..5] of string[4];      {4.2N  - gphn5}
      gph6:array[1..5] of string[4];      {4.2N  - gphn6}
      gph7:array[1..5] of string[4];      {4.2N  - gphn7}
                                          {      (5 such set for each
laneset)}

                                          {** NOTE: lane blank when no ssi
                                                   gph zero when no data **}
      end;

      score_rec = record                  {        scores}
      id        : string[3];              {3C      - identifier}
      mrm       : string[3];              {3.2N    - MRM utility}
      skid      : string[3];              {3.2N    - skid utility}
      mncost    : string[3];              {3.2N    - maintenance cost}
      uwvis     : string[3];              {3.2N    - unweighted visual}
      adjvis    : string[3];              {3.2N    - adjusted visual}
      wvis      : string[3];              {3.2N    - weighted visual}
      uwpvmt    : string[3];              {3.2N    - unweigthed pavement}
      pvmt      : string[3];              {3.2N    - Pavement score}
                                          {        (1 such set for each
laneset)}

                                          {** NOTE: id blank when no score
                                                   others 0.0 when no cal.**}
      end;

   {>>> variables declarations <<<}
   var
      hwydist       : string[2];          {2N    district no.}
      hwysect       : string[2];          {2N    maintenance section no.}
      cntynum       : string[3];          {3N    county no.}

                                          {      highway}
      hwyprefx      : string[2];          {2C    - prefix}
      hwynum        : string[4];          {4N    - number}
      hwysuffx      : char;               {1C    - suffix}

                                          {      segemnt begin}
      bmpnum        : string[3];          {3N    - milepost no.}
      bmpdispsgn    : char;               {1C    - displacement sign}
      bmpdispval    : string[2];          {2.1N  - displacement value}
```

```
                                        {       segment end}
    empnum          : string[3];        {3N     - milepost no.}
    empdispsgn      : char;             {1C     - displacement sign}
    empdispval      : string[2];        {2.1N   - displacement value}

                                        {       intemediate post}
    intpst1         : string[3];        {3N     - 1st post no.}
    intpst2         : string[3];        {3N     - 2nd post no.}
    intdis1         : string[2];        {2.1N   - distance to 1st}
    intdis2         : string[2];        {2.1N   - distance to 2nd}

                                        {       route-thru-county}
    rtcntybeg       : string[4];        {4.1N   - begin value}
    rtcntyend       : string[4];        {4.1N   - end value}

    seglngth        : string[2];        {2.1N   segment length}
    segupdat        : string[6];        {6N     last updated date}
    segmncost       : string[9];        {9.2N   segment maintenance cost}
    adt             : string[6];        {6N     average daily traffic}
    a18k            : string[5];        {5N     18-kip equivalency (x 1000)}
    funclas         : char;             {1N     functional classification}
    numlanes        : string[2];        {2N     no. of lanes}
    spdlimit        : string[2];        {2N     speed limit}
    rissurf         : string[2];        {2N     RIS surface type}

                                        {       visual evaluation}
    vislane : array[1..6] of char;      {1C     - lane}
    visdate : array[1..6] of string[6];{6N      - date evaluated}
    viscomm : array[1..6] of string[2];{2N      - comment}
    vispvmt : array[1..6] of string[2];{2N      - pavement type}
    visfail : array[1..6] of string[3];{3N      - fail-acp-b}
    vistran : array[1..6] of string[3];{3N      - tran-b-appjntsp}
    vislong : array[1..6] of string[3];{3N      - long-b-pccp}
    visrut  : array[1..6] of string[3];{3N      - rut-spall-failjnt}
    visallg : array[1..6] of string[3];{3N      - allg-crckspc-lngcrck}
    visblck : array[1..6] of string[3];{3N      - blck-pccp-shatslb}
    visptch : array[1..6] of string[3];{3N      - patch-pnch-failure}
                                        {       (6 occurrences of vis.
eval.)}
                                        {** NOTE: lane blank if no eval. **}

                                        {       MRM general}
    mrmupdat        : string[6];        {6N     - last updated date YYMMDD}
    mrmavgsi        : string[2];        {2.1N   - average SI value}
                                        {         0.0 when none to avg}

                                        {       MRM}
    mrm_data        : array [laneset] of mrm_rec;

                                        {       Skid General}
    skidupdat       : string[6];        {6N     - Last updated date YYMMDD}
    skidavgsn       : string[2];        {2N     - average Sn value}
                                        {** NOTE: 0.0 when none to avg **}
```

```
                                        {      Skid}
      skid_data      : array [laneset] of skid_rec;

      surfwidth      : string[3];         {3N    surface width}
      hwydesign      : char;              {1N    highway design type}

                                          {      sample selection flag}
      smplflgman     : char;              {1C    - mandatory}
      smplflgoth     : char;              {1C    - other}
                                          {** NOTE: mandatory non-blank
                                                    -> segment must be tested
                                                    other non-blank
                                                    -> special-purpose
segment}

                                          {      SSI general}
      ssi_gen_data   : array [laneset] of ssi_gen_rec;

                                          {      SSI}
      ssi_data       : array [laneset] of ssi_rec;

                                          {      scores}
      score_data     : array [laneset] of score_rec;

      lane_set       : string[3];         {3N  laneset}

      simean         : real;              {MRM 6.5N Mean for the laneset}
      sisd           : real;              {    6.5N Std. Dev. for the laneset}
      sihi           : real;              {    2.1N high value for the
laneset}
      silo           : real;              {    2.1N low value for the laneset}
      sicount        : integer;           {    3N count of obs. for the
laneset}

      skidmean       : real;              {skid 2N Mean for the laneset}
      skidsd         : real;              {     2N Std. Dev for the laneset}
      skidhi         : real;              {     2N High value for the laneset}
      skidlo         : real;              {     2N Low value for the laneset}
      skidcount      : integer;           {     2N count of obs. for the
laneset}

      ls,ls1,begin_ls: laneset;
      l              : characters;
      i,j,
      vis_count,visln_pos,
      temp_si, temp_sn,
      err_code       : integer;
      mrm_stat_array,
      skid_stat_array: item_array_range;
      mrm_set_flag,
      skid_set_flag,
      ssigen_set_flag,
      ssi_set_flag,
```

```
      score_set_flag,
      vis_set_flag,
      mrm_ln_flag,
      skid_ln_flag,
      ssi_ln_flag,
      vis_ln_flag   : boolean;
      tempout,
      report_file,
      report_data_file,
      in_file,
      gen_file,
      vis_file,
      mrm_file,
      skd_file,
      ssi_file,
      scr_file      : text;
      filename      : file_str;
      krecords_checked,
      records_checked,
      krecord_no,
      record_no     : integer;
      record_no_pos : integer;

{-------------------------------------------------------------------------
--}
   {>>> procedure stat: It takes an array of real numbers and <<<}
   {>>>                 find the high and low value, mean, std<<<}
   {>>>                 deviation, and count of non-zero data <<<}

   procedure stat (item_array: item_array_range; array_size: integer;
                   var hi,lo,mean,sd: real;
                   var count: integer);
      {----variable declaration---}
      var
         sum,sd_sum: real;
         i: integer;
         still_have_data: boolean;

      {---beginning of the procedure---}
      begin
         {---initialize variables---}
         sum:= 0.0;
         sd_sum:= 0.0;
         count:= 1;
         hi:= item_array[1];
         lo:= item_array[1];
         still_have_data:= true;

         {---loop to read data,perform summation and counting---}
         while still_have_data do
            if item_array[count]=0.0 then
               still_have_data:= false
            else
               begin
```

```
                sum:= sum + item_array[count];
                if item_array[count] > hi then hi:= item_array[count];
                if item_array[count] < lo then lo:= item_array[count];
                count:= count + 1;
                if count > array_size then still_have_data:= false
             end;
          count:= count - 1;

          {---calculate the mean---}
          if count <> 0 then mean:= sum/count;

          {---calculate the std. dev.---}
          if count = 1 then
             sd:= 0.0
          else begin
             for i:= 1 to count do
                sd_sum:= sd_sum + sqr (item_array[i] - mean);
             sd:= sqrt (sd_sum/(count-1));
          end;

       end;
       {---ending of the procedure stat---}

{-------------------------------------------------------------------------
--}

   {>>> procedure OpenInFile : This procedure asks user for the input <<<}
   {>>>                        file name and tries to open it.        <<<}

   procedure OpenInFile (var in_file: text; var filename: file_str);
      {---variable declaration---}
      var
         infilename: string[20];
         ok: boolean;
         yr: string[2];

      {---beginning of procedure openinfile---}
      begin
         repeat
            filename:= '\PAVEDB\PES';
            infilename:= filename+ '.DAT';

            assign (in_file, infilename);
            {$I-} reset (in_file) {$I+};
            ok:= (IOresult = 0);
            if not ok then
               begin
               clrscr;
               gotoxy (14,14);
               writeln ('Cannot find the file ',infilename);
               end;
         until ok;
      end;
      {---ending of the procedure openinfile---}
```

```
{-------------------------------------------------------------------------
--}

   {>>> procedure OpenOutFile : This procedure opens the output file. <<<}

   procedure OpenOutFile (var gen,
                              vis,
                              mrm,
                              skd,
                              ssi,
                              scr : text;
                          filename  : file_str);

      {---beginning of the procedure openoutfile---}
      begin
         assign (gen, '\PAVEDB\PESGEN.DLM');
         rewrite (gen);
         assign (vis, '\PAVEDB\PESVIS.DLM');
         rewrite (vis);
         assign (mrm, '\PAVEDB\PESMRM.DLM');
         rewrite (mrm);
         assign (skd, '\PAVEDB\PESSKD.DLM');
         rewrite (skd);
         assign (ssi, '\PAVEDB\PESSSI.DLM');
         rewrite (ssi);
         assign (scr, '\PAVEDB\PESSCR.DLM');
         rewrite (scr);
      end;
      {---ending of the procedure openoutfile---}


{-------------------------------------------------------------------------
--}

   {>>> procedure OpenReportFile: This procedure asks the user for the name
<<<}
   {>>>                          of the report file and open it for output.
<<<}

   procedure OpenReportFile (var report_file, report_data_file: text;
filename: file_str);
      {---variable declarations---}
      var
         reportfilename: string[30];
         reportdatafile: string[30];
      {---beginning of the procedure openreportfile---}
      begin
         reportfilename:= filename + '.rep';
         assign (report_file, reportfilename);
         rewrite (report_file);
         reportdatafile:= filename + '.rpd';
         assign (report_data_file, reportdatafile);
         rewrite (report_data_file);
```

```
      end;
      {---ending of the procedure openreportfile---}

{-------------------------------------------------------------------------
--}

   {>>> procedure ProduceReport: This procedure prodeuces the missing data
<<<}
   {>>>                          report using the report data file.
<<<}

   procedure ProduceReport (var report_file, report_data_file: text);
      var
         record_line: integer;
         data_line:   string[132];
      begin
         for i:= 1 to 132 do data_line[i]:= ' ';
         close (report_data_file);
         reset (report_data_file);
         record_line:= 0;
         while not eof (report_data_file) do
         begin
            if (record_line = 0) or (record_line = 50) then
            begin
               record_line:= 0;
               {>>> write the heading for the report <<<}
               writeln (report_file, ^L);
               writeln (report_file,'                                Texas Flexible
Pavement Database');
               writeln (report_file,'                                    PES
Modification Program');
               writeln (report_file,'                                       Missing
Data Report');
               writeln(report_file);
               writeln(report_file);
               writeln (report_file, '          Hwy  Hwy Cnty          Begin
   Hwy          ---Missing data----');
               writeln (report_file, '          Dist Sec Num  Highway  Mile
pt.  Des Laneset Vis MRM Skd SSI Scr');
               writeln (report_file, '
_____________________________________________________________');
            end;
            readln (report_data_file, data_line);
            writeln (report_file, data_line);
            record_line:= record_line + 1;
         end;
      end;

{-------------------------------------------------------------------------
--}


   {>>> beginning of the program <<<}
   begin
```

```
      for i:= 1 to 15 do read (in_file, lane[i], si[i]);

{--- read skid data ---}
read (in_file, skidupdat, skidavgsn);
for ls:= rv to xz do
   with skid_data[ls] do
      for i:= 1 to 15 do read (in_file, lane[i], sn[i]);

{--- read some more general data ---}
read (in_file, surfwidth, hwydesign);
read (in_file, smplflgman, smplflgoth);

{--- read ssi data ---}
for ls:= rv to xz do
   with ssi_gen_data[ls] do
      read (in_file, date, avg, temp);

for ls:= rv to xz do
   with ssi_data[ls] do
      for i:= 1 to 5 do
         begin
            read (in_file, lane[i], gph1[i], gph2[i]);
            read (in_file, gph3[i], gph4[i], gph5[i]);
            read (in_file, gph6[i], gph7[i]);
         end;

{--- read score data ---}
for ls:= rv to xz do
   with score_data[ls] do
      begin
         read (in_file, id, mrm, skid, mncost, uwvis);
         read (in_file, adjvis, wvis, uwpvmt, pvmt);
      end;

{--- goto the next line ---}
readln (in_file);
if records_checked = 999 then
   begin
   krecords_checked:= krecords_checked + 1;
   records_checked:= 0;
   end
else
   records_checked:= records_checked + 1;

{>>> only records with an 'A' in the Mandatory Sample Selection<<<}
{>>> Flag field and an 'E','F','G','H','I','N','O','P','Q' or  <<<}
{>>> 'R' in the Other Sample Selection Flag field should be    <<<}
{>>> processed.                                                <<<}
if (smplflgman = 'A') and  ((smplflgoth='E') or (smplflgoth='F')
   or (smplflgoth='G') or (smplflgoth='H') or (smplflgoth='I')
   or (smplflgoth='N') or (smplflgoth='O') or (smplflgoth='P')
   or (smplflgoth='Q') or (smplflgoth='R') )then
begin
```

```
{>>> let user know that the program is running <<<}
clrscr;
gotoxy (15,3);
write ('*****************************************************');
gotoxy (15,4);
write ('* Program PESMOD is running.                        *');
gotoxy (15,5);
write ('* This program will modifiy the PES data tape file. *');
gotoxy (15,6);
write ('*****************************************************');

{>>> assign input and output files <<<}
clrscr;
OpenInFile(in_file,filename);
OpenOutFile(gen_file,vis_file,mrm_file,skd_file,ssi_file,scr_file,
        filename);
OpenReportFile (report_file,report_data_file,filename);

{>>> initialize record counter <<<}
gotoxy (14,12);
write ('Processing...');
gotoxy (21,13);
write ('new modified records');
record_no:= 0;
record_no_pos:= 17;
krecords_checked:= 0;
records_checked:= 0;


while not eof (in_file) do
begin
{>>> read in data <<<}
{--- read some general data ---}
read (in_file, hwydist, hwysect, cntynum);
read (in_file, hwyprefx, hwynum, hwysuffx);
read (in_file, bmpnum, bmpdispsgn, bmpdispval);
read (in_file, empnum, empdispsgn, empdispval);
read (in_file, intpst1, intpst2, intdis1, intdis2);
read (in_file, rtcntybeg, rtcntyend);
read (in_file, seglngth, segupdat, segmncost);
read (in_file, adt, a18k, funclas, numlanes, spdlimit, rissurf);

{--- read visual data ---}
for i:= 1 to 6 do
   begin
      read (in_file, vislane[i], visdate[i], viscomm[i], vispvmt[i]);
      read (in_file, visfail[i], vistran[i], vislong[i], visrut[i]);
      read (in_file, visallg[i], visblck[i], visptch[i]);
   end;

{--- read mrm data ---}
read (in_file, mrmupdat, mrmavgsi);
for ls:= rv to xz do
   with mrm_data[ls] do
```

```
{>>> check and see what kind of data is available <<<}
{>>> for the different lanes in the four lanesets <<<}

{--- if highway design number is 1 or 2, ---}
{--- laneset rv and lp is considered the same ---}
if (hwydesign = '1') or (hwydesign='2') then
   begin_ls:= lp
else
   begin_ls:= rv;

{--- for each laneset, check to see what data is available ---}
for ls:= begin_ls to xz do
   begin
      { if data is available, assign the set_flag to be true }
      mrm_set_flag:= false;
      skid_set_flag:= false;
      ssigen_set_flag:= false;
      ssi_set_flag:= false;
      score_set_flag:= false;
      vis_set_flag:= false;

      if mrm_data[ls].lane[1] <> ' ' then mrm_set_flag:= true;
      if skid_data[ls].lane[1] <> ' ' then skid_set_flag:= true;
      if ssi_gen_data[ls].date<>'000000' then ssigen_set_flag:= true;
      if ssi_data[ls].lane[1] <> ' ' then ssi_set_flag:= true;
      if score_data[ls].id <> '   ' then score_set_flag:= true;
      vis_count:= 1;
      while (not vis_set_flag) and (vis_count <= 6) do
         begin
            if (vislane[vis_count] <> ' ') then
               vis_set_flag:= true;
            vis_count:= vis_count + 1;
         end;


      { when there is some data for the laneset, proceed to write }
      { records for the lanes in the laneset that have data       }
      if mrm_set_flag or skid_set_flag or ssigen_set_flag
         or ssi_set_flag or score_set_flag or vis_set_flag
      then begin
         { assign the correct lane range of the present laneset }
         if (hwydesign='1') or (hwydesign='2') then
            case ls of
               lp: l:= ['R','L'];
               ac: l:= ['A'..'C'];
               xz: l:= ['X'..'Z'];
            end
         else
            case ls of
               rv: l:= ['R'..'V'];
               lp: l:= ['L'..'P'];
               ac: l:= ['A'..'C'];
               xz: l:= ['X'..'Z'];
            end;
```

```
{ for each laneset, write a record if data exists }

{ if data is available, assign the ln_flag to be true }
mrm_ln_flag:= false;
skid_ln_flag:= false;
ssi_ln_flag:= false;
vis_ln_flag:= false;

if mrm_set_flag and (mrm_data[ls].lane[1] in l)
then mrm_ln_flag := true;
if skid_set_flag and (skid_data[ls].lane[1] in l)
then skid_ln_flag := true;
if ssi_set_flag and (ssi_data[ls].lane[1] in l)
then ssi_ln_flag := true;
{ This is assuming that there will only be }
{ a set of visual data for one lane per laneset. }
{ Meaning that there should not be data for }
{ lane 'R' and 'S'. Otherwise, only the first one}
{ will be picked up. }
vis_count:= 1;
visln_pos:= 0;
while (visln_pos = 0) and (vis_count <= 6) do
   begin
      if (vislane[vis_count] in l) then
         visln_pos:= vis_count;
      vis_count:= vis_count + 1;
   end;
if visln_pos <> 0 then vis_ln_flag:= true;

{ when there is some data for the laneset, proceed to write }
{ out the record                                            }
if mrm_ln_flag or skid_ln_flag or ssi_ln_flag
   or vis_ln_flag then
   begin
      {check for the laneset}
      if (hwydesign = '1') or (hwydesign = '2') then
         case ls of
            lp: lane_set:= 'R-L';
            ac: lane_set:= 'A-C';
            xz: lane_set:= 'X-Z';
         end
      else
         case ls of
            rv: lane_set:= 'R-V';
            lp: lane_set:= 'L-P';
            ac: lane_set:= 'A-C';
            xz: lane_set:= 'X-Z';
         end;
      { write out some key data }
      write (gen_file,
      hwydist,comma,          hwysect,comma,
      cntynum,comma,          '"',hwyprefx,'"',comma,
      hwynum,comma,           '"',hwysuffx,'"',comma,
```

```
bmpnum,comma,          '"',bmpdispsgn,'"',comma,
bmpdispval,comma);
write (gen_file, '"',lane_set,'"',comma);

write (vis_file,
hwydist,comma,         hwysect,comma,
cntynum,comma,         '"',hwyprefx,'"',comma,
hwynum,comma,          '"',hwysuffx,'"',comma,
bmpnum,comma,          '"',bmpdispsgn,'"',comma,
bmpdispval,comma);
write (vis_file,'"', lane_set,'"',comma);

write (mrm_file,
hwydist,comma,         hwysect,comma,
cntynum,comma,         '"',hwyprefx,'"',comma,
hwynum,comma,          '"',hwysuffx,'"',comma,
bmpnum,comma,          '"',bmpdispsgn,'"',comma,
bmpdispval,comma);
write (mrm_file,'"', lane_set,'"',comma);

write (skd_file,
hwydist,comma,         hwysect,comma,
cntynum,comma,         '"',hwyprefx,'"',comma,
hwynum,comma,          '"',hwysuffx,'"',comma,
bmpnum,comma,          '"',bmpdispsgn,'"',comma,
bmpdispval,comma);
write (skd_file,'"', lane_set,'"',comma);

write (ssi_file,
hwydist,comma,         hwysect,comma,
cntynum,comma,         '"',hwyprefx,'"',comma,
hwynum,comma,          '"',hwysuffx,'"',comma,
bmpnum,comma,          '"',bmpdispsgn,'"',comma,
bmpdispval,comma);
write (ssi_file,'"', lane_set,'"',comma);

write (scr_file,
hwydist,comma,         hwysect,comma,
cntynum,comma,         '"',hwyprefx,'"',comma,
hwynum,comma,          '"',hwysuffx,'"',comma,
bmpnum,comma,          '"',bmpdispsgn,'"',comma,
bmpdispval,comma);
write (scr_file,'"', lane_set,'"',comma);


{ write out some general data }
write (gen_file, empnum,comma,
'"',empdispsgn,'"',comma,empdispval,comma);
if (intpst1 = '   ') then
   write (gen_file, zero:3,comma)
else
   write (gen_file, intpst1,comma);
if (intpst2 = '   ') then
   write (gen_file, zero:3,comma)
```

```
else
   write (gen_file, intpst2,comma);
if (intdis1 = '  ') then
   write (gen_file, zero,decpt,zero,comma)
else
   write (gen_file, copy(intdis1,1,1),decpt,
                    copy(intdis1,2,1),comma);
if (intdis2 = '  ') then
   write (gen_file, zero,decpt,zero,comma)
else
   write (gen_file, copy(intdis2,1,1),decpt,
                    copy(intdis2,2,1),comma);
write (gen_file,
copy(rtcntybeg,1,3),decpt,copy(rtcntybeg,4,1),comma,
copy(rtcntyend,1,3),decpt,copy(rtcntyend,4,1),comma,
copy(seglngth,1,1),decpt,copy(seglngth,2,1),comma,
segupdat,comma,
copy(segmncost,1,7),decpt,copy(segmncost,8,2),comma,
adt,comma,              a18k,comma,
funclas,comma,          numlanes,comma,
spdlimit,comma,         rissurf,comma);

{ write out the visual data}
if vis_ln_flag then
   begin { for existing data }
      write(vis_file,
      '"',vislane[visln_pos],'"',comma,
      visdate[visln_pos],comma,
      viscomm[visln_pos],comma,
      vispvmt[visln_pos],comma,
      visfail[visln_pos],comma,
      vistran[visln_pos],comma,
      vislong[visln_pos],comma,
      visrut[visln_pos],comma,
      visallg[visln_pos],comma,
      visblck[visln_pos],comma,
      visptch[visln_pos]);
   end
else
   begin { for empty fields }
      write(vis_file, '"',blank,'"');
      for j:= 1 to 10 do write(vis_file,comma,zero);
   end;
writeln (vis_file);

{ write out mrm data }
write(mrm_file, mrmupdat,comma,copy(mrmavgsi,1,1),
                decpt,copy(mrmavgsi,2,1),comma);
if mrm_ln_flag then
   for i:= 1 to 15 do { for existing data }
      write (mrm_file, '"',mrm_data[ls].lane[i],'"',
             comma,copy(mrm_data[ls].si[i],1,1),decpt,
             copy(mrm_data[ls].si[i],2,1),comma)
else
```

```
   for i:= 1 to 15 do { for empty fields }
      write (mrm_file, '"',blank,'"',comma,
             zero,comma);

{ write out skid data }
write (skd_file, skidupdat,comma,skidavgsn,comma);
if skid_ln_flag then
   for i:= 1 to 15 do { for existing data }
      write (skd_file, '"',skid_data[ls].lane[i],'"',
             comma,skid_data[ls].sn[i],comma)
else
   for i:= 1 to 15 do { for empty fields }
      write (skd_file, '"',blank,'"',comma,
             zero,comma);

{ write out some more general data }
writeln (gen_file, surfwidth,comma,
                   hwydesign,comma,
                   '"',smplflgman,'"',comma,
                   '"',smplflgoth,'"');

{ write out ssi general data }
if ssigen_set_flag then
   with ssi_gen_data[ls] do { for existing data }
      write (ssi_file, date,comma,
                       avg,comma,
                       temp)
else { for empty fields }
   write(ssi_file,zero,comma,zero,comma,zero);

{ write out ssi data }
if ssi_ln_flag then
   with ssi_data[ls] do
   for i:= 1 to 5 do { for existing data }
      begin
      write(ssi_file, comma,'"',lane[i],'"');
      write (ssi_file,
      comma,copy(gph1[i],1,2),decpt,copy(gph1[i],3,2),
      comma,copy(gph2[i],1,2),decpt,copy(gph2[i],3,2),
      comma,copy(gph3[i],1,2),decpt,copy(gph3[i],3,2),
      comma,copy(gph4[i],1,2),decpt,copy(gph4[i],3,2),
      comma,copy(gph5[i],1,2),decpt,copy(gph5[i],3,2),
      comma,copy(gph6[i],1,2),decpt,copy(gph6[i],3,2),
      comma,copy(gph7[i],1,2),decpt,copy(gph7[i],3,2));
      end
else
   for i:= 1 to 5 do { for empty fields }
      begin
         write (ssi_file, comma,'"',blank,'"');
         for j:= 1 to 7 do
            write (ssi_file,comma,zero,decpt,zero);
      end;
writeln (ssi_file);
```

```
{ write out score data }
if score_set_flag then
   with score_data[ls] do { for existing data }
      write(scr_file, '"',id,'"',comma,
      copy(mrm,1,1),decpt,copy(mrm,2,2),comma,
      copy(skid,1,1),decpt,copy(skid,2,2),comma,
      copy(mncost,1,1),decpt,copy(mncost,2,2),comma,
      copy(uwvis,1,1),decpt,copy(uwvis,2,2),comma,
      copy(adjvis,1,1),decpt,copy(adjvis,2,2),comma,
      copy(wvis,1,1),decpt,copy(wvis,2,2),comma,
      copy(uwpvmt,1,1),decpt,copy(uwpvmt,2,2),comma,
      copy(pvmt,1,1),decpt,copy(pvmt,2,2))
else
   begin { for empty fields }
      write (scr_file,'"',blank,blank,blank,'"');
      for i:= 1 to 8 do
         write (scr_file,comma,zero,decpt,zero);
   end;
writeln (scr_file);


{--- calculate the statistics on si values ---}
sicount := 0;
sihi:= 0.0;
silo:= 0.0;
simean:= 0.0;
sisd:= 0.0;
if mrm_ln_flag then
begin
   with mrm_data[ls] do
      for i:= 1 to 15 do
         begin
            val (si[i], temp_si, err_code);
            mrm_stat_array[i]:= temp_si/10.0;
         end;
   stat(mrm_stat_array,item_array_size,sihi,
      silo,simean,sisd,sicount);
end;

{ write out the mrm stat. }
writeln (mrm_file, simean:7:5,comma,sisd:7:5,comma,
                   sihi:3:1,comma,
                   silo:3:1,comma,sicount:3);

{--- calculate the statistics on skid values ---}
skidcount:= 0;
skidhi:= 0;
skidlo:= 0;
skidmean:= 0;
skidsd:= 0;
if skid_ln_flag then
begin
   with skid_data[ls] do
      for i:= 1 to 15 do
```

```
            begin
               val (sn[i], temp_sn, err_code);
               skid_stat_array[i]:= temp_sn;
            end;
      stat ( skid_stat_array,item_array_size,skidhi,
             skidlo,skidmean,skidsd,skidcount);
end;

{ write out the skid stat. }
writeln (skd_file, skidmean:2:0,comma,skidsd:2:0,comma,
                   skidhi:2:0,comma,skidlo:2:0,comma,
                   skidcount:2);

{ write out the missing data report for lanes }
write (report_data_file,'           ');
write (report_data_file, hwydist,'  ',
hwysect,'  ',
cntynum,'  ',
hwyprefx,' ',hwynum,' ',hwysuffx,' ',
bmpnum,' ',bmpdispsgn,' ',bmpdispval,'  ',
hwydesign,'    ',
lane_set,'    ');
if vis_ln_flag then
   write (report_data_file, ' ')
else
   write (report_data_file, 'X');
write (report_data_file, '   ');
if mrm_ln_flag then
   write (report_data_file, ' ')
else
   write (report_data_file, 'X');
write (report_data_file, '   ');
if skid_ln_flag then
   write (report_data_file, ' ')
else
   write (report_data_file, 'X');
write (report_data_file, '   ');
if ssi_ln_flag then
   write (report_data_file, ' ')
else
   write (report_data_file, 'X');
write (report_data_file, '   ');
if score_set_flag then
   write (report_data_file, ' ')
else
   write (report_data_file, 'X');
writeln (report_data_file);

{--- increment the record counter ---}
if (record_no < 999) then
   begin
   gotoxy (record_no_pos,15);
   record_no:= record_no+1;
   write (record_no:3);
```

```
                    end
                 else
                    begin
                    gotoxy (record_no_pos-3,15);
                    krecord_no:= krecord_no +1;
                    write (krecord_no:3,'000');
                    record_no:= 0;
                    end;
              end; {then}

        end {then}
        else
        begin
           { write out the missing data report for lanes }
           write (report_data_file, '          ');
            writeln (report_data_file, hwydist,'   ',
                 hwysect,' ',
                 cntynum,' ',
                 hwyprefx,' ',hwynum,' ',hwysuffx,' ',
                 bmpnum,' ',bmpdispsgn,' ',bmpdispval,'  ',
                 hwydesign,'    ',
                 '---','    ','X   X   X   X   X');

        end; {else}
     end; {for}
  end; {if}
  end; {while}

  ProduceReport (report_file, report_data_file);

  {>>> inform user that the process is done. <<<}
  gotoxy (45,13);
  write (krecords_checked:3,records_checked:3, ' PES records checked');
  gotoxy (14,14);
  write ('(NOTE: Missing data for PES records is reported in file ');
  gotoxy (14,15);
  write ('       ',filename,'.REP.)');
  gotoxy (14,16);
  write ('Done.');

  {>>> close input/ouput files <<<}
  close (in_file);
  close (gen_file);
  close (vis_file);
  close (mrm_file);
  close (skd_file);
  close (ssi_file);
  close (scr_file);
  close (report_file);
  close (report_data_file);

end.
```

PES Update Program Narrative

This program is called after PESMOD.PAS is run. PESMOD.PAS creates the six comma delimited files that PESUP.PRG uses. These six files are appended to six dBASE temporary files (PESSKD.DBF, PESMRM.DBF, PESSSI.DBF, PESVISL.DBF, PESSCR.DBF, PESGEN.DBF). The program then checks to see if the same dat ahas been added previously. If the data is already present in the Master Monitoring files, the program terminates itself. Otherwise, it adds data to the Master Monitoring files (Skid, Visual, Serviceability Index and Falling Weight) and the Location File.

The PESUP.PRG program checks to see if all the PES comma delimited files and PES temporary dBASE files are present before continuing further. An error report is produced for those SID numbers thaqt do not have a matching PES record and those that have inconsistent data for Functional Classification, Number of Lanes, Ending Milepost and No Surface Layer. These programs and the temporary dBase files are in the subdirectory \PAVEDB\EDITUPDT\PES.

PES UPDATE – PROGRAM FLOW DIAGRAM

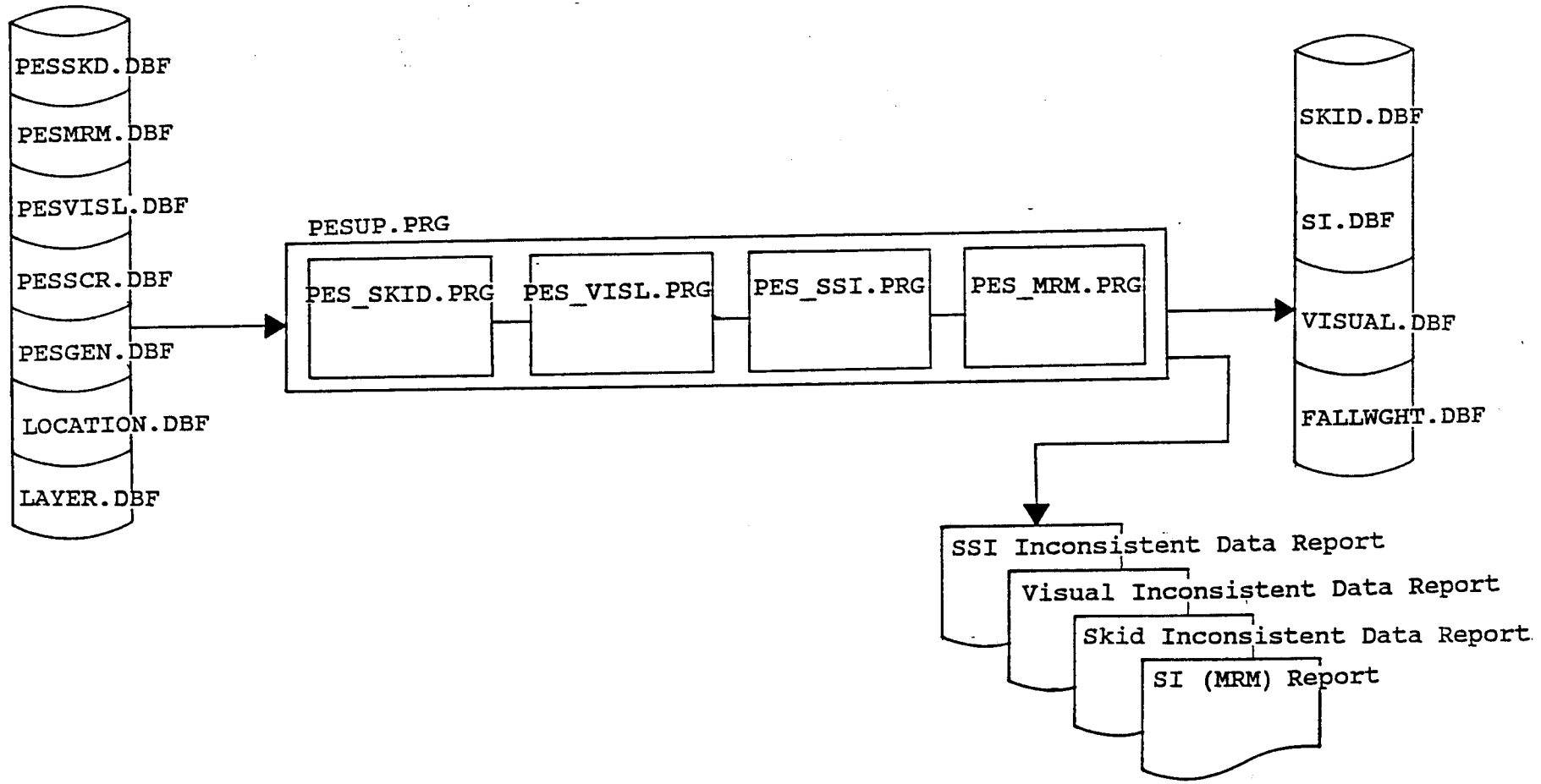


FIGURE 12

PES Database Update Program
Program Specification

Program Name: PESUP.PRG

Purpose: Update the database with monitoring data from the PES File

Input File:

1) Modified PES temporary dBASE Files -
   - Skid (PESSKD.DBF)
   - Serviceability Index (PESMRM.DBF)
   - Falling Weight SSI (PESSSI.DBF)
   - Visual (PESVISL.DBF)
   - Scores (PESSCR.DBF)
   - Other Data (PESGEN.DBF)
2) dBASE III Location File (LOCATION.DBF)
3) dBASE III Layer ID File (LAYER.DBF)

Procedures/Edits:

See the Procedures pages.

Output Files:

Note: The following files are all dBASE III Files
1) SKID.DBF - Skid Measurement File
2) SI.DBF - Serviceability Index File
3) VISUAL.DBF - Visual Rating File
4) FALLWGHT.DBF - Falling Weight SSI File
5) LOCATION.DBF - Location File

dBASE Programs Called:
PES_VISL.PRG
PES_MRM.PRG
PES_SSI.PRG
PES_SKID.PRG

Report: Missing and Inconsistent Data - see the Report Layout.

Note: The procedures pages are followed by the modified PES dBASE temporary file layouts and the Missing and Inconsistent Data Report Layout.

PROCEDURES

1. New records are created for each active test section in the LOCATION File for the SKID, SI, SSI, and VISUAL files.

2. PESUPD.PRG matchs the Modified PES File Records with the appropriate SID records using the LOCATION File. The files are matched to the Highway System, Number, Suffix, Beginning Milepoint Number, Beginning Milepoint Displacement Sign, Beginning Milepoint Displacement, and the Lane ID-Laneset.
   A LANEID of 'R' matches the LANESET 'R-L' and 'R-V'.
   A LANEID of 'L' matches the LANESET 'R-L' and 'L-P'

3. For each type of monitoring data, the monitoring date is checked against the LAYER file to obtain the appropriate Structure Number and Layer Number for the new records. The Layer Number used is the last existing layer at the time of monitoring. The Structure Number corresponding to the chosen Layer Number is used.

4. For the SKID, SSI, and SI data, the information is used as it exists in the Modified PES File (The dates, however, are split into month, day, and year fields).

5. The VISUAL File data requires some mapping according to the following criteria:

PES		Texas Flexible Pavement Database		
Failures		FAILMILE		
0 0 0	---------->	0		
1 0 0	---------->	1		
0 1 0	---------->	2		
0 0 1	---------->	3		
Transverse Cracks		TRANCRSL	TRANCRMD	TRANCRSV
0 0 0	---------->	0	0	0
1 0 0	---------->	0	1	0
0 1 0	---------->	0	2	0
0 0 1	---------->	0	3	0
Longitudinal Cracks		LONGCRSL	LONGCRMD	LONGCRSV
0 0 0	---------->	0	0	0
1 0 0	---------->	0	1	0
0 1 0	---------->	0	2	0
0 0 1	---------->	0	3	0

5. (cont.)

PES		Texas Flexible Pavement Database		
Rutting		RUTTSL	RUTTMD	RUTTSV
0 0 0	---------->	0	0	0
1 0 0	---------->	0	1	0
0 1 0	---------->	0	2	0
0 0 1	---------->	0	3	0
2 0 0	---------->	0	0	1
0 2 0	---------->	0	0	2
0 0 2	---------->	0	0	3
Alligator Cracks		ALLGCRSL	ALLGCRMD	ALLGCRSV
0 0 0	---------->	0	0	0
1 0 0	---------->	0	1	0
0 1 0	---------->	0	2	0
0 0 1	---------->	0	3	0
Block Cracks			BLKCRKMD	
0 0 0	---------->		0	
1 0 0	---------->		1	
0 1 0	---------->		2	
0 0 1	---------->		3	
Maintenance Patching		PATCHGD	PATCHFR	PATCHPR
0 0 0	---------->	0	0	0
1 0 0	---------->	0	1	0
0 1 0	---------->	0	2	0
0 0 1	---------->	0	3	0

6. PESUPD.PRG reports:

   a) Active test sections which do not have matching PES Records.

   b) The SID Number when the number of lanes is not the same on the PES and the Texas Flexible Pavement Database Files.

   c) The SID number when the Ending Milepoint, Ending Milepoint Displacement Sign, and Ending Milepoint Displacement are not the same on the PES and the Texas Flexible Pavement Database Files.

   d) The SID number if the Functional Classification is not the same on the PES and the Texas Flexible Pavement Database Files.

   e) The SID number if the Surface layer is absent in the Layer Identification file.

PES-Skid File Layout

File Name: PESSKD.DBF

Field	Key	Size/Type	Description
HWYDIST	*	2N	SDHPT Highway District
CNTYNUM	*	2N	County Number
HWYPREFX	*	2C	Highway Number Preffix
HWYNUM	*	4N	Highway Number
HWYSUFFX	*	1C	Highway Number Suffix
BEGMPST	*	3N	Beginning Milepost
BDISSIGN	*	1C	Beginning Displacement Sign
BMPSTDIS	*	2N	Beginning Milepost Displacement
LANESET	*	3C	Lane Set
LASTUPDT		6N	Last Update (YYMMDD)
SKIDNUMN		2N	Skid - Mean value
SKIDSTDD		2N	Skid - Standard Deviation
SKIDNUMH		2N	Skid - High Value
SKIDNUML		2N	Skid - Low Value
SKDCNTOB		2N	Skid - Count of Observation

PES-Serviceability Index File Layout

File Name: PESMRM.DBF

Field	Key	Size/Type	Description
HWYDIST	*	2N	SDHPT Highway District
CNTYNUM	*	2N	County Number
HWYPREFX	*	2C	Highway Number Preffix
HWYNUM	*	4N	Highway Number
HWYSUFFX	*	1C	Highway Number Suffix
BEGMPST	*	3N	Beginning Milepost
BDISSIGN	*	1C	Beginning Displacement Sign
BMPSTDIS	*	2N	Beginning Milepost Displacement
LANESET	*	3C	Lane Set
LASTUPDT		6N	Last Update (YYMMDD)
SIMEAN		7.5N	Serviceability Index - Mean
SISTDDEV		7.5N	Serviceability Index - Standard Deviation
SIHIVAL		3.1N	Serviceability Index - High Value
SILOWVAL		3.1N	Serviceability Index - Low Value
SICOUNT		3N	Serviceability Index - Count of Observation

PES-Falling Weight SSI File Layout

File Name: PESSSI.DBF

Field	Key	Size/Type	Description
HWYDIST	*	2N	SDHPT Highway District
CNTYNUM	*	2N	County Number
HWYPREFX	*	2C	Highway Number Preffix
HWYNUM	*	4N	Highway Number
HWYSUFFX	*	1C	Highway Number Suffix
BEGMPST	*	3N	Beginning Milepost
BDISSIGN	*	1C	Beginning Displacement Sign
BMPSTDIS	*	2N	Beginning Milepost Displacement
LANESET	*	3C	Lane Set
SSIAVG		4.1N	Average SSI for the Roadway
SSITEMP		3N	Temperature For the Roadway
GEOPHON11		5.2N	SSI Reading 1 Geophone no. 1
GEOPHON12		5.2N	SSI Reading 1 Geophone no. 2
GEOPHON13		5.2N	SSI Reading 1 Geophone no. 3
GEOPHON14		5.2N	SSI Reading 1 Geophone no. 4
GEOPHON15		5.2N	SSI Reading 1 Geophone no. 5
GEOPHON16		5.2N	SSI Reading 1 Geophone no. 6
GEOPHON17		5.2N	SSI Reading 1 Geophone no. 7
GEOPHON21		5.2N	SSI Reading 2 Geophone no. 1
GEOPHON22		5.2N	SSI Reading 2 Geophone no. 2
GEOPHON23		5.2N	SSI Reading 2 Geophone no. 3
GEOPHON24		5.2N	SSI Reading 2 Geophone no. 4
GEOPHON25		5.2N	SSI Reading 2 Geophone no. 5
GEOPHON26		5.2N	SSI Reading 2 Geophone no. 6
GEOPHON27		5.2N	SSI Reading 2 Geophone no. 7
GEOPHON31		5.2N	SSI Reading 3 Geophone no. 1
GEOPHON32		5.2N	SSI Reading 3 Geophone no. 2
GEOPHON33		5.2N	SSI Reading 3 Geophone no. 3
GEOPHON34		5.2N	SSI Reading 3 Geophone no. 4
GEOPHON35		5.2N	SSI Reading 3 Geophone no. 5
GEOPHON36		5.2N	SSI Reading 3 Geophone no. 6
GEOPHON37		5.2N	SSI Reading 3 Geophone no. 7
GEOPHON41		5.2N	SSI Reading 4 Geophone no. 1

Pes-Visual File Layout

File Name: PESVISL.DBF

Field	Key	Size/Type	Description
HWYDIST	*	2N	SDHPT Highway District
CNTYNUM	*	2N	County Number
HWYPREFX	*	2C	Highway Number Preffix
HWYNUM	*	4N	Highway Number
HWYSUFFX	*	1C	Highway Number Suffix
BEGMPST	*	3N	Beginning Milepost
BDISSIGN	*	1C	Beginning Displacement Sign
BMPSTDIS	*	2N	Beginning Milepost Displacement
LANESET	*	3C	Lane Set
DATE		6N	Date (YYMMDD)
FAILMILE		3N	Failures per Mile Occurance Rating
TRANCR		3N	Transverse Crack Area Rating
LONGCR		3N	Longitudinal Crack Area Rating
RUTT		3N	Rutting Area Rating
ALLGCR		3N	Alligator Crack Area Rating
BLKCR		3N	Block Crack Area Rating
PATCH		3N	Patching Area Rating

PES-Falling Weight SSI File Layout (continued)

Field	Key	Size/Type	Description
GEOPHON42		5.2N	SSI Reading 4 Geophone no. 2
GEOPHON43		5.2N	SSI Reading 4 Geophone no. 3
GEOPHON44		5.2N	SSI Reading 4 Geophone no. 4
GEOPHON45		5.2N	SSI Reading 4 Geophone no. 5
GEOPHON46		5.2N	SSI Reading 4 Geophone no. 6
GEOPHON47		5.2N	SSI Reading 4 Geophone no. 7
GEOPHON51		5.2N	SSI Reading 5 Geophone no. 1
GEOPHON52		5.2N	SSI Reading 5 Geophone no. 2
GEOPHON53		5.2N	SSI Reading 5 Geophone no. 3
GEOPHON54		5.2N	SSI Reading 5 Geophone no. 4
GEOPHON55		5.2N	SSI Reading 5 Geophone no. 5
GEOPHON56		5.2N	SSI Reading 5 Geophone no. 6
GEOPHON57		5.2N	SSI Reading 5 Geophone no. 7

PES Update Program
Score Information File Record Layout

File Name: PESSCR.DBF

Field	Key	Size/ Type	Description
HWYDIST	*	2N	Highway District Number
MAINTSEC	*	2N	Maintenance Section Number
CNTYNUM	*	3N	County Number
HWYPREFX	*	2C	Highway System
HWYNUM	*	4N	Highway Number
HWYSUFFX	*	1C	Highway Suffix
BEGMPST	*	3N	Beginning Milepost Number
BDISSIGN	*	1C	Beginning Milepost Displacement Sign
BMPSTDIS	*	3.1N	Beginning Milepost Displacement Value
LANESET	*	3C	Laneset
SCRIDENT		3C	Identifier Score
SSIUTIL		4.2N	MRM Utility Score
SKIDUTIL		4.2N	Skid Utility Score
MAINTCST		4.2N	Maintenance Cost Score
UVURS		4.2N	Unweighted Visual Score
ADJSVISL		4.2N	Adjusted Visual Score
WGHTVISL		4.2N	Weighted Visual Score
UNWGHTPMT		4.2N	Unweighted Pavement Score
PVMTSCR		4.2N	Pavement Score

* Key Field

PES Update Program
General Information File Record Layout (PESGEN.DBF)

File Name: PESGEN.DBF

Field	Key	Size/ Type	Description
HWYDIST	*	2N	Highway District Number
MAINTSEC	*	2N	Maintenance Section Number
CNTYNUM	*	3N	County Number
HWYPREFX	*	2C	Highway System
HWYNUM	*	4N(C)	Highway Number
HWYSUFFX	*	1C	Highway Suffix
BEGMPST	*	3N	Segment Begin - Milepost No
BDISSIGN	*	1C	- Displacement Sign
BMPSTDIS	*	3.1N	- Displacement Value
LANESET	*	3C	Laneset
ENDMPST		3N	Segment End - Milepost No.
EDISSIGN		1C	- Displacement Sign
EMPSTDIS		3.1N	- Displacement Value
POST_1		3C/N	Intermediate Post- 1st Post
POST_2		3C/N	- 2nd Post
DISTTO1		3.1N	- Distance to 1st
DISTTO2		3.1N	- Distance to 2nd
BEGVAL		5.1N	Route-Thru-County - Begin Value
ENDVAL		5.1N	- End Value
SEGLENTH		3.1N	Segment Length
SEGLSTUP		6N	Last Update Date (YYMMDD)
SEGMAINT		10.2N	Segment Maintenance Cost
AVGDLTRF		6N	Average Daily Traffic
KIP18EQ		5N	18-Kip Equivalency
FUNCLAS		1N	Functional Classification
NOLANES		2N	Number of lanes
SPEEDLMT		2N	Speed Limit
RISURFTY		2N	RIS Surface Type
SEGSURWD		3N	Surface Width
HWYDSNTY		1N	Highway Design Type
SMPLSELM		1C	Sample Selection Flag - Mandatory
SMPLSELO		1C	- Other

* Key Field

PES Database Update Program
Missing and Inconsistent Data Report Layout

MM/DD/YY Texas Flexible Pavement Database

PES Database Update Program
Missing and Inconsistent Data Report Layout

		INCONSISTENT DATA			
SID Number	No PES Record	Number Of Lanes	Ending Milepoint	Functional Classification	No Surface Layer
XXXX	X	X	X	X	X

Note: An 'X' is placed in the appropriate column(s) depending on whether or not there is a matching PES File record, or the Number of Lanes, the Ending Milepoint information, or the Functional Classification do not agree between the PES and Texas Flexible Pavement Database File.

Edit & Update Subsystem
Program Listing

```
*
* SUBSYSTEM:      EDIT & UPDATE MONITORING DATA
* PROGRAM NAME:   PESUP.PRG          07/08/88
* PROJECT 2456 - TEXAS FLEXIBLE PAVEMENT DATABASE CONVERSION
* TAMU/TTI
* AUTHOR:         TREVOR X. PEREIRA
* PURPOSE:        CALL THE RESPECTIVE PROGRAMS TO ADD DATA TO THE
*                 MONITORING FILES
*

CLEAR
* checks to see if all the files are present
? "Checking files . . ."
IF .NOT. FILE('\PAVEDB\FILES\PESSKD.DBF')
   ? "PES SKID FILE (PESSKD.DBF) not found.  Please Check . . ."
   WAIT
   RETURN
ENDIF
IF .NOT. FILE('\PAVEDB\INDEXES\PESSKD.NDX')
   ? "PES SKID INDEX (PESSKD.NDX) not found.  Please Check . . ."
   WAIT
   RETURN
ENDIF
IF .NOT. FILE('\PAVEDB\FILES\PESMRM.DBF')
   ? "PES SI FILE (PESMRM.DBF) not found.  Please Check . . ."
   WAIT
   RETURN
ENDIF
IF .NOT. FILE('\PAVEDB\INDEXES\PESMRM.NDX')
   ? "PES SI INDEX (PESMRM.NDX) not found.  Please Check . . ."
   WAIT
   RETURN
ENDIF
IF .NOT. FILE('\PAVEDB\FILES\PESSSI.DBF')
   ? "PES SSI FILE (PESSSI.DBF) not found.  Please Check . . ."
   WAIT
   RETURN
ENDIF
IF .NOT. FILE('\PAVEDB\INDEXES\PESSSI.NDX')
   ? "PES SSI INDEX (PESSSI.NDX) not found.  Please Check . . ."
   WAIT
   RETURN
ENDIF
IF .NOT. FILE('\PAVEDB\FILES\PESVISL.DBF')
   ? "PES VISUAL FILE (PESVISL.DBF) not found.  Please Check . . ."
   WAIT
   RETURN
ENDIF
IF .NOT. FILE('\PAVEDB\INDEXES\PESVISL.NDX')
```

```
   ? "PES VISUAL INDEX (PESVISL.NDX) not found.  Please Check . . ."
   WAIT
   RETURN
ENDIF
IF .NOT. FILE('\PAVEDB\EDITUPDT\PES\PESTEMP1.DBF')
   ? "PES TEMPORARY FILE (PESTEMP1.DBF) not found.  Please Check . . ."
   WAIT
   RETURN
ENDIF
IF .NOT. FILE('\PAVEDB\EDITUPDT\PES\PESTEMP2.DBF')
   ? "PES TEMPORARY FILE (PESTEMP2.DBF) not found.  Please Check . . ."
   WAIT
   RETURN
ENDIF
IF .NOT. FILE('\PAVEDB\EDITUPDT\PES\PESTEMP3.DBF')
   ? "PES TEMPORARY FILE (PESTEMP3.DBF) not found.  Please Check . . ."
   WAIT
   RETURN
ENDIF
IF .NOT. FILE('\PAVEDB\EDITUPDT\PES\PESTEMP4.DBF')
   ? "PES TEMPORARY FILE (PESTEMP4.DBF) not found.  Please Check . . ."
   WAIT
   RETURN
ENDIF

@ 10, 10 SAY "Please wait.  Setting up files . . ."
SET SAFETY OFF
USE \PAVEDB\FILES\PESSKD INDE \PAVEDB\INDEXES\PESSKD
  ZAP
  APPEND FROM \PAVEDB\EDITUPDT\PES\PESSKD.DLM DELIMITED
USE \PAVEDB\FILES\PESGEN INDE \PAVEDB\INDEXES\PESGEN
  ZAP
  APPEND FROM \PAVEDB\EDITUPDT\PES\PESGEN.DLM DELIMITED
USE \PAVEDB\FILES\PESVISL INDE \PAVEDB\INDEXES\PESVISL
  ZAP
  APPEND FROM \PAVEDB\EDITUPDT\PES\PESVIS.DLM DELIMITED
USE \PAVEDB\FILES\PESMRM INDE \PAVEDB\INDEXES\PESMRM
  ZAP
  APPEND FROM \PAVEDB\EDITUPDT\PES\PESMRM.DLM DELIMITED
USE \PAVEDB\FILES\PESSSI INDE \PAVEDB\INDEXES\PESSSI
  ZAP
  APPEND FROM \PAVEDB\EDITUPDT\PES\PESSSI.DLM DELIMITED
USE \PAVEDB\FILES\PESSCR INDE \PAVEDB\INDEXES\PESSCR
  ZAP
  APPEND FROM \PAVEDB\EDITUPDT\PES\PESSCR.DLM DELIMITED

CLEAR
SET STAT ON
SET SAFETY OFF
@ 10, 10 SAY "Adding data to SKID file . . ."
* add data to the skid file
DO \PAVEDB\EDITUPDT\PES\PES_SKID

@ 12, 10 SAY "Adding data to VISUAL file . . ."
```

```
* add data to the visual file
DO \PAVEDB\EDITUPDT\PES\PES_VISL

@ 14, 10 SAY "Adding data to FALLING WEIGHT file . . ."
* add data to the Falling Weight SSI file
DO \PAVEDB\EDITUPDT\PES\PES_SSI

@ 16, 10 SAY "Adding data to SERVICEABILITY INDEX file . . ."
* add data to the Serviceability Index (SI) File
DO \PAVEDB\EDITUPDT\PES\PES_MRM

* Print out Inconsistent Data Report
USE \PAVEDB\EDITUPDT\PES\PESTEMP1
IF RECCOUNT() <> 0
    SORT ON SID_NO TO TEMP1
    USE TEMP1
    REPORT FORM \PAVEDB\REPORTS\PES_SKID TO PRINT
    EJECT
    USE
ENDIF
USE \PAVEDB\EDITUPDT\PES\PESTEMP2
IF RECCOUNT() <> 0
    SORT ON SID_NO TO TEMP1
    USE TEMP1
    REPORT FORM \PAVEDB\REPORTS\PES_MRM TO PRINT
    EJECT
    USE
ENDIF
USE \PAVEDB\EDITUPDT\PES\PESTEMP3
IF RECCOUNT() <> 0
    SORT ON SID_NO TO TEMP1
    USE TEMP1
    REPORT FORM \PAVEDB\REPORTS\PES_VISL TO PRINT
    EJECT
    USE TEMP1
ENDIF
USE \PAVEDB\EDITUPDT\PES\PESTEMP4
IF RECCOUNT() <> 0
    SORT ON SID_NO TO TEMP1
    USE TEMP1
    REPORT FORM \PAVEDB\REPORTS\PES_SSI TO PRINT
    EJECT
    USE
ENDIF

SET STAT OFF
close database
RETURN
```

Edit & Update Subsystem
Program Listing

```
*
* SUBSYSTEM:      EDIT & UPDATE SKID dBASE III FILE
* PROGRAM NAME:   PES_SKID.PRG          04/14/88
* MODIFIED ON:    09/19/88
* PROJECT 2456 - TEXAS FLEXIBLE PAVEMENT DATABASE CONVERSION
* TAMU/TTI
* AUTHOR:         TREVOR X. PEREIRA
* PURPOSE:        ADD THE PES DATA TO THE SKID FILE
*
* THE FOLLOWING FILES ARE USED BY THIS PROGRAM
*    LOCATION.DBF
*    LOCSID  .NDX
*    PESTEMP1.DBF (A temporary PES file to print out the Missing )
*                 (and Inconsistent Data Report)
*    SKID.DBF
*    SKID.NDX
*    LAYER   .DBF
*    LAYNDX  .NDX
*    PESSKD  .DBF
*    PESSKD  .NDX
*    PESGEN  .DBF
*    PESGEN  .NDX
*

SELECT 1
USE \PAVEDB\FILES\LOCATION
SELE 2
USE \PAVEDB\FILES\PESSKD INDEX \PAVEDB\INDEXES\PESSKD
SELE 3
USE \PAVEDB\EDITUPDT\PES\PESTEMP1
DELE ALL
PACK
SELE 4
USE \PAVEDB\FILES\SKID INDEX \PAVEDB\INDEXES\SKID
SELE 5
USE \PAVEDB\FILES\PESGEN INDEX \PAVEDB\INDEXES\PESGEN
sele 6
USE \PAVEDB\FILES\LAYER INDEX \PAVEDB\INDEXES\LAYNDX

MSID_ADDED  = 0

SELE 1
DO WHILE .NOT. EOF()
   * check if record is active
   IF .NOT. ACTVFLAG
      SKIP
      LOOP
   ENDIF
   * assign to memory variables
```

```
   MSID      = SID_NO
   MHWYDIST  = STR(HWYDIST,2)
   MCNTYNUM  = STR(CNTYNUM,3)
   MHWYPREFX = HWYPREFX
   MHWYNUM   = STR(HWYNUM,4)
   MHWYSUFFX = HWYSUFFX
   MBEGMPST  = STR(BEGMPST,3)
   MBDISSIGN = BDISSIGN
   MBMPSTDIS = STR(BMPSTDIS,2)
   MLANEID   = LANEID
   MENDMPST  = ENDMPST
   MEDISSIGN = EDISSIGN
   MEMPSTDIS = EMPSTDIS
   MFOUND1   = .F.
   MFOUND2   = .F.
   MERRORS   = 0
   MLANESET  = 'R-L'

   * find the record in PES SKID file
   SELECT 5
   seek MHWYDIST+MCNTYNUM+MHWYPREFX+MHWYNUM+MHWYSUFFX+MBEGMPST+MBDISSIGN-
+MBMPSTDIS+MLANESET
   IF FOUND()
     MFOUND1 = .T.
   ELSE
     IF MLANEID = 'R'
        MLANESET   = 'R-V'
     ELSE
        MLANESET   = 'L-P'
     ENDIF
     seek MHWYDIST+MCNTYNUM+MHWYPREFX+MHWYNUM+MHWYSUFFX+MBEGMPST+MBDISSIGN-
+MBMPSTDIS+MLANESET
     IF FOUND()
       MFOUND1 = .T.
     ENDIF
   ENDIF
   SELE 2
   seek MHWYDIST+MCNTYNUM+MHWYPREFX+MHWYNUM+MHWYSUFFX+MBEGMPST+MBDISSIGN-
+MBMPSTDIS+MLANESET
   IF FOUND()
     MFOUND2 = .T.
   ENDIF
   IF MFOUND1 .AND. MFOUND2
     SELE 5
     IF ENDMPST <> MENDMPST .OR. EDISSIGN <> MEDISSIGN .OR. MEMPSTDIS <>
EMPSTDIS
        SELE 3
        IF MERRORS = 0
          APPEND BLANK
        ENDIF
        REPLACE SID_NO WITH MSID
        REPLACE END_MPST WITH 'X'
        MERRORS = MERRORS + 1
        SELE 5
```

```
  ENDIF
  IF A->NUMLANES <> (NOLANES/2)
     SELE 3
     IF MERRORS = 0
       APPEND BLANK
     ENDIF
     REPLACE SID_NO WITH MSID
     REPLACE NO_LANES WITH 'X'
     MERRORS = MERRORS + 1
     SELE 5
  ENDIF
  IF A->FUNCLAS = 0
     REPL A->FUNCLAS WITH FUNCLAS
  ELSE
     IF A->FUNCLAS <> FUNCLAS
       IF MERRORS = 0
          APPEND BLANK
       ENDIF
       REPLACE SID_NO WITH MSID
       REPLACE NO_FUNCL WITH 'X'
       MERRORS = MERRORS + 1
     ENDIF
  ENDIF
  SELE 2
  IF LASTUPDT = 0
       * Adds to inconsistent report because key is 0
       SELE 3
       IF MERRORS = 0
          APPEND BLANK
       ENDIF
       REPLACE SID_NO WITH MSID
       REPLACE NO_PES WITH 'X'
       REPLACE NO_FUNCL WITH ' '
       REPLACE NO_LANES WITH ' '
       REPLACE END_MPST WITH ' '
       MERRORS = MERRORS + 1
  ENDIF
ELSE
    SELE 3
    APPEND BLANK
    REPLACE SID_NO WITH MSID
    REPLACE NO_PES WITH 'X'
ENDIF
IF MERRORS = 0 .AND. MFOUND1 .AND. MFOUND2
   * get the structure number and the layer number for the record
   * Assign to memory variables

   SELE 2
   MDATE = STR(LASTUPDT,6)
   MYEAR = VAL(LEFT(MDATE,2))
   MMONTH = VAL(SUBSTR(MDATE,3,2))

   SELECT 6
   * find the sid # in layer file
```

```
seek str(MSID,4)
IF FOUND()
   DO WHILE MSID = SID_NO
       SKIP
   ENDDO
   SKIP -1
   * find the layer # which corresponds to the skid year
   do while jobcmpyr > myear
     skip -1
   enddo
   * if the layer year and skid year are the same, need to
   * check the month to get the right layer number
   DO WHILE JOBCMPYR = MYEAR .AND. JOBCMPMO > MMONTH
         SKIP -1
         IF SID_NO <> MSID
            SKIP
            MERRORS = MERRORS + 1
            EXIT
         ENDIF
   ENDDO

   * check to avoid if layer is a base, subbase or subgrade
   * if not, then replace the structure and layer # from layer file
   IF LAYDESC=5 .OR. LAYDESC=6 .OR. LAYDESC=7 .OR. LAYDESC=11
      * Print out on report because layer is a base
      SELE 3
      APPEND BLANK
      REPLACE SID_NO WITH MSID
      REPLACE NO_SURF WITH 'X'
      MERRORS = MERRORS + 1
   ELSE
      MSTRUCNUM = F->STRUCNUM
      MLAYNUM   = F->LAYNUM
   ENDIF
ELSE
   SELE 3
   APPEND BLANK
   REPLACE SID_NO WITH MSID
   REPLACE NO_SURF WITH 'X'
   MERRORS = MERRORS + 1
ENDIF
IF MERRORS = 0
   SELE 4
   * check to see if data is already present
   SEEK
STR(MSID,4)+STR(MSTRUCNUM,2)+STR(MLAYNUM,2)+STR(MYEAR,2)+STR(MMONTH,2)
   IF FOUND()
       CLEAR
       ? "This program has already been run previously. "
       ? "Terminating process and Returning to main menu"
       ? " "
       ? " "
       ? " "
       WAIT
```

```
            CLOSE ALL
            RETURN TO MASTER
         ENDIF
         * add the data to the skid file
         MSID_ADDED = MSID_ADDED + 1
         APPEND BLANK
         REPLACE SID_NO WITH MSID
         REPLACE STRUCNUM WITH MSTRUCNUM
         REPLACE LAYNUM WITH MLAYNUM
         REPLACE YEAR WITH MYEAR
         REPLACE MONTH WITH MMONTH
         REPLACE SKIDNUMM WITH B->SKIDNUMN
         REPLACE SKIDNUMH WITH B->SKIDNUMH
         REPLACE SKIDNUML WITH B->SKIDNUML
      ENDIF
   ENDIF
   SELE 1
   SKIP
   STORE 0 TO MSTRUCNUM, MLAYNUM, MSID, MYEAR, MMONTH
ENDDO

close database
RETURN
```

```
*
* SUBSYSTEM:      EDIT & UPDATE VISUAL FILE
* PROGRAM NAME:   PES_VISL.PRG            05/16/88
* PROJECT 2456 - TEXAS FLEXIBLE PAVEMENT DATABASE CONVERSION
* TAMU/TTI
* AUTHOR:         TREVOR X. PEREIRA
* PURPOSE:        ADD THE PES DATA TO THE VISUAL FILE
*
* THE FOLLOWING FILES ARE USED BY THIS PROGRAM
*    LOCATION.DBF
*    LOCSID  .NDX
*    PESTEMP3.DBF (A temporary PES file to print out the Missing )
*                  (and Inconsistent Data Report for VISUAL file)
*    VISUAL  .DBF
*    VISUAL  .NDX
*    LAYER   .DBF
*    LAYNDX  .NDX
*    PESVISL .DBF
*    PESVISL .NDX
*    PESSCR  .DBF
*    PESSCR  .NDX
*
SELECT 1
USE \PAVEDB\FILES\LOCATION
SELE 2
USE \PAVEDB\FILES\PESVISL INDEX \PAVEDB\INDEXES\PESVISL
SELE 3
USE \PAVEDB\EDITUPDT\PES\PESTEMP3
DELE ALL
PACK
SELE 5
USE \PAVEDB\FILES\PESGEN INDE \PAVEDB\INDEXES\PESGEN
sele 6
USE \PAVEDB\FILES\LAYER INDEX \PAVEDB\INDEXES\LAYNDX
SELE 7
USE \PAVEDB\FILES\PESSCR INDEX \PAVEDB\INDEXES\PESSCR

MSID_ADDED  = 0
SELE 1
DO WHILE .NOT. EOF()
   * check if record is active
   IF .NOT. ACTVFLAG
       SKIP
       LOOP
   ENDIF
   * assign to memory variables
   MSID      = SID_NO
   MHWYDIST  = STR(HWYDIST,2)
   MCNTYNUM  = STR(CNTYNUM,3)
```

```
   MHWYPREFX = HWYPREFX
   MHWYNUM   = STR(HWYNUM,4)
   MHWYSUFFX = HWYSUFFX
   MBEGMPST  = STR(BEGMPST,3)
   MBDISSIGN = BDISSIGN
   MBMPSTDIS = STR(BMPSTDIS,2)
   MLANEID   = LANEID
   MENDMPST  = ENDMPST
   MEDISSIGN = EDISSIGN
   MEMPSTDIS = EMPSTDIS
   MFOUND1   = .F.
   MFOUND2   = .F.
   MFOUND3   = .F.
   MERRORS   = 0
   MLANESET  = 'R-L'

   * find the record in PES SI file
   SELE 2
   seek MHWYDIST+MCNTYNUM+MHWYPREFX+MHWYNUM+MHWYSUFFX+MBEGMPST+MBDISSIGN-
+MBMPSTDIS+MLANESET
   IF FOUND()
     MFOUND1 = .T.
   ELSE
     IF MLANEID = 'R'
        MLANESET   = 'R-V'
     ELSE
        MLANESET   = 'L-P'
     ENDIF
     seek MHWYDIST+MCNTYNUM+MHWYPREFX+MHWYNUM+MHWYSUFFX+MBEGMPST+MBDISSIGN-
+MBMPSTDIS+MLANESET
     IF FOUND()
       MFOUND1 = .T.
     ENDIF
   ENDIF
   SELE 7
   seek MHWYDIST+MCNTYNUM+MHWYPREFX+MHWYNUM+MHWYSUFFX+MBEGMPST+MBDISSIGN-
+MBMPSTDIS+MLANESET
   IF FOUND()
     MFOUND2 = .T.
   ENDIF
   SELE 5
   seek MHWYDIST+MCNTYNUM+MHWYPREFX+MHWYNUM+MHWYSUFFX+MBEGMPST+MBDISSIGN-
+MBMPSTDIS+MLANESET
   IF FOUND()
     MFOUND3 = .T.
   ENDIF

   IF MFOUND1 .AND. MFOUND2 .AND. MFOUND3
     SELE 5
     IF ENDMPST <> MENDMPST .OR. EDISSIGN <> MEDISSIGN .OR. MEMPSTDIS <>
EMPSTDIS
        SELE 3
        IF MERRORS = 0
          APPEND BLANK
```

```
      ENDIF
      REPLACE SID_NO WITH MSID
      REPLACE END_MPST WITH 'X'
      MERRORS = MERRORS + 1
      SELE 5
   ENDIF
   IF A->NUMLANES <> (NOLANES/2)
      SELE 3
      IF MERRORS = 0
        APPEND BLANK
      ENDIF
      REPLACE SID_NO WITH MSID
      REPLACE NO_LANES WITH 'X'
      MERRORS = MERRORS + 1
      SELE 5
   ENDIF
   IF A->FUNCLAS = 0
      REPL A->FUNCLAS WITH FUNCLAS
   ELSE
      IF A->FUNCLAS <> FUNCLAS
        IF MERRORS = 0
           APPEND BLANK
        ENDIF
        REPLACE SID_NO WITH MSID
        REPLACE NO_FUNCL WITH 'X'
        MERRORS = MERRORS + 1
      ENDIF
   ENDIF
   SELE 2
   IF DATE = 0
        * Adds to inconsistent report because key is 0
        SELE 3
        IF MERRORS = 0
           APPEND BLANK
        ENDIF
        REPLACE SID_NO WITH MSID
        REPLACE NO_PES WITH 'X'
        REPLACE NO_FUNCL WITH ' '
        REPLACE NO_LANES WITH ' '
        REPLACE END_MPST WITH ' '
        MERRORS = MERRORS + 1
   ENDIF
ELSE
     SELE 3
     APPEND BLANK
     REPLACE SID_NO WITH MSID
     REPLACE NO_PES WITH 'X'
ENDIF
IF MFOUND1 .AND. MFOUND2 .AND. MFOUND3 .AND. MERRORS = 0
    * get the structure number and the layer number for the record
    * Assign to memory variables
    SELE 2
    MDATE = STR(DATE,6)
    MYEAR = VAL(LEFT(MDATE,2))
```

```
MMONTH = VAL(SUBSTR(MDATE,3,2))
MDAY   = VAL(SUBSTR(MDATE,5,2))

SELECT 6
* find the sid # in layer file
seek str(MSID,4)
IF FOUND()
   DO WHILE MSID = SID_NO
      SKIP
   ENDDO
   SKIP -1
   * find the layer # which corresponds to the skid year
   do while jobcmpyr > myear
     skip -1
   enddo
   * if the layer year and skid year are the same, need to
   * check the month to get the right layer number
   DO WHILE JOBCMPYR = MYEAR .AND. JOBCMPMO > MMONTH
         SKIP -1
         IF SID_NO <> MSID
            SKIP
            MERRORS = MERRORS + 1
            EXIT
         ENDIF
   ENDDO
   * check to avoid if layer is a base, subbase or subgrade
   * if not, then replace the structure and layer # from layer file
   IF LAYDESC=5 .OR. LAYDESC=6 .OR. LAYDESC=7 .OR. LAYDESC=11
      SELE 3
      APPEND BLANK
      REPLACE SID_NO WITH MSID
      REPLACE NO_SURF WITH 'X'
      MERRORS = MERRORS + 1
   ELSE
      MSTRUCNUM = F->STRUCNUM
      MLAYNUM   = F->LAYNUM
   ENDIF
ELSE
   SELE 3
   APPEND BLANK
   REPLACE SID_NO WITH MSID
   REPLACE NO_SURF WITH 'X'
   MERRORS = MERRORS + 1
ENDIF
IF MERRORS = 0
   * add the data to the skid file
   SELE 5
   USE
   SELE 6
   USE
   SELE 4
   USE \PAVEDB\FILES\VISUAL INDEX \PAVEDB\INDEXES\VISUAL
   APPEND BLANK
   REPLACE SID_NO WITH MSID
```

```
REPLACE STRUCNUM WITH MSTRUCNUM
REPLACE LAYNUM WITH MLAYNUM
REPLACE YEAR WITH MYEAR
REPLACE ACTYEAR WITH MYEAR
REPLACE ACTMONTH WITH MMONTH
DO CASE
 CASE B->RUTT = 000
      REPL RUTTSL WITH 0
      REPL RUTTMD WITH 0
      REPL RUTTSV WITH 0
 CASE B->RUTT = 100
      REPL RUTTSL WITH 0
      REPL RUTTMD WITH 1
      REPL RUTTSV WITH 0
 CASE B->RUTT = 010
      REPL RUTTSL WITH 0
      REPL RUTTMD WITH 2
      REPL RUTTSV WITH 0
 CASE B->RUTT = 001
      REPL RUTTSL WITH 0
      REPL RUTTMD WITH 3
      REPL RUTTSV WITH 0
 CASE B->RUTT = 200
      REPL RUTTSL WITH 0
      REPL RUTTMD WITH 0
      REPL RUTTSV WITH 1
 CASE B->RUTT = 020
      REPL RUTTSL WITH 0
      REPL RUTTMD WITH 0
      REPL RUTTSV WITH 2
 CASE B->RUTT = 002
      REPL RUTTSL WITH 0
      REPL RUTTMD WITH 0
      REPL RUTTSV WITH 3
ENDCASE
DO CASE
  CASE B->BLKCR = 000
       REPL BLKCRKSL WITH 0
       REPL BLKCRKMD WITH 0
       REPL BLKCRKSV WITH 0
  CASE B->BLKCR = 100
       REPL BLKCRKSL WITH 0
       REPL BLKCRKMD WITH 1
       REPL BLKCRKSV WITH 0
  CASE B->BLKCR = 010
       REPL BLKCRKSL WITH 0
       REPL BLKCRKMD WITH 2
       REPL BLKCRKSV WITH 0
  CASE B->BLKCR = 001
       REPL BLKCRKSL WITH 0
       REPL BLKCRKMD WITH 3
       REPL BLKCRKSV WITH 0
ENDCASE
DO CASE
```

```
  CASE B->ALLGCR = 000
       REPL ALLGCRSL WITH 0
       REPL ALLGCRMD WITH 0
       REPL ALLGCRSV WITH 0
  CASE B->ALLGCR = 100
       REPL ALLGCRSL WITH 0
       REPL ALLGCRMD WITH 1
       REPL ALLGCRSV WITH 0
  CASE B->ALLGCR = 010
       REPL ALLGCRSL WITH 0
       REPL ALLGCRMD WITH 2
       REPL ALLGCRSV WITH 0
  CASE B->ALLGCR = 001
       REPL ALLGCRSL WITH 0
       REPL ALLGCRMD WITH 3
       REPL ALLGCRSV WITH 0
ENDCASE
DO CASE
  CASE B->LONGCR = 000
       REPL LONGCRSL WITH 0
       REPL LONGCRMD WITH 0
       REPL LONGCRSV WITH 0
  CASE B->LONGCR = 100
       REPL LONGCRSL WITH 0
       REPL LONGCRMD WITH 1
       REPL LONGCRSV WITH 0
  CASE B->LONGCR = 010
       REPL LONGCRSL WITH 0
       REPL LONGCRMD WITH 2
       REPL LONGCRSV WITH 0
  CASE B->LONGCR = 001
       REPL LONGCRSL WITH 0
       REPL LONGCRMD WITH 3
       REPL LONGCRSV WITH 0
ENDCASE
DO CASE
  CASE B->TRANCR = 000
       REPL TRANCRSL WITH 0
       REPL TRANCRMD WITH 0
       REPL TRANCRSV WITH 0
  CASE B->TRANCR = 100
       REPL TRANCRSL WITH 0
       REPL TRANCRMD WITH 1
       REPL TRANCRSV WITH 0
  CASE B->TRANCR = 010
       REPL TRANCRSL WITH 0
       REPL TRANCRMD WITH 2
       REPL TRANCRSV WITH 0
  CASE B->TRANCR = 001
       REPL TRANCRSL WITH 0
       REPL TRANCRMD WITH 3
       REPL TRANCRSV WITH 0
ENDCASE
DO CASE
```

```
            CASE B->PATCH  = 000
                 REPL PATCHGD WITH 0
                 REPL PATCHFR WITH 0
                 REPL PATCHPR WITH 0
            CASE B->PATCH  = 100
                 REPL PATCHGD WITH 0
                 REPL PATCHFR WITH 1
                 REPL PATCHPR WITH 0
            CASE B->PATCH  = 010
                 REPL PATCHGD WITH 0
                 REPL PATCHFR WITH 2
                 REPL PATCHPR WITH 0
            CASE B->PATCH  = 001
                 REPL PATCHGD WITH 0
                 REPL PATCHFR WITH 3
                 REPL PATCHPR WITH 0
          ENDCASE
          DO CASE
            CASE B->FAILMILE = 000
                 REPL FAILMILE WITH 0
            CASE B->FAILMILE = 100
                 REPL FAILMILE WITH 1
            CASE B->FAILMILE = 010
                 REPL FAILMILE WITH 2
            CASE B->FAILMILE = 001
                 REPL FAILMILE WITH 3
          ENDCASE
          REPL PESPVTRS WITH G->PVMTSCR
          REPL UVURS WITH G->UNWGHTVL
          sele 4
          USE
          SELE 5
          USE \PAVEDB\FILES\PESGEN INDE \PAVEDB\INDEXES\PESGEN
          sele 6
          USE \PAVEDB\FILES\LAYER INDEX \PAVEDB\INDEXES\LAYNDX
        ENDIF
      ELSE
        SELE 3
        APPEND BLANK
        REPL SID_NO WITH MSID
        REPL NO_PES WITH 'X'
      ENDIF
      SELE 1
      SKIP
    ENDDO
    CLOSE DATABASES
    RETURN
```

```
*
* SUBSYSTEM:      EDIT & UPDATE FALLING WEIGHT SSI FILE
* PROGRAM NAME:   PES_SSI.PRG            05/16/88
* PROJECT 2456 - TEXAS FLEXIBLE PAVEMENT DATABASE CONVERSION
* TAMU/TTI
* AUTHOR:         TREVOR X. PEREIRA
* PURPOSE:        ADD THE PES DATA TO THE SSI FILE
*
* THE FOLLOWING FILES ARE USED BY THIS PROGRAM
*    LOCATION.DBF
*    LOCSID  .NDX
*    PESTEMP4.DBF (A temporary PES file to print out the Missing )
*                  (and Inconsistent Data Report for SI file)
*    FALLWGHT.DBF
*    FALLWGHT.NDX
*    LAYER   .DBF
*    LAYNDX  .NDX
*    PESSSI  .DBF
*    PESSSI  .NDX
*

SELECT 1
USE \PAVEDB\FILES\LOCATION
SELE 2
USE \PAVEDB\FILES\PESSSI INDEX \PAVEDB\INDEXES\PESSSI
SELE 3
USE \PAVEDB\EDITUPDT\PES\PESTEMP4
DELE ALL
PACK
SELE 4
USE \PAVEDB\FILES\FALLWGHT INDEX \PAVEDB\INDEXES\FALLWGHT
SELE 5
USE \PAVEDB\FILES\PESGEN INDE \PAVEDB\INDEXES\PESGEN
sele 6
USE \PAVEDB\FILES\LAYER INDEX \PAVEDB\INDEXES\LAYNDX

MSID_ADDED  = 0

SELE 1
DO WHILE .NOT. EOF()
   * check if record is active
   IF .NOT. ACTVFLAG
       SKIP
       LOOP
   ENDIF
   * assign to memory variables
   MSID      = SID_NO
   MHWYDIST  = STR(HWYDIST,2)
   MCNTYNUM  = STR(CNTYNUM,3)
```

```
   MHWYPREFX = HWYPREFX
   MHWYNUM   = STR(HWYNUM,4)
   MHWYSUFFX = HWYSUFFX
   MBEGMPST  = STR(BEGMPST,3)
   MBDISSIGN = BDISSIGN
   MBMPSTDIS = STR(BMPSTDIS,2)
   MLANEID   = LANEID
   MENDMPST  = ENDMPST
   MEDISSIGN = EDISSIGN
   MEMPSTDIS = EMPSTDIS
   MFOUND1   = .F.
   MFOUND2   = .F.
   MERRORS   = 0
   MLANESET  = 'R-L'

   * find the record in PES SSI file
   SELECT 5
   seek MHWYDIST+MCNTYNUM+MHWYPREFX+MHWYNUM+MHWYSUFFX+MBEGMPST+MBDISSIGN-
+MBMPSTDIS+MLANESET
   IF FOUND()
     MFOUND1 = .T.
   ELSE
     IF MLANEID = 'R'
        MLANESET   = 'R-V'
     ELSE
        MLANESET   = 'L-P'
     ENDIF
     seek MHWYDIST+MCNTYNUM+MHWYPREFX+MHWYNUM+MHWYSUFFX+MBEGMPST+MBDISSIGN-
+MBMPSTDIS+MLANESET
     IF FOUND()
       MFOUND1 = .T.
     ENDIF
   ENDIF
   SELE 2
   seek MHWYDIST+MCNTYNUM+MHWYPREFX+MHWYNUM+MHWYSUFFX+MBEGMPST+MBDISSIGN-
+MBMPSTDIS+MLANESET
   IF FOUND()
     MFOUND2 = .T.
   ENDIF
   IF MFOUND1 .AND. MFOUND2
     SELE 5
     IF ENDMPST <> MENDMPST .OR. EDISSIGN <> MEDISSIGN .OR. MEMPSTDIS <>
EMPSTDIS
        SELE 3
        IF MERRORS = 0
          APPEND BLANK
        ENDIF
        REPLACE SID_NO WITH MSID
        REPLACE END_MPST WITH 'X'
        MERRORS = MERRORS + 1
        SELE 5
     ENDIF
     IF A->NUMLANES <> (NOLANES/2)
        SELE 3
```

```
      IF MERRORS = 0
        APPEND BLANK
      ENDIF
      REPLACE SID_NO WITH MSID
      REPLACE NO_LANES WITH 'X'
      MERRORS = MERRORS + 1
      SELE 5
   ENDIF
   IF A->FUNCLAS = 0
      REPL A->FUNCLAS WITH FUNCLAS
   ELSE
      IF A->FUNCLAS <> FUNCLAS
        IF MERRORS = 0
           APPEND BLANK
        ENDIF
        REPLACE SID_NO WITH MSID
        REPLACE NO_FUNCL WITH 'X'
        MERRORS = MERRORS + 1
      ENDIF
   ENDIF
   SELE 2
   IF SSIDATE = 0
        * Adds to inconsistent report because key is 0
        SELE 3
        IF MERRORS = 0
           APPEND BLANK
        ENDIF
        REPLACE SID_NO WITH MSID
        REPLACE NO_PES WITH 'X'
        REPLACE NO_FUNCL WITH ' '
        REPLACE NO_LANES WITH ' '
        REPLACE END_MPST WITH ' '
        MERRORS = MERRORS + 1
   ENDIF
ELSE
     SELE 3
     APPEND BLANK
     REPLACE SID_NO WITH MSID
     REPLACE NO_PES WITH 'X'
ENDIF

IF MFOUND1 .AND. MFOUND2 .AND. MERRORS = 0
   * get the structure number and the layer number for the record
   * Assign to memory variables
   SELE 2
   MDATE = STR(SSIDATE,6)
   MYEAR = VAL(LEFT(MDATE,2))
   MMONTH = VAL(SUBSTR(MDATE,3,2))
   MDAY   = VAL(SUBSTR(MDATE,5,2))

   SELECT 6
   * find the sid # in layer file
   seek str(MSID,4)
   IF FOUND()
```

```
   DO WHILE MSID = SID_NO
       SKIP
   ENDDO
   SKIP -1
   * find the layer # which corresponds to the skid year
   do while jobcmpyr > myear
     skip -1
   enddo
   * if the layer year and skid year are the same, need to
   * check the month to get the right layer number
   DO WHILE JOBCMPYR = MYEAR .AND. JOBCMPMO > MMONTH
         SKIP -1
         IF SID_NO <> MSID
            SKIP
            MERRORS = MERRORS + 1
            EXIT
         ENDIF
   ENDDO

   * check to avoid if layer is a base, subbase or subgrade
   * if not, then replace the structure and layer # from layer file
   IF LAYDESC=5 .OR. LAYDESC=6 .OR. LAYDESC=7 .OR. LAYDESC=11
      SELE 3
      APPEND BLANK
      REPLACE SID_NO WITH MSID
      REPLACE NO_SURF WITH 'X'
      MERRORS = MERRORS + 1
   ELSE
      MSTRUCNUM = F->STRUCNUM
      MLAYNUM   = F->LAYNUM
   ENDIF
ELSE
   SELE 3
   APPEND BLANK
   REPLACE SID_NO WITH MSID
   REPLACE NO_SURF WITH 'X'
   MERRORS = MERRORS + 1
ENDIF
IF MERRORS = 0
   * add the data to the skid file
   MSID_ADDED = MSID_ADDED + 1
   SELE 4
   APPEND BLANK
   REPLACE SID_NO WITH MSID
   REPLACE STRUCNUM WITH MSTRUCNUM
   REPLACE LAYNUM WITH MLAYNUM
   REPLACE YEAR WITH MYEAR
   REPLACE MONTH WITH MMONTH
   REPLACE DAY WITH MDAY
   REPL RWSSIAVG WITH B->SSIAVG
   REPL SSITEMP WITH B->SSITEMP
   * Lane 1
   REPL SSIGP11 WITH B->GEOPHON11
   REPL SSIGP12 WITH B->GEOPHON12
```

```
            REPL SSIGP13 WITH B->GEOPHON13
            REPL SSIGP14 WITH B->GEOPHON14
            REPL SSIGP15 WITH B->GEOPHON15
            REPL SSIGP16 WITH B->GEOPHON16
            REPL SSIGP17 WITH B->GEOPHON17
            * Lane 2
            REPL SSIGP21 WITH B->GEOPHON21
            REPL SSIGP22 WITH B->GEOPHON22
            REPL SSIGP23 WITH B->GEOPHON23
            REPL SSIGP24 WITH B->GEOPHON24
            REPL SSIGP25 WITH B->GEOPHON25
            REPL SSIGP26 WITH B->GEOPHON26
            REPL SSIGP27 WITH B->GEOPHON27
            * Lane 3
            REPL SSIGP31 WITH B->GEOPHON31
            REPL SSIGP32 WITH B->GEOPHON32
            REPL SSIGP33 WITH B->GEOPHON33
            REPL SSIGP34 WITH B->GEOPHON34
            REPL SSIGP35 WITH B->GEOPHON35
            REPL SSIGP36 WITH B->GEOPHON36
            REPL SSIGP37 WITH B->GEOPHON37
            * Lane 4
            REPL SSIGP41 WITH B->GEOPHON41
            REPL SSIGP42 WITH B->GEOPHON42
            REPL SSIGP43 WITH B->GEOPHON43
            REPL SSIGP44 WITH B->GEOPHON44
            REPL SSIGP45 WITH B->GEOPHON45
            REPL SSIGP46 WITH B->GEOPHON46
            REPL SSIGP47 WITH B->GEOPHON47
            * Lane 5
            REPL SSIGP51 WITH B->GEOPHON51
            REPL SSIGP52 WITH B->GEOPHON52
            REPL SSIGP53 WITH B->GEOPHON53
            REPL SSIGP54 WITH B->GEOPHON54
            REPL SSIGP55 WITH B->GEOPHON55
            REPL SSIGP56 WITH B->GEOPHON56
            REPL SSIGP57 WITH B->GEOPHON57
         ENDIF
      ELSE
        SELE 3
        APPEND BLANK
        REPL SID_NO WITH MSID
        REPL NO_PES WITH 'X'
      ENDIF
      SELE 1
      SKIP
   ENDDO
   CLOSE DATABASE
   RETURN
```

```
*
* SUBSYSTEM:      EDIT & UPDATE SERVICEABILITY INDEX FILE
* PROGRAM NAME:   PES_MRM.PRG            05/13/88
* PROJECT 2456 - TEXAS FLEXIBLE PAVEMENT DATABASE CONVERSION
* TAMU/TTI
* AUTHOR:         TREVOR X. PEREIRA
* PURPOSE:        ADD THE PES DATA TO THE SI FILE
*
* THE FOLLOWING FILES ARE USED BY THIS PROGRAM
*    LOCATION.DBF
*    LOCSID  .NDX
*    PESTEMP2.DBF (A temporary PES file to print out the Missing )
*                  (and Inconsistent Data Report for SI file)
*    SI       .DBF
*    SI       .NDX
*    LAYER    .DBF
*    LAYNDX   .NDX
*    PESMRM   .DBF
*    PESMRM   .NDX
*

SELECT 1
USE \PAVEDB\FILES\LOCATION
SELE 2
USE \PAVEDB\FILES\PESMRM INDEX \PAVEDB\INDEXES\PESMRM
SELE 3
USE \PAVEDB\EDITUPDT\PES\PESTEMP2
DELE ALL
PACK
SELE 4
USE \PAVEDB\FILES\SI INDEX \PAVEDB\INDEXES\SI
SELE 5
USE \PAVEDB\FILES\PESGEN INDE \PAVEDB\INDEXES\PESGEN
sele 6
USE \PAVEDB\FILES\LAYER INDEX \PAVEDB\INDEXES\LAYNDX

MSID_ADDED  = 0

SELE 1
DO WHILE .NOT. EOF()
   * check if record is active
   IF .NOT. ACTVFLAG
       SKIP
       LOOP
   ENDIF
   * assign to memory variables
   MSID      = SID_NO
   MHWYDIST  = STR(HWYDIST,2)
   MCNTYNUM  = STR(CNTYNUM,3)
   MHWYPREFX = HWYPREFX
```

```
   MHWYNUM   = STR(HWYNUM,4)
   MHWYSUFFX = HWYSUFFX
   MBEGMPST  = STR(BEGMPST,3)
   MBDISSIGN = BDISSIGN
   MBMPSTDIS = STR(BMPSTDIS,2)
   MLANEID   = LANEID
   MENDMPST  = ENDMPST
   MEDISSIGN = EDISSIGN
   MEMPSTDIS = EMPSTDIS
   MFOUND1   = .F.
   MFOUND2   = .F.
   MERRORS   = 0
   MLANESET  = 'R-L'

   * find the record in PES SI file
   SELECT 5
   seek MHWYDIST+MCNTYNUM+MHWYPREFX+MHWYNUM+MHWYSUFFX+MBEGMPST+MBDISSIGN-
+MBMPSTDIS+MLANESET
   IF FOUND()
     MFOUND1 = .T.
   ELSE
     IF MLANEID = 'R'
        MLANESET   = 'R-V'
     ELSE
        MLANESET   = 'L-P'
     ENDIF
     seek MHWYDIST+MCNTYNUM+MHWYPREFX+MHWYNUM+MHWYSUFFX+MBEGMPST+MBDISSIGN-
+MBMPSTDIS+MLANESET
     IF FOUND()
       MFOUND1 = .T.
     ENDIF
   ENDIF
   SELE 2
   seek MHWYDIST+MCNTYNUM+MHWYPREFX+MHWYNUM+MHWYSUFFX+MBEGMPST+MBDISSIGN-
+MBMPSTDIS+MLANESET
   IF FOUND()
     MFOUND2 = .T.
   ENDIF
   IF MFOUND1 .AND. MFOUND2
     SELE 5
     IF ENDMPST <> MENDMPST .OR. EDISSIGN <> MEDISSIGN .OR. MEMPSTDIS <>
EMPSTDIS
        SELE 3
        IF MERRORS = 0
          APPEND BLANK
        ENDIF
        REPLACE SID_NO WITH MSID
        REPLACE END_MPST WITH 'X'
        MERRORS = MERRORS + 1
        SELE 5
     ENDIF
     IF A->NUMLANES <> (NOLANES/2)
        SELE 3
        IF MERRORS = 0
```

```
      APPEND BLANK
    ENDIF
    REPLACE SID_NO WITH MSID
    REPLACE NO_LANES WITH 'X'
    MERRORS = MERRORS + 1
    SELE 5
  ENDIF
  IF A->FUNCLAS = 0
    REPL A->FUNCLAS WITH FUNCLAS
  ELSE
    IF A->FUNCLAS <> FUNCLAS
      IF MERRORS = 0
        APPEND BLANK
      ENDIF
      REPLACE SID_NO WITH MSID
      REPLACE NO_FUNCL WITH 'X'
      MERRORS = MERRORS + 1
    ENDIF
  ENDIF
  SELE 2
  IF LASTUPDT = 0
      * Adds to inconsistent report because key is 0
      SELE 3
      IF MERRORS = 0
        APPEND BLANK
      ENDIF
      REPLACE SID_NO WITH MSID
      REPLACE NO_PES WITH 'X'
      REPLACE NO_FUNCL WITH ' '
      REPLACE NO_LANES WITH ' '
      REPLACE END_MPST WITH ' '
      MERRORS = MERRORS + 1
  ENDIF
ELSE
    SELE 3
    APPEND BLANK
    REPLACE SID_NO WITH MSID
    REPLACE NO_PES WITH 'X'
ENDIF

IF MFOUND1 .AND. MFOUND2 .AND. MERRORS = 0
   * get the structure number and the layer number for the record
   * Assign to memory variables
   SELE 2
   MDATE = STR(LASTUPDT,6)
   MYEAR = VAL(LEFT(MDATE,2))
   MMONTH = VAL(SUBSTR(MDATE,3,2))
   MDAY   = VAL(SUBSTR(MDATE,5,2))

   SELECT 6
   * find the sid # in layer file
   seek str(MSID,4)
   IF FOUND()
      DO WHILE MSID = SID_NO
```

```
      SKIP
   ENDDO
   SKIP -1
   * find the layer # which corresponds to the skid year
   do while jobcmpyr > myear
     skip -1
   enddo
   * if the layer year and skid year are the same, need to
   * check the month to get the right layer number
   DO WHILE JOBCMPYR = MYEAR .AND. JOBCMPMO > MMONTH
         SKIP -1
         IF SID_NO <> MSID
            SKIP
            MERRORS = MERRORS + 1
            EXIT
         ENDIF
   ENDDO

   * check to avoid if layer is a base, subbase or subgrade
   * if not, then replace the structure and layer # from layer file
   IF LAYDESC=5 .OR. LAYDESC=6 .OR. LAYDESC=7 .OR. LAYDESC=11
      SELE 3
      APPEND BLANK
      REPLACE SID_NO WITH MSID
      REPLACE NO_SURF WITH 'X'
      MERRORS = MERRORS + 1
   ELSE
      MSTRUCNUM = F->STRUCNUM
      MLAYNUM   = F->LAYNUM
   ENDIF
ELSE
   SELE 3
   APPEND BLANK
   REPLACE SID_NO WITH MSID
   REPLACE NO_SURF WITH 'X'
   MERRORS = MERRORS + 1
ENDIF
IF MERRORS = 0
   * add the data to the skid file
   SELE 4
   APPEND BLANK
   REPLACE SID_NO WITH MSID
   REPLACE STRUCNUM WITH MSTRUCNUM
   REPLACE LAYNUM WITH MLAYNUM
   REPLACE YEAR WITH MYEAR
   REPLACE ACTYEAR WITH MYEAR
   REPLACE ACTMONTH WITH MMONTH
   REPLACE ACTDAY WITH MDAY
   REPLACE SICOUNT WITH B->SICOUNT
   REPLACE SIMEAN WITH B->SIMEAN
   REPLACE SISD WITH B->SISTDDEV
   REPLACE SILOWVAL WITH B->SILOWVAL
   REPLACE SIHIVAL WITH B->SIHIVAL
ENDIF
```

```
    ELSE
      SELE 3
      APPEND BLANK
      REPL SID_NO WITH MSID
      REPL NO_PES WITH 'X'
    ENDIF
    SELE 1
    SKIP
ENDDO
CLOSE DATABASES
RETURN
```

Section 3: Inventory Data Update

Narrative on Adding Data Inventory Data

The ENTRLAYR.PRG program displays the ADD Inventory Data Menu. The program uses the procedure ADDLAYR.PRG to add data to the Inventory Files (Location, Layer Identification, Geometric and Shoulder, Surface, Subgrade and Layer Thickness Across The Road). Procedure ADDLAYR.PRG consists of the following five programs: LOCATION to add data to Location File, LAYER for Layer Identification File, GEOSHO for Geometric and Shoulder File, SURFACE for Surface File, SUBGRADE for Subgrade file.

Each of these five programs within ADDLAYR.PRG draws the screen forms to allow the user to enter data for a particular file. This data is stored in temporary dBASE files (LOCN_NEW.DBF, LAYR_NEW.DBF, LAYT_NEW.DBF, GEOS_NEW.DBF, SURF_NEW.DBF and SUBG_NEW.DBF).

After the user has entered all the data, he must backup the temporary dBASE files and run the Edit/Check programs to check the data entered. To accomplish this, ENTRLAYR.PRG calls INV_BKUP.PRG to backup up the newly entered data. After backup of the temporary files, INV_BKUP.PRG calls INV_UPDT.PRG which in turn calls the six Edit/Check procedures (LOCNCHEK.PRG, LAYRCHEK.PRG, LAYTCHEK.PRG, GEOSCHEK.PRG, SURFCHEK.PRG, SUBGCHEK.PRG). If there are no errors in the newly entered data, INV_UPDT.PRG updates the master files. If errors are present, an error listing is printed out.

The programs and the temporary dBASE files for this section (ADD Inventory Data) are stored in the subdirectory \PAVEDB\EDITUPDT.

The inventory add process is illustrated in Figures 13 through 15. Figure 13 depicts the add process on a global level, Figure 14 illustrates the high level program flow, logic, and Figure 15 charts the programs procedures, and input and output files used in the inventory add process.

Inventory Data - Add Process

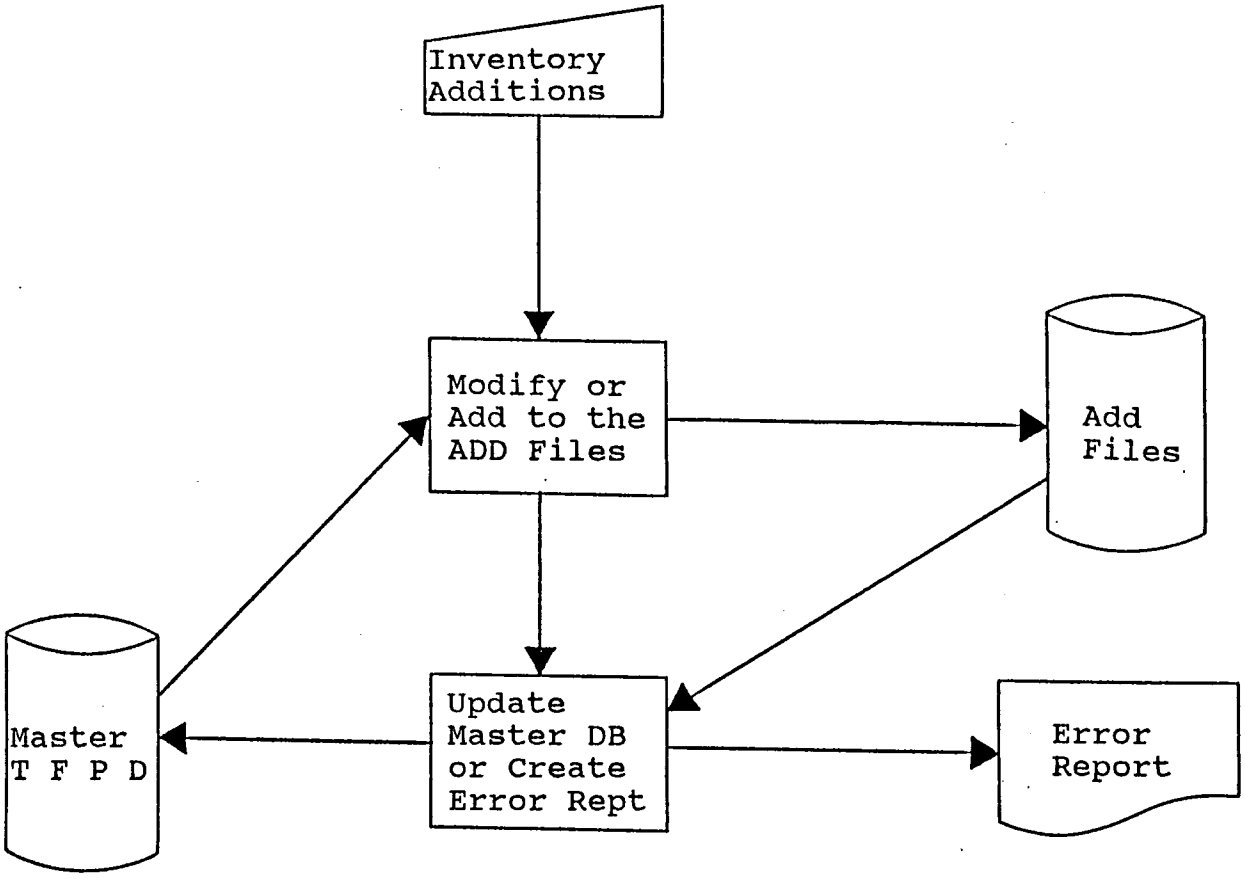


FIGURE 13

Inventory Data - Add Process
Program Flow Chart

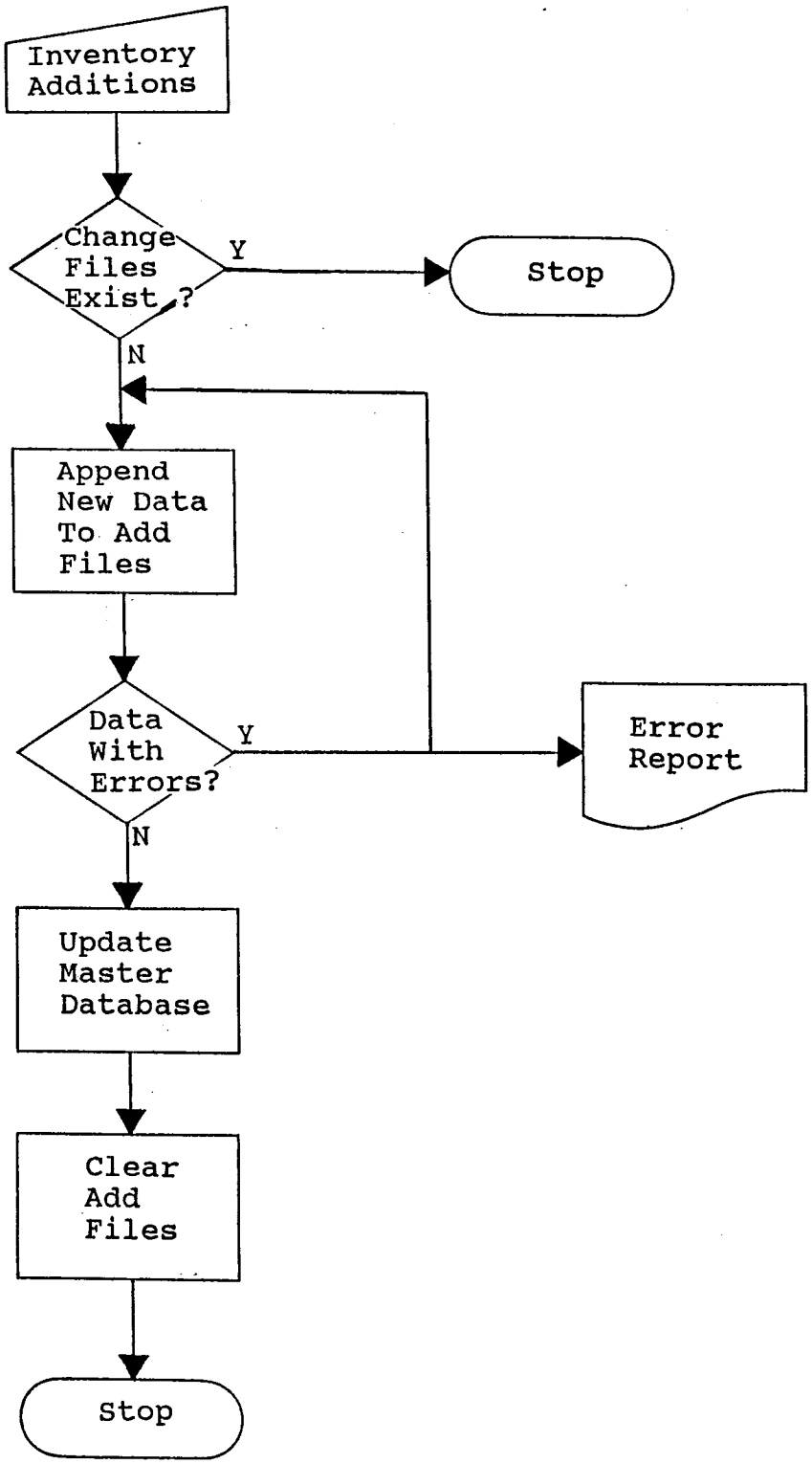


FIGURE 14

INVENTORY DATA - ADD PROCESS
PROGRAM FLOW DIAGRAM

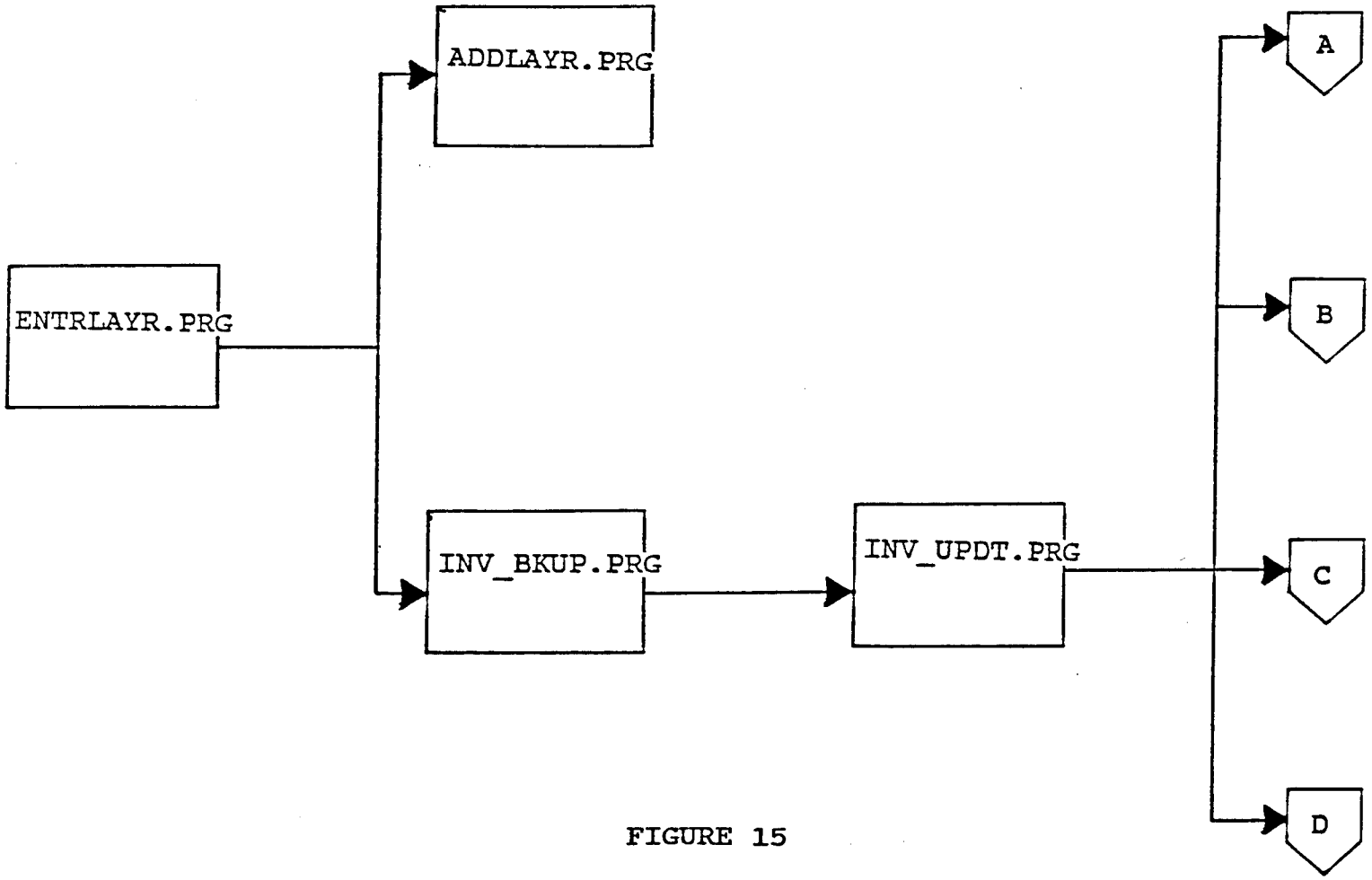


FIGURE 15

INVENTORY DATA - ADD PROCESS
PROGRAM FLOW DIAGRAM (Continued)

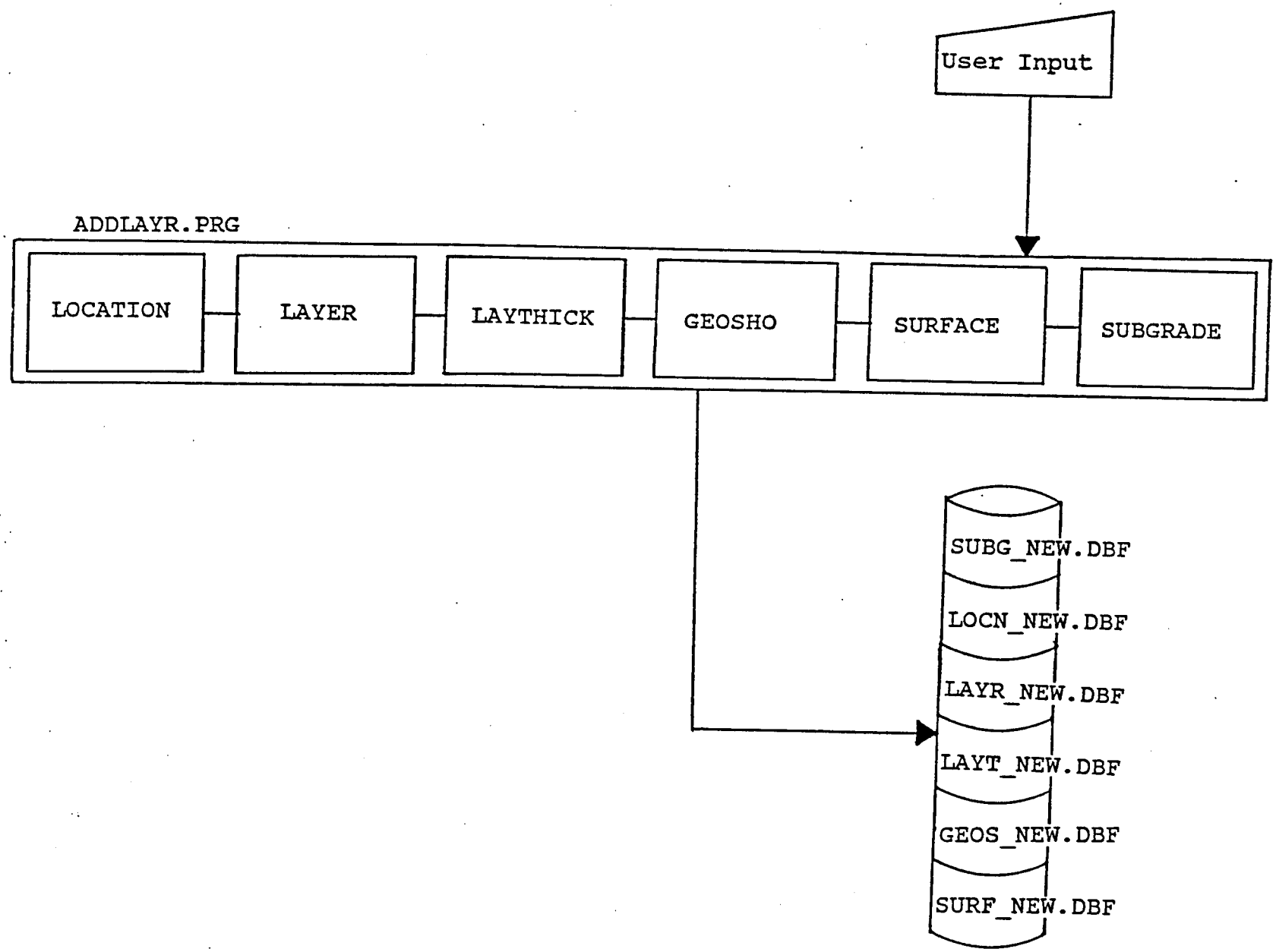


FIGURE 15 (Continued)

INVENTORY DATA - ADD PROCESS
PROGRAM FLOW DIAGRAM (Continued)

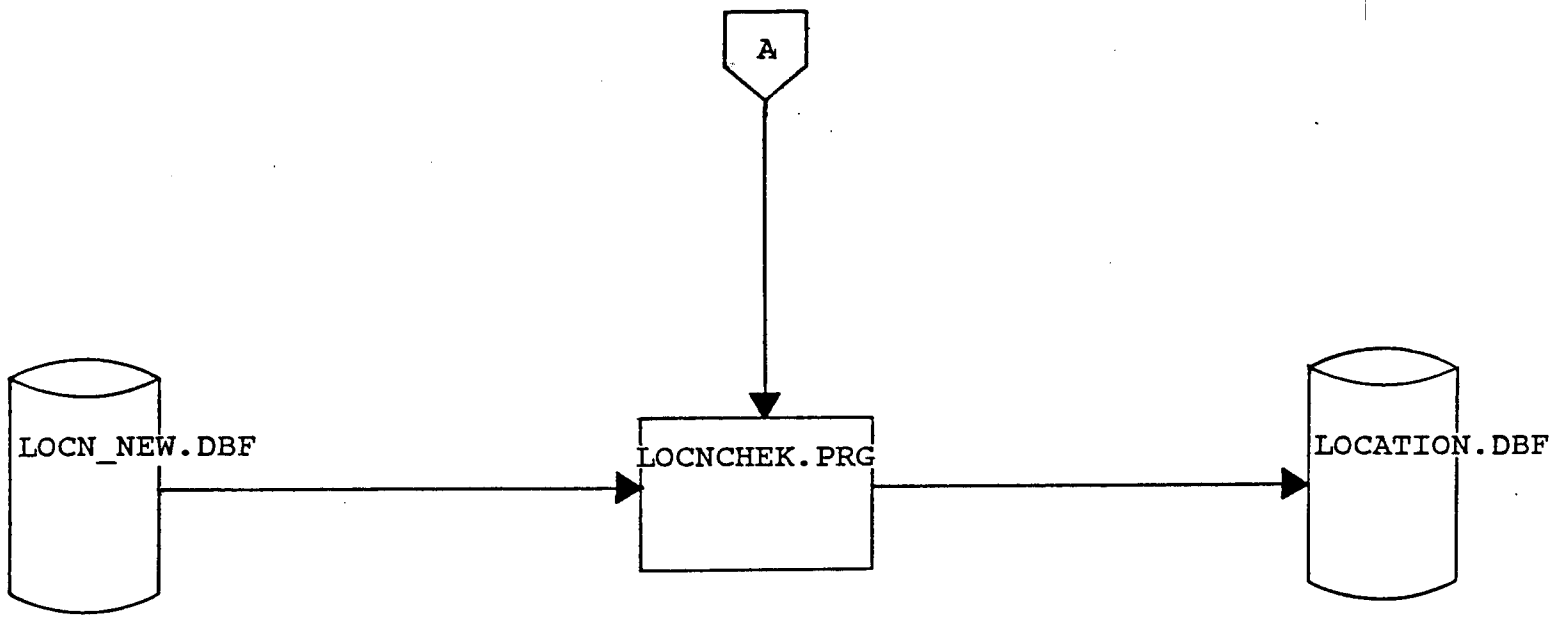


FIGURE 15 (Continued)

INVENTORY DATA - ADD PROCESS
PROGRAM FLOW DIAGRAM (Continued)

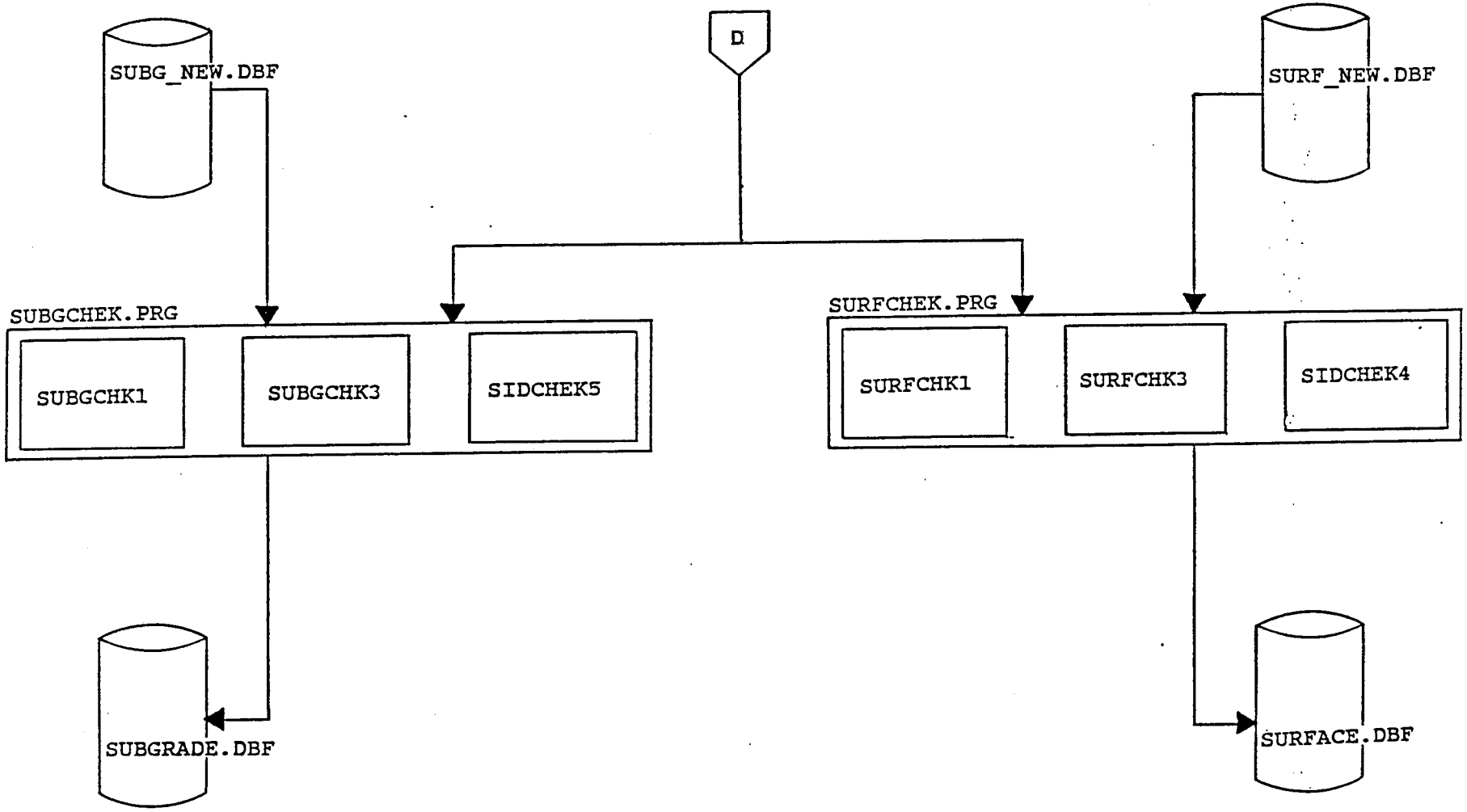


FIGURE 15 (Continued)

INVENTORY DATA - ADD PROCESS
PROGRAM FLOW DIAGRAM (Continued)

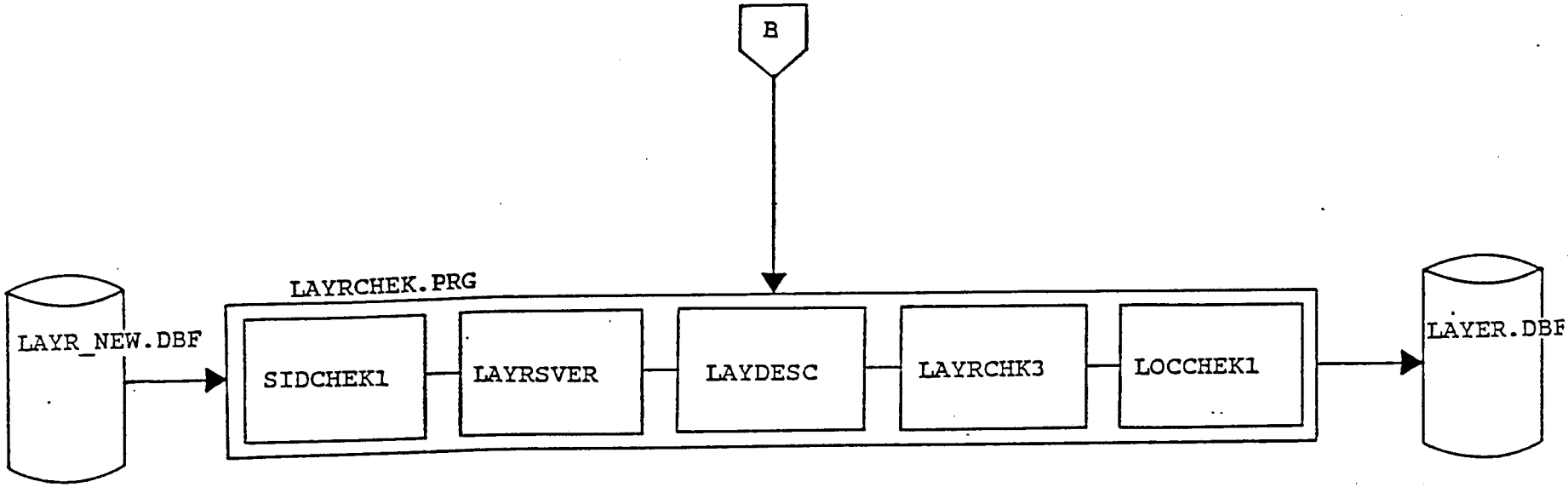


FIGURE 15 (Continued)

INVENTORY DATA - ADD PROCESS
PROGRAM FLOW DIAGRAM (Continued)

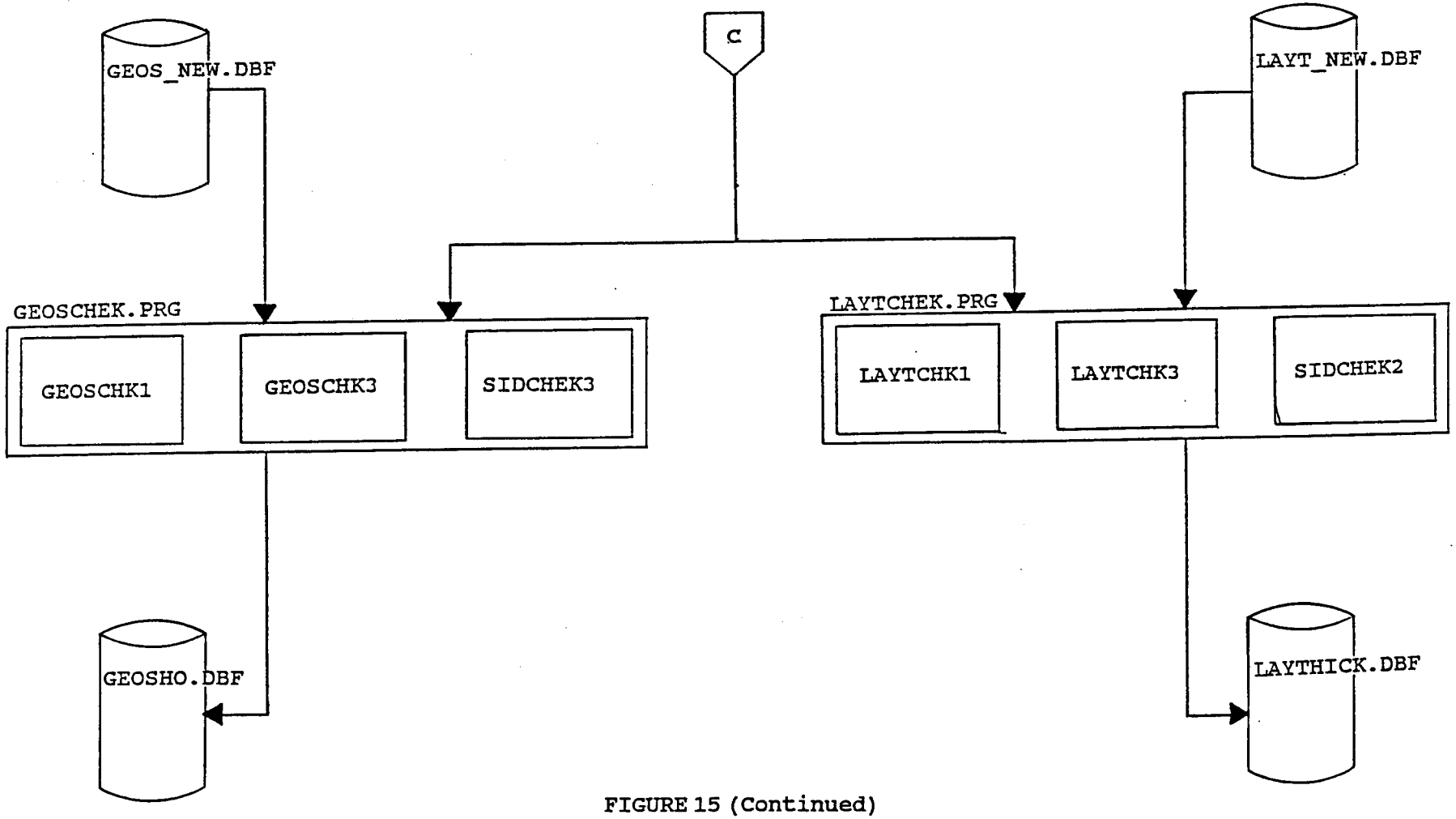


FIGURE 15 (Continued)

PROGRAM SPECIFICATION

Program Name: ENTRLAYR.PRG

Purpose: To display the ADD Inventory data menu and to call the programs that add data to the Inventory master files.

Procedures\Edits:

The following are the procedures:

1) Add data to the files using procedure ADDLAYR.PRG (includes programs LOCATION, LAYER, LAYTHICK, GEOSHO, SURFACE and SUBGRADE).
2) Run the Edit/Check procedures to flag errors in the newly entered data (LOCNCHEK.PRG, LAYRCHEK.PRG, LAYTCHEK.PRG, GEOSCHEK.PRG, SURFCHEK.PRG and SUBGCHEK.PRG). Each of these programs have a number of dBASE procedures within them - refer to dBASE Programs Called.

Input\Output Files:

	Files	Indexes
Temporary files -	LOCN_NEW.DBF	LOCN_NEW.NDX
	LAYR_NEW.DBF	LAYR_NEW.NDX
	LAYT_NEW.DBF	LAYT_NEW.NDX
	GEOS_NEW.DBF	GEOS_NEW.NDX
	SURF_NEW.DBF	SURF_NEW.NDX
	SUBG_NEW.DBF	SUBG_NEW.NDX
Master Files -	LOCATION.DBF	LOCSID.NDX
	LAYER.DBF	LAYNDX.NDX
	LAYTHICK.DBF	LAYTNDX.NDX
	GEOSHO.DBF	GEONDX.NDX
	SURFACE.DBF	SURFNDX.NDX
	SUBGRADE.DBF	SUBGNDX.NDX

dBASE Programs Called (See Program Flow Diagram):

- ADDLAYR.PRG
    - LOCATION
    - LAYER
    - LAYTHICK
    - GEOSHO
    - SURFACE
    - SUBGRADE
- INV_BKUP.PRG
- INV_UPDT.PRG
- LOCNCHEK.PRG
- LAYRCHEK.PRG
    - SIDCHEK1
    - LAYRSVER
    - LAYDESC
    - LAYRCHK3
    - LOCCHEK1
- LAYTCHEK.PRG
    - LAYTCHK1
    - LAYTCHK3
    - SIDCHEK2
- GEOSCHEK.PRG
    - GEOSCHK1
    - GEOSCHK3
    - SIDCHEK3
- SURFCHEK.PRG
    - SURFCHK1
    - SURFCHK3
    - SIDCHEK4
- SUBGCHEK.PRG
    - SUBGCHK1
    - SUBGCHK3
    - SIDCHEK5

ADD INVENTORY SCREENS

The following screens are produced by ENTRLAYR.PRG and its subprograms except for the next menu (Edit & Update menu 3.0) which is produced by EDITUPDT.PRG.

```
        TEXAS FLEXIBLE PAVEMENT DATABASE          3.0
                  Edit & Update



        1 - Pavement Condition Data
        2 - Inventory Data
        3 - Traffic Data
        4 - Tables

                                  OPTION ===>
```

Choice 2 Inventory Data asks the user if he wants to ADD or CHANGE data to the inventory files. If he wants to ADD, ENTRLAYR.PRG program is run which produces the next screen (ADD Inventory Data 3.2.A).

```
        TEXAS FLEXIBLE PAVEMENT DATABASE          3.2.A
                  EDIT & UPDATE
                ADD Inventory Data


        1 - Location
        2 - Layer ID
        3 - Geometric & Shoulder
        4 - Surface
        5 - Subgrade
        6 - Layer Thickness Across The Road

        K - Check New Data Entered and ADD to Files
        E - Edit New Data Entered

                                  OPTION ===>
```

Choice 1 Location displays the first screen on this page to let the user add Location data.
Choice 2 Layer ID displays the second screen on this page to let the user add Layer data.

TEXAS FLEXIBLE PAVEMENT DATABASE
EDIT & UPDATE - ADD Inventory
Location File

SID Number 0 District 0 County 0

Highway Ident. 0 Control/Section 0/ 0

Mile Post 0 0 TO 0 0 Lane Identification

Mile Point 0.000 TO 0.000 Mile Point Date 0/ 0

HPMS Sample Number HPMS Section Subdivision 0

Functional Classification 0 Number of Lanes 0

Active ? T Inactive Date 0/ 0 Previous SID 0 Next SID 0

Comment

TEXAS FLEXIBLE PAVEMENT DATABASE
EDIT & UPDATE - ADD Inventory
Layer Identification

Sid Number 0

Structure Number	Layr No.	Layer Desc.	Center Thick	Material Type Class.	Date Job Compltd Mnth	Date Job Compltd Year	Widened Mnth	Widened Year
0	0	0	0.0	0	0	0	0	0
0	0	0	0.00	0	0	0	0	0

Choice 3 Geometric & Shoulder displays the first screen on this page to enter data into.

Choice 4 Surface displays the second screen on this page to enter data into.

TEXAS FLEXIBLE PAVEMENT DATABASE
EDIT & UPDATE - ADD Inventory
Geometric & Shoulder Information

Sid Number 0
Structure Number 0

Type of Pavement (See TTI Codes)	0
Lane Width (Feet)	0
Outside Shoulder Width (Feet)	0
Shoulder Surface Type	0
Shoulder Base Type (See Base Type Code, Table A.6)	0
Shoulder Surface Thickness (Inches)	0.0
Shoulder Base Thickness (Inches)	0.00
Widened Flag (0-2)	0

TEXAS FLEXIBLE PAVEMENT DATABASE
EDIT & UPDATE - ADD Inventory
Surface Layer

SID NUMBER 0

Structure Number	Layer Number	Aggregate Application Rate	Type Admixture	Percent Admixture (Mean Asphalt Content)	Asphalt Application Rate
0	0	0		0.00	0.00

Choice 3 Subgrade displays the first screen on this page to enter data into.

Choice 4 Layer Thickness Across The Road displays the second screen on this page to enter data into.

TEXAS FLEXIBLE PAVEMENT DATABASE
EDIT & UPDATE - ADD Inventory
Subgrade File

SID NUMBER 0

Percent Passing No. 200 Sieve	0.0
Texas Triaxial Class	0.0
Liquid Limit	0.0
Plasticity Index	0.0
Permeability Index	0.00

TEXAS FLEXIBLE PAVEMENT DATABASE
EDIT & UPDATE - ADD Inventory
Layer Thickness Across The Road

SID NUMBER 0

Structure Number	Layer Number	Thickness - From Center				Distance From Center		
		3rd Pos	2nd Pos	1st Pos	Center	3rd Pos	2nd Pos	1st Pos
0	0	0.00	0.00	0.00	0.00	0.0	0.0	0.0
0	0	0.00	0.00	0.00	0.00	0.0	0.0	0.0

```
*
* SUBSYSTEM:      EDIT & UPDATE INVENTORY dBASE III FILES
* PROGRAM NAME:   ENTRLAYR.PRG                06/06/88
* CALLED FROM:    EDITUPDT.PRG
* PROJECT 2456 - TEXAS FLEXIBLE PAVEMENT DATABASE CONVERSION
* TAMU/TTI
* AUTHOR:         TREVOR X. PEREIRA
* PURPOSE:        THIS PROGRAM DISPLAYS THE ADD INVENTORY DATA MENU
*

*
* checks to see if there are any inventory files in the process of
* being changed
*
MCOUNT = 0
IF FILE('\PAVEDB\EDITUPDT\LOCNCHNG.DBF')
     MCOUNT = MCOUNT + 1
ENDIF
IF FILE('\PAVEDB\EDITUPDT\LAYRCHNG.DBF')
     MCOUNT = MCOUNT + 1
ENDIF
IF FILE('\PAVEDB\EDITUPDT\LAYTCHNG.DBF')
     MCOUNT = MCOUNT + 1
ENDIF
IF FILE('\PAVEDB\EDITUPDT\GEOSCHNG.DBF')
     MCOUNT = MCOUNT + 1
ENDIF
IF FILE('\PAVEDB\EDITUPDT\SURFCHNG.DBF')
     MCOUNT = MCOUNT + 1
ENDIF
IF FILE('\PAVEDB\EDITUPDT\SUBGCHNG.DBF')
     MCOUNT = MCOUNT + 1
ENDIF
IF MCOUNT <> 0
   CLEAR
   @ 2, 7 SAY "                    ! ! ! WARNING ! ! !"
   @ 5, 7 SAY "You were in the process of changing data in the inventory "
   @ 6, 7 SAY "   files.   Therefore you cannot add data until you finish."
   @ 7, 7 SAY "   the Change process."
   @ 9, 7 SAY "To complete the change process, "
   @ 10, 7 SAY "   1) Choose choice '3' from the Edit & Update Menu."
   @ 11,7 SAY "   2) Type 'C' to the next question to Change Data"
   @ 12,7 SAY "   3) Now you can either"
   @ 13,7 SAY "          A) Change the data"
   @ 14,7 SAY "               OR"
   @ 15,7 SAY "          B) Correct the Data and run the Edit/Check programs"
   @ 16,7 SAY "               i.e. Choice 'K'."
   @ 19,7 SAY "   Press any key to continue . . ."
   CLEA TYPE
   WAIT " "
   CLEAR
```

```
  DO SURFACE
CASE EDITCOLL = "5"
  @ 19, 23  SAY "Please enter SID Number ======> " GET MSID_NO PICTURE "9999"
  READ
  CLEAR
  DO SUBGRADE
CASE EDITCOLL = "6"
  @ 19, 23  SAY "Please enter SID Number ======> " GET MSID_NO PICTURE "9999"
  READ
  CLEAR
  DO LAYTHICK
CASE EDITCOLL = "K"
  SET PROC TO
  DO \PAVEDB\EDITUPDT\INV_BKUP
  CLEA TYPE
  SET PROC TO \PAVEDB\EDITUPDT\ADDLAYR
CASE EDITCOLL = "E"
  SET PROC TO
  USE \PAVEDB\EDITUPDT\LOCN_NEW
  IF RECCOUNT() <> 0
    SET FORMAT TO \PAVEDB\EDITUPDT\LOCATION.FMT
    CHANGE
    USE \PAVEDB\EDITUPDT\LOCN_NEW INDE \PAVEDB\EDITUPDT\LOCN_NEW
    PACK
  ENDIF
  USE \PAVEDB\EDITUPDT\LAYR_NEW
  IF RECCOUNT() <> 0
    SET FORMAT TO \PAVEDB\EDITUPDT\LAYER.FMT
    CHANGE
    USE \PAVEDB\EDITUPDT\LAYR_NEW INDE \PAVEDB\EDITUPDT\LAYR_NEW
    PACK
  ENDIF
  USE \PAVEDB\EDITUPDT\LAYT_NEW
  IF RECCOUNT() <> 0
    SET FORMAT TO \PAVEDB\EDITUPDT\LAYTHICK.FMT
    CHANGE
    USE \PAVEDB\EDITUPDT\LAYT_NEW INDE \PAVEDB\EDITUPDT\LAYT_NEW
    PACK
  ENDIF
  USE \PAVEDB\EDITUPDT\GEOS_NEW
  IF RECCOUNT() <> 0
    SET FORMAT TO \PAVEDB\EDITUPDT\GEOSHO.FMT
    CHANGE
    USE \PAVEDB\EDITUPDT\GEOS_NEW INDE \PAVEDB\EDITUPDT\GEOS_NEW
    PACK
  ENDIF
  USE \PAVEDB\EDITUPDT\SURF_NEW
  IF RECCOUNT() <> 0
    SET FORMAT TO \PAVEDB\EDITUPDT\SURFACE.FMT
    CHANGE
    USE \PAVEDB\EDITUPDT\SURF_NEW INDE \PAVEDB\EDITUPDT\SURF_NEW
    PACK
  ENDIF
  USE \PAVEDB\EDITUPDT\SUBG_NEW
```

```
   RETURN
ENDIF

*****************
* Inventory Enter data menu
*****************
CLEAR
GETREPLY = " "
* Display the Inventory data menu on the screen
MCONTINUE = .T.
DO WHILE MCONTINUE
 STORE " " TO EDITCOLL
 MSID_NO = 0
 DO WHILE .NOT. (EDITCOLL $ '123456KE')
   @  3, 21  SAY "TEXAS FLEXIBLE PAVEMENT DATABASE      3.3.A"
   @  4, 21  SAY "            EDIT & UPDATE"
   @  5, 21  SAY "         ADD Inventory Data"
   @  8, 15  SAY "1 - Location"
   @  9, 15  SAY "2 - Layer ID"
   @ 10, 15  SAY "3 - Geometric & Shoulder"
   @ 11, 15  SAY "4 - Surface"
   @ 12, 15  SAY "5 - Subgrade"
   @ 13, 15  SAY "6 - Layer Thickness Across The Road"
   @ 15, 15  SAY "K - Check New Data Entered and ADD to Files"
   @ 16, 15  SAY "E - Edit New Data Entered"
   @ 18, 40  SAY "OPTION ======> " GET EDITCOLL
   @  2,  9  TO 20, 65    DOUBLE
   READ
   IF READKEY() = 12
      MCONTINUE = .F.
      SET PROC TO
      EXIT
   ENDIF
 ENDDO
 IF MCONTINUE
   SET PROCEDURE TO \PAVEDB\EDITUPDT\ADDLAYR
   DO CASE
    CASE EDITCOLL = "1"
      CLEAR
      DO LOCATION
    CASE EDITCOLL = "2"
      @ 19, 23  SAY "Please enter SID Number ======> " GET MSID_NO PICTURE "9999"
      READ
      CLEAR
      DO LAYER
    CASE EDITCOLL = "3"
      @ 19, 23  SAY "Please enter SID Number ======> " GET MSID_NO PICTURE "9999"
      READ
      CLEAR
      DO GEOSHO
    CASE EDITCOLL = "4"
      @ 19, 23  SAY "Please enter SID Number ======> " GET MSID_NO PICTURE "9999"
      READ
      CLEAR
```

```
      IF RECCOUNT() <> 0
        SET FORMAT TO \PAVEDB\EDITUPDT\SUBGRADE.FMT
        CHANGE
        USE \PAVEDB\EDITUPDT\SUBG_NEW INDE \PAVEDB\EDITUPDT\SUBG_NEW
        PACK
      ENDIF
      SET FORMAT TO
    ENDCASE
   CLEAR
  ENDIF
  CLEAR
  SET PROCEDURE TO
ENDDO
RETURN
```

```
*
* SUBSYSTEM:      EDIT & UPDATE INVENTORY dBASE III FILES
* PROGRAM NAME:   ADDLAYR.PRG                06/06/88
* CALLED FROM:    ENTRLAYR.PRG
* PROJECT 2456 - TEXAS FLEXIBLE PAVEMENT DATABASE CONVERSION
* TAMU/TTI
* AUTHOR:         TREVOR X. PEREIRA
* PURPOSE:        THIS PROCEDURE CALLS UP THE RESPECTIVE INVENTORY
*                 DATA FILES TO ENTER DATA INTO.  THE FOLLOWING PROCEDURES
*                 CAN BE FOUND HERE:
*                    - LAYER
*                    - LAYTHICK
*                    - GEOSHO
*                    - SURFACE
*                    - SUBGRADE
*                    - LOCATION
*

********************
* PROCEDURE LAYER
********************
PROC LAYER
*
* To enter data into the layer identification file
*
STORE .T. TO MAGAIN
STORE 0 TO MSTRUCNUM1, MLAYNUM1, MLAYDESC1, MLAYMATC1, MJOBCMPM1
STORE 0.0 TO MCENTTHK1
STORE 0 TO MJOBCMPY1, MWIDENIM1, MWIDENLY1

USE \PAVEDB\EDITUPDT\LAYR_NEW INDEX \PAVEDB\EDITUPDT\LAYR_NEW
STORE 0 TO MSTRUCNUM, MLAYNUM, MLAYDESC, MLAYMATC
STORE 0.0 TO MCENTTHK
STORE 0 TO MJOBCMPY, MJOBCMPM, MWIDENLY, MWIDENIM, MWIDEN

DO WHILE MAGAIN
    * initialize all variables
    CLEAR
    MSURE = " "
    DO WHILE .NOT. (MSURE $ 'Yy')
      * set up entry screen and get data
      @  3, 23  SAY "TEXAS FLEXIBLE PAVEMENT DATABASE"
      @  4, 23  SAY "EDIT & UPDATE - ADD Inventory"
      @  5, 23  SAY "   Layer Identification"
      @  7, 55  SAY "Sid Number "
      @  7, 66  SAY MSID_NO PICTURE "9999"
      @ 10, 54  SAY "Date"
      @ 11,  4  SAY "                                    Material  Job Compltd
Widened"
      @ 12,  4  SAY "Structure  Layr   Layer  Center   Type     ----------
```

```
---------"
      @ 13,  4  SAY " Number     No.   Desc.  Thick   Class.    Mnth   Year
Mnth Year"
      @ 14,  4  SAY "--------  -----  -----  ------  --------  ----   ----
---- ----"
      @ 15,  8  SAY  STR(MSTRUCNUM1,1)
      @ 15, 17  SAY  STR(MLAYNUM1,2)
      @ 15, 24  SAY  STR(MLAYDESC1,2)
      @ 15, 30  SAY  STR(MCENTIHK1,4,1)
      @ 15, 40  SAY  STR(MLAYMATC1,2)
      @ 15, 48  SAY  STR(MJOBCMPM1,2)
      @ 15, 55  SAY  STR(MJOBCMPY1,2)
      @ 15, 62  SAY  STR(MWIDENLM1,2)
      @ 15, 67  SAY  STR(MWIDENLY1,2)
      @ 16,  8  GET  MSTRUCNUM PICTURE "9" RANGE 1,9
      @ 16, 17  GET  MLAYNUM PICTURE "99" RANGE 1,13
      @ 16, 24  GET  MLAYDESC PICTURE "99" RANGE 1,14
      @ 16, 30  GET  MCENTIHK PICTURE "99.99" RANGE 0,25.0
      @ 16, 40  GET  MLAYMATC PICTURE "99" RANGE 01,45
      @ 16, 48  GET  MJOBCMPM PICTURE "99" RANGE 1,12
      @ 16, 55  GET  MJOBCMPY PICTURE "99" RANGE 28,87
      @ 16, 62  GET  MWIDENLM PICTURE "99" RANGE 0,12
      @ 16, 67  GET  MWIDENLY PICTURE "99" RANGE 0,87
      @  2,  2  TO 18, 74    DOUBLE
      READ

      * set up the escape key if want to abort
      IF READKEY() = 12
           CLEAR
           USE
           RETURN
      ENDIF
      @ 20, 23 SAY "IS THE DATA CORRECT ? "  GET MSURE PICTURE "!"
      READ

      * if data is said to be not correct, the flags (sure and required) are
      * set to true so that the do loop will be executed again.
      IF MSURE = "Y"
         MSTRUCNUM1 = MSTRUCNUM
         MLAYNUM1  = MLAYNUM
         MLAYDESC1 = MLAYDESC
         MCENTIHK1 = MCENTIHK
         MLAYMATC1 = MLAYMATC
         MJOBCMPM1 = MJOBCMPM
         MJOBCMPY1 = MJOBCMPY
         MWIDENLM1 = MWIDENLM
         MWIDENLY1 = MWIDENLY
         APPEND BLANK
         REPLACE SID_NO WITH MSID_NO
         REPLACE STRUCNUM WITH MSTRUCNUM
         REPLACE LAYNUM WITH MLAYNUM
         REPLACE LAYDESC WITH MLAYDESC
         REPLACE CENTIHK WITH MCENTIHK
         REPLACE LAYMATCL WITH MLAYMATC
```

```
         REPLACE JOBCMPYR WITH MJOBCMPY
         REPLACE JOBCMPMO WITH MJOBCMPM
         REPLACE WIDENLYR WITH MWIDENLY
         REPLACE WIDENLMO WITH MWIDENLM
         REPLACE ERRORCHAR WITH .F.

         STORE 0 TO MSTRUCNUM, MLAYNUM, MLAYDESC, MLAYMATC
         STORE 0.0 TO MCENTTHK
         STORE 0 TO MJOBCMPY, MJOBCMPM, MWIDENLY, MWIDENLM, MWIDEN
      ELSE
         @ 20, 0 CLEAR
      ENDIF
   ENDDO
ENDDO
USE
CLEAR
RETURN



***************
* PROCEDURE LAYTHICK
***************
PROC LAYTHICK

*
* To enter data into the layer thickness across the road file
*
CLEAR
STORE 0 TO MSTRUCNUM, MLAYNUM
STORE 0.0 TO MFC3THK, MFC2THK, MFC1THK
STORE 0.0 TO MCENTTHK, MFC3DIS, MFC2DIS, MFC1DIS
STORE 0 TO MSTRUCNUM2, MLAYNUM2
STORE 0.0 TO MFC3THK2, MFC2THK2, MFC1THK2
STORE 0.0 TO MCENTTHK2, MFC3DIS2, MFC2DIS2, MFC1DIS2
USE \PAVEDB\EDITUPDT\LAYT_NEW INDEX \PAVEDB\EDITUPDT\LAYT_NEW
STORE .T. TO MAGAIN
DO WHILE MAGAIN
   CLEAR
   MSURE = " "
   DO WHILE .NOT. (MSURE $ 'Y')
      * set up entry screen and get data
      @  3, 23  SAY "TEXAS FLEXIBLE PAVEMENT DATABASE"
      @  4, 23  SAY "  EDIT & UPDATE - ADD Inventory"
      @  5, 23  SAY " Layer Thickness Across The Road"
      @  7, 56  SAY "SID NUMBER "
      @  7, 67  SAY MSID_NO PICTURE "9999"
      @ 10, 25  SAY "Thickness - From Center       Distance From Center"
      @ 11,  3  SAY "Structure  Layer   ----------------------------
----------------------"
      @ 12,  4  SAY "Number    Number  3rd Pos 2nd Pos 1st Pos Center  3rd Pos
2nd Pos 1st Pos"
      @ 13,  3  SAY "---------  ------  ------- ------- ------- ------  -------
------- -------"
```

```
@ 14,  6  SAY  str(MSTRUCNUM2,2)
@ 14, 16  SAY  str(MLAYNUM2,2)
@ 14, 24  SAY  str(MFC3THK2,5,2)
@ 14, 32  SAY  str(MFC2THK2,5,2)
@ 14, 40  SAY  str(MFC1THK2,5,2)
@ 14, 47  SAY  str(MCENTTHK2,5,2)
@ 14, 56  SAY  str(MFC3DIS2,4,1)
@ 14, 64  SAY  str(MFC2DIS2,4,1)
@ 14, 72  SAY  str(MFC1DIS2,4,1)
@ 15,  6  GET  MSTRUCNUM  PICTURE "99" RANGE 1,9
@ 15, 16  GET  MLAYNUM    PICTURE "99" RANGE 1,13
@ 15, 24  GET  MFC3THK    PICTURE "99.99" RANGE 0.0,25.0
@ 15, 32  GET  MFC2THK    PICTURE "99.99" RANGE 0.0,25.0
@ 15, 40  GET  MFC1THK    PICTURE "99.99" RANGE 0.1,25.0
@ 15, 47  GET  MCENTTHK   PICTURE "99.99" RANGE 0.1,25.0
@ 15, 56  GET  MFC3DIS    PICTURE "99.9" RANGE 0.0,99.0
@ 15, 64  GET  MFC2DIS    PICTURE "99.9" RANGE 0.0,99.0
@ 15, 72  GET  MFC1DIS    PICTURE "99.9" RANGE 0.0,99.0
@  2,  0  TO 16, 79    DOUBLE
READ

* set up the escape key if want to abort
IF READKEY() = 12
   CLEAR
   USE
   RETURN
ENDIF

@ 20, 23 SAY "IS THE DATA CORRECT ? " GET MSURE PICTURE "!"
READ

* if data is not correct, redraw the entry screen; otherwise add
* the data to the file
IF MSURE = "Y"
   * data is added to the Layer Thickness Across the Road File
   MSTRUCNUM2 = MSTRUCNUM
   MLAYNUM2   = MLAYNUM
   MFC3THK2   = MFC3THK
   MFC2THK2   = MFC2THK
   MFC1THK2   = MFC1THK
   MCENTTHK2  = MCENTTHK
   MFC3DIS2   = MFC3DIS
   MFC2DIS2   = MFC2DIS
   MFC1DIS2   = MFC1DIS

   APPEND BLANK
   REPLACE SID_NO WITH MSID_NO
   REPLACE STRUCNUM WITH MSTRUCNUM
   REPLACE LAYNUM WITH MLAYNUM
   REPLACE FC3THK WITH MFC3THK
   REPLACE FC2THK WITH MFC2THK
   REPLACE FC1THK WITH MFC1THK
   REPLACE CENTTHK WITH MCENTTHK
   REPLACE FC3DIS WITH MFC3DIS
```

```
         REPLACE FC2DIS WITH MFC2DIS
         REPLACE FC1DIS WITH MFC1DIS
         REPLACE ERRORCHAR WITH .F.
         STORE 0 TO MSTRUCNUM, MLAYNUM
         STORE 0.0 TO MFC3THK, MFC2THK, MFC1THK
         STORE 0.0 TO MCENTTHK, MFC3DIS, MFC2DIS, MFC1DIS
      ELSE
         @ 20,  0 CLEAR
      ENDIF
   ENDDO
ENDDO
USE
CLEAR
RETURN


****************
* PROCEDURE GEOMETRIC & SHOULDER
****************
PROC GEOSHO

*
* TO ENTER DATA INTO THE GEOMETRIC AND SHOULDER
* INFORMATION FILE
*

* set parameters and initialize variables
STORE .T. TO MAGAIN
CLEAR
USE \PAVEDB\EDITUPDT\GEOS_NEW INDEX \PAVEDB\EDITUPDT\GEOS_NEW

DO WHILE MAGAIN
   CLEAR
   MSURE = " "
   STORE 0 TO MCONSTNU, MPAVETYP, MLANEWID, MMONLANE, MOUTSHOW
   STORE 0 TO MSHOSFTY, MSHOBSTY, MWIDEN
   STORE 0.0 TO MSHOSFTH, MSHOBSTH
   DO WHILE .NOT. (MSURE $ 'Yy')
      * set up data entry screen
      @  3, 23  SAY "TEXAS FLEXIBLE PAVEMENT DATABASE"
      @  4, 23  SAY "  EDIT & UPDATE - ADD Inventory"
      @  5, 23  SAY "Geometric & Shoulder Information"
      @  7, 56  SAY "Sid Number "
      @  7, 67  SAY MSID_NO PICTURE "9999"
      @  8, 50  SAY "Structure Number"
      @  8, 69  GET  MCONSTNU PICTURE "99" RANGE 1,9
      @ 10,  8  SAY "Type of Pavement (See TTI Codes)"
      @ 10, 61  GET  MPAVETYP PICTURE "99" RANGE 1,27
      @ 11,  8  SAY "Lane Width (Feet)"
      @ 11, 61  GET  MLANEWID PICTURE "99" RANGE 8,15
      @ 12,  8  SAY "Outside Shoulder Width (Feet)"
      @ 12, 61  GET  MOUTSHOW PICTURE "99" RANGE 0,15
      @ 13,  8  SAY "Shoulder Surface Type"
      @ 13, 61  GET  MSHOSFTY PICTURE "99" RANGE 1,6
```

```
      @ 14,  8  SAY "Shoulder Base Type (See Base Type Code, Table A.6)"
      @ 14, 61  GET  MSHOBSTY PICTURE "99" RANGE 0,33
      @ 15,  8  SAY "Shoulder Surface Thickness (Inches)"
      @ 15, 59  GET  MSHOSFTH PICTURE "99.9" RANGE 0.0,10.1
      @ 16,  8  SAY "Shoulder  Base Thickness (Inches)"
      @ 16, 58  GET  MSHOBSTH PICTURE "99.99" RANGE 0.0,21.0
      @ 17,  8  SAY  "Widened Flag (0-2)"
      @ 17, 62  GET  MWIDEN   PICTURE "9" RANGE 0,2
      @  1,  3  TO 19, 74    DOUBLE
      * get data
      READ
      * set up the escape key if want to abort
      IF READKEY() = 12
         CLEAR
         USE
         RETURN
      ENDIF
      @ 20, 23  SAY "Is The Data Entered Correct? " GET MSURE PICTURE "!"
      READ

      * If the data is not correct, the inventory screen is displayed again
      * Otherwise, add data to the Geometric And Shoulder Information File
      IF MSURE = "Y"
         APPEND BLANK
         REPLACE SID_NO WITH MSID_NO
         REPLACE STRUCNUM WITH MCONSTNU
         REPLACE PAVETYP  WITH MPAVETYP
         REPLACE LANEWID  WITH MLANEWID
         REPLACE OUTSHOWD WITH MOUTSHOW
         REPLACE SHOSFTYP WITH MSHOSFTY
         REPLACE SHOBSTYP WITH MSHOBSTY
         REPLACE SHOSFTHK WITH MSHOSFTH
         REPLACE SHOBSTHK WITH MSHOBSTH
         REPLACE WIDENFLG WITH MWIDEN
         REPLACE ERRORCHAR WITH .F.
      ELSE
         @ 20, 0 CLEAR
      ENDIF
   ENDDO
ENDDO
USE
CLEAR
RETURN



*****************
* PROCEDURE SURFACE
*****************
PROC SURFACE

*
* TO ENTER DATA INTO SURFACE dBASE III FILE
*
```

```
* set parameters and initialize variables
STORE .T. TO MAGAIN
CLEAR
STORE 0 TO MCONSTNU, MLAYNUM, MAGAPPLR
STORE "                " TO MADMXTYP
STORE 0 TO MADMXPER
STORE 0.0 TO MASAPPLR, MADMXPER
USE \PAVEDB\EDITUPDT\SURF_NEW INDEX \PAVEDB\EDITUPDT\SURF_NEW

DO WHILE MAGAIN
   CLEAR
   MSURE = " "
   DO WHILE .NOT. (MSURE $ 'Y')
      * set up menu and get data
      @  4, 20  SAY "   TEXAS FLEXIBLE PAVEMENT DATABASE"
      @  5, 20  SAY "    EDIT & UPDATE - Inventory Data"
      @  6, 20  SAY "         ADD  Surface Layer"
      @  8, 57  SAY "SID NUMBER "
      @  8, 68  SAY MSID_NO PICTURE "9999"
      @ 10,  3  SAY "                                                                    Percent"
      @ 11,  3  SAY "                          Aggregate                                Admixture
    Asphalt"
      @ 12,  3  SAY "Structure  Layer   Application       Type          (Mean
Asphalt  Application"
      @ 13,  3  SAY "Number     Number     Rate          Admixture      Content)
       Rate"
      @ 14,  3  SAY "---------  ------  ----------  ---------------
------------  -----------"
      @ 15,  6  GET  MCONSTNU PICTURE "99" RANGE 1,9
      @ 15, 16  GET  MLAYNUM PICTURE "99" RANGE 1,13
      @ 15, 26  GET  MAGAPPLR PICTURE "999" RANGE 0,200
      @ 15, 36  GET  MADMXTYP PICTURE "XXXXXXXXXXXXXXX"
      @ 15, 56  GET  MADMXPER PICTURE "9.99" RANGE 0.0,8.0
      @ 15, 71  GET  MASAPPLR PICTURE "9.99"  RANGE 0,.6
      @  2,  1  TO 17, 78    DOUBLE
      READ
      * set up escape key if want to abort
      IF READKEY() = 12
         CLEAR
         USE
         RETURN
      ENDIF
      @ 20, 23  SAY "Is The Data Entered Correct ? " GET MSURE PICT "!"
      READ

      IF MSURE = "Y"
         * if data so far is correct, add it to the surface file
         APPEND BLANK
         REPLACE SID_NO WITH MSID_NO
         REPLACE STRUCNUM WITH MCONSTNU
         REPLACE LAYNUM WITH MLAYNUM
         REPLACE AGAPPLRT WITH MAGAPPLR
         REPLACE ADMXTYP WITH MADMXTYP
```

```
            REPLACE ADMXPER WITH MADMXPER
            REPLACE ASAPPLRT WITH MASAPPLR
            REPLACE ERRORCHAR WITH .F.
            STORE 0 TO MCONSTNU, MLAYNUM, MAGAPPLR
            STORE "                " TO MADMXTYP
            STORE 0 TO MADMXPER
            STORE 0.0 TO MASAPPLR, MADMXPER
         ELSE
            @ 20, 0 CLEAR
         ENDIF
      ENDDO
   ENDDO

* clear all memory variables
USE
CLEAR
RETURN


************************
* PROCEDURE SUBGRADE
************************
PROC SUBGRADE

*
* TO ENTER DATA INTO THE SUBGRADE FILE
*

* set parameters and initialize all variables
CLEAR
STORE 0.0 TO MPPSV200, MPLASTIX, MLIQLIM, MTXTRIAX, MPERMIX
USE \PAVEDB\EDITUPDT\SUBG_NEW INDEX \PAVEDB\EDITUPDT\SUBG_NEW
STORE .T. TO MAGAIN

MSURE = " "
CLEAR
DO WHILE .NOT. (MSURE $ 'Y')
   * set up data entry screen
   @  3, 20  SAY "TEXAS FLEXIBLE PAVEMENT DATABASE"
   @  4, 20  SAY " EDIT & UPDATE - ADD Inventory"
   @  5, 20  SAY "          Subgrade File"
   @  7, 55  SAY "SID NUMBER"
   @  7, 66  SAY MSID_NO PICTURE "9999"
   @ 10, 15  SAY "Percent Passing No. 200 Sieve"
   @ 10, 47  GET  MPPSV200 PICTURE "99.9"
   @ 11, 15  SAY "Texas Triaxial Class"
   @ 11, 48  GET  MTXTRIAX PICTURE "9.9" RANGE 0.0,6.0
   @ 12, 15  SAY "Liquid Limit"
   @ 12, 47  GET  MLIQLIM PICTURE "99.9" RANGE 0.0,80.0
   @ 13, 15  SAY "Plasticity Index"
   @ 13, 47  GET  MPLASTIX  PICTURE "99.9" RANGE 0.0,70.0
   @ 14, 15  SAY "Permeability Index"
   @ 14, 46  GET  MPERMIX PICTURE "99.99" RANGE 0.00,10.00
   @  1,  8  TO 17, 75    DOUBLE
```

```
   * get data
   READ
   * set up the escape key if want to abort
   IF READKEY() = 12
      CLEAR
      USE
      MSURE = " "
      MAGAIN = .F.
      EXIT
   ENDIF
   * verify if data is correct
   @ 18, 23  SAY "Is The Data Correct ? "  GET MSURE PICT "!"
   READ
   IF MSURE = "Y"
      * add data to the subgrade file
      APPEND BLANK
      REPLACE SID_NO   WITH MSID_NO
      REPLACE STRUCNUM WITH 1
      REPLACE LAYNUM   WITH 1
      REPLACE PPSV200  WITH MPPSV200
      REPLACE PLASTIX  WITH MPLASTIX
      REPLACE LIQLIM   WITH MLIQLIM
      REPLACE TXTRIAXL WITH MTXTRIAX
      REPLACE PERMIX   WITH MPERMIX
      REPLACE ERRORCHAR WITH .F.
      STORE 0.0 TO MPPSV200, MPLASTIX, MLIQLIM, MTXTRIAX, MPERMIX
   ELSE
      @ 20, 0 CLEAR
   ENDIF
ENDDO
USE
CLEAR
RETURN


*******************
* PROCEDURE ADD LOCATION RECORD
*******************
PROCEDURE LOCATION
CLEAR
*
* To enter data into the transaction LOCATION identification file
*
STORE .T. TO MAGAIN
STORE 0 TO MHWYDIST,MCNTYNUM,MCONTROL,MSECTION,MHWYNUM,MBEGMPNT,MENDMPNT,MMPNTMO
STORE 0 TO MMPNTYR,MBEGMPST,MEMPSTDIS,MENDMPST,MEMPSTDIS,MFUNCLAS,MHPMSSEC
STORE 0 TO MINACTMO,MINACTYR,MPREVSID,MNEXTSID,MNUMLANES,MSID_NO
STORE "  " TO MHWYPREFX
STORE " " TO MHWYSUFFX,MLANEID,MBDISSIGN, MEDISSIGN
STORE "            " TO MHPMSSAM
STORE "                                        " TO MCOMMENT
STORE .T. TO MACTVFLAG

USE \PAVEDB\EDITUPDT\LOCN_NEW INDEX \PAVEDB\EDITUPDT\LOCN_NEW
```

```
DO WHILE MAGAIN
    CLEAR
    MSURE = " "
    DO WHILE .NOT. (MSURE $ 'Yy')
      * set up entry screen and get data
      @  2, 23  SAY "TEXAS FLEXIBLE PAVEMENT DATABASE"
      @  3, 23  SAY " EDIT & UPDATE - ADD Inventory"
      @  4, 23  SAY "         Location File"
      @  6,  6  SAY "SID Number"
      @  6, 18  GET  MSID_NO PICTURE "9999"
      @  6, 44  SAY "District"
      @  6, 53  GET  MHWYDIST PICTURE "99" RANGE 0, 25
      @  6, 59  SAY "County"
      @  6, 66  GET  MCNTYNUM PICTURE "999" RANGE 0, 500
      @  8,  6  SAY "Highway Ident."
      @  8, 21  GET  MHWYPREFX
      @  8, 24  GET  MHWYNUM PICTURE "9999"
      @  8, 29  GET  MHWYSUFFX
      @  8, 44  SAY "Control/Section"
      @  8, 60  GET  MCONTROL PICTURE "9999"
      @  8, 64  SAY "/"
      @  8, 65  GET  MSECTION PICTURE "99"
      @ 10,  6  SAY "Mile Post"
      @ 10, 17  GET  MBEGMPST PICTURE "999"
      @ 10, 21  GET  MBDISSIGN
      @ 10, 23  GET  MBMPSTDIS PICTURE "99"
      @ 10, 26  SAY "TO"
      @ 10, 29  GET  MENDMPST PICTURE "999"
      @ 10, 33  GET  MEDISSIGN
      @ 10, 35  GET  MEMPSTDIS PICTURE "99"
      @ 10, 44  SAY "Lane Identification"
      @ 10, 64  GET  MLANEID
      @ 12,  6  SAY "Mile Point"
      @ 12, 17  GET  MBEGMPNT PICTURE "99.999"
      @ 12, 24  SAY "TO"
      @ 12, 27  GET  MENDMPNT PICTURE "99.999"
      @ 12, 44  SAY "Mile Point Date"
      @ 12, 61  GET  MMPNTMO PICTURE "99"  RANGE 0, 12
      @ 12, 63  SAY "/"
      @ 12, 64  GET  MMPNTYR PICTURE "99"
      @ 14,  6  SAY "HPMS Sample Number"
      @ 14, 25  GET  MHPMSSAM
      @ 14, 44  SAY "HPMS Section Subdivision"
      @ 14, 69  GET  MHPMSSEC PICTURE "9"
      @ 16,  6  SAY "Functional Classification"
      @ 16, 32  GET  MFUNCLAS PICTURE "99"
      @ 16, 44  SAY "Number of Lanes"
      @ 16, 61  GET  MNUMLANES PICTURE "99"
      @ 18,  6  SAY "Active ?"
      @ 18, 15  GET  MACTVFLAG PICTURE "L"
      @ 18, 19  SAY "Inactive Date"
      @ 18, 33  GET  MINACTMO  PICTURE "99"  RANGE 0, 12
      @ 18, 35  SAY "/"
      @ 18, 36  GET  MINACTYR PICTURE "99"
```

```
      @ 18, 41  SAY "Previous SID"
      @ 18, 54  GET  MPREVSID PICTURE "9999"
      @ 18, 61  SAY "Next SID"
      @ 18, 70  GET  MNEXTSID PICTURE "9999"
      @ 20,  6  SAY "Comment"
      @ 20, 15  GET  MCOMMENT PICTURE
"XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX"
      @  1,  2  TO 21, 77    DOUBLE
      READ
      * set up the escape key if want to abort
      IF READKEY() = 12
           CLEAR
           USE
           RETURN
      ENDIF
      @ 22, 23 SAY "IS THE DATA CORRECT ? "  GET MSURE PICTURE "!"
      READ
      * if data is said to be not correct, the flags (sure and required) are
      * set to true so that the do loop will be executed again.
      IF MSURE = "Y"
         APPEND BLANK
         REPLACE SID_NO WITH MSID_NO
         REPLACE HWYDIST WITH MHWYDIST
         REPLACE CNTYNUM WITH MCNTYNUM
         REPLACE CONTROL  WITH MCONTROL
         REPLACE SECTION  WITH MSECTION
         REPLACE HWYPREFX WITH MHWYPREFX
         REPLACE HWYNUM  WITH MHWYNUM
         REPLACE HWYSUFFX WITH MHWYSUFFX
         REPLACE BEGMPNT WITH MBEGMPNT
         REPLACE ENDMPNT WITH MENDMPNT
         REPLACE MPNTMO WITH MMPNTMO
         REPLACE MPNTYR WITH MMPNTYR
         REPLACE LANEID WITH MLANEID
         REPLACE BEGMPST WITH MBEGMPST
         REPLACE BDISSIGN WITH MBDISSIGN
         REPLACE BMPSTDIS WITH MBMPSTDIS
         REPLACE ENDMPST WITH MENDMPST
         REPLACE EDISSIGN WITH MEDISSIGN
         REPLACE EMPSTDIS WITH MEMPSTDIS
         REPLACE FUNCLAS WITH MFUNCLAS
         REPLACE HPMSSAM WITH MHPMSSAM
         REPLACE HPMSSEC WITH MHPMSSEC
         REPLACE INACTMO WITH MINACTMO
         REPLACE INACTYR WITH MINACTYR
         REPLACE PREVSID WITH MPREVSID
         REPLACE NEXTSID WITH MNEXTSID
         REPLACE ACTVFLAG WITH MACTVFLAG
         REPLACE NUMLANES WITH MNUMLANES
         REPLACE COMMENT WITH MCOMMENT
         REPLACE ERRORCHAR WITH .F.
         STORE 0 TO
MHWYDIST,MCNTYNUM,MCONTROL,MSECTION,MHWYNUM,MBEGMPNT,MENDMPNT,MMPNTMO
         STORE 0 TO
```

```
MMPNTYR,MBEGMPST,MBMPSTDIS,MENDMPST,MEMPSTDIS,MFUNCLAS,MHPMSSEC
         STORE 0 TO MINACTMO,MINACTYR,MPREVSID,MNEXTSID, MNUMLANES,MSID_NO
         STORE "  " TO MHWYPREFX
         STORE " " TO MHWYSUFFX,MLANEID,MBDISSIGN, MEDISSIGN
         STORE "            " TO MHPMSSAM
         STORE "                                                " TO MCOMMENT
         STORE .T. TO MACTVFLAG
      ELSE
         @ 20, 0 CLEAR
      ENDIF
    ENDDO
ENDDO
USE
CLEAR
RETURN
```

```
*
* SUBSYSTEM:        EDIT & UPDATE
* PROGRAM NAME:     INV_BKUP.PRG              06/23/88
* CALLED FROM:      ENTRLAYR.PRG
* PROJECT 2456 -    TEXAS FLEXIBLE PAVEMENT DATABASE CONVERSION
* TAMU/TTI
* AUTHOR:           TREVOR X. PEREIRA
* PURPOSE:          BACK UP MASTER INVENTORY FILES (ORIGINAL FILES) AND
*                      TRANSACTION INVENTORY FILES (NEWLY ENTERED DATA)
*


SET TALK OFF
SET ECHO OFF
SET ESCAPE OFF
CLEAR
CLEAR TYPE
STORE 0 TO MSIZE, MNUMFIELDS, MHEADER, MTOTALSIZE, MCOUNT
STORE "" TO MNULL
MBACKDRV = MDRIVE2 + ":"
@ 5,5 say "The new data you entered is about to be checked for errors."
@ 8,5 SAY "First, the Inventory files will be backed up.  For this,"
@ 9,5 say "place a FORMATTED DISKETTE in drive " + MBACKDRV + " and close the
drive."
@ 12,5 say "If you DO NOT want to continue, press the ESC key."
@ 13,5 say "Otherwise press any other key to continue."
READ
IF READKEY() = 12
   CLEAR
   RETURN
ENDIF
CLEAR
SET TYPE TO 0

* backing up LOCATION files
USE \PAVEDB\EDITUPDT\LOCN_NEW INDEX \PAVEDB\EDITUPDT\LOCN_NEW
MCOUNT = RECCOUNT()
USE
IF MCOUNT <> 0
   CLEAR
   @ 9,5 SAY "Please wait.  Backing up LOCATION files . . ."
   SET DEFAUL TO &MBACKDRV
   IF FILE('LOCATION.DBF')
     DELE FILE LOCATION.DBF
   ENDIF

   SET DEFAU TO &MDRIVE
   * Checking diskspace on backup drive and backing up LOCATION master files
   USE \PAVEDB\FILES\LOCATION
   STORE RECCOUNT() * RECSIZE() TO MSIZE
```

```
   MNULL = ""
   DO WHILE MNULL < FIELD(MNUMFIELDS + 1)
      MNUMFIELDS = MNUMFIELDS + 1
   ENDDO
   MHEADER = (32 * MNUMFIELDS) + 34
   MTOTALSIZE = MSIZE + MHEADER + 20
   SET DEFAULT TO &MBACKDRV
   MDISKSPACE = DISKSPACE()
   SET DEFAULT TO &MDRIVE
   DO WHILE MDISKSPACE < MTOTALSIZE
     ? "Not enough space on diskette in drive " + MBACKDRV
     ? "Please replace with another diskette and press any key to continue"
     WAIT " "
     IF READKEY() = 12
        ? "PROCESS ABORTED"
        RETURN
     ENDIF
     SET DEFAULT TO &MBACKDRV
     MDISKSPACE = DISKSPACE()
     SET DEFAULT TO &MDRIVE
   ENDDO
   USE
   IF FILE('\pavedb\files\backup\LOCATION.DBF')
     DELE FILE \PAVEDB\FILES\BACKUP\LOCATION.DBF
     SET TALK ON
   ENDIF
   COPY FILE \PAVEDB\FILES\LOCATION.DBF TO &MBACKDRV\LOCATION.DBF
   COPY FILE \PAVEDB\FILES\LOCATION.DBF TO \PAVEDB\FILES\BACKUP\LOCATION.DBF

   * Backing up LOCATION  transaction files i.e. newly entered data
   * to drive Backup directory
   IF FILE('\PAVEDB\EDITUPDT\BACKUP\LOCN_OLD.DBF')
      SET TALK OFF
      DELE FILE \PAVEDB\EDITUPDT\BACKUP\LOCN_OLD.DBF
      SET TALK ON
   ENDIF
   IF FILE('\PAVEDB\EDITUPDT\BACKUP\LOCN_NEW.DBF')
      RENA \PAVEDB\EDITUPDT\BACKUP\LOCN_NEW.DBF TO
\PAVEDB\EDITUPDT\BACKUP\LOCN_OLD.DBF
   ENDIF
   COPY FILE \PAVEDB\EDITUPDT\LOCN_NEW.DBF TO
\PAVEDB\EDITUPDT\BACKUP\LOCN_NEW.DBF
   CLEAR
   SET TALK OFF
ENDIF

* backing up layer id files
USE \PAVEDB\EDITUPDT\LAYR_NEW INDEX \PAVEDB\EDITUPDT\LAYR_NEW
MCOUNT = RECCOUNT()
USE
IF MCOUNT <> 0
   CLEAR
   @ 9,5 SAY "Please wait.  Backing up Layer ID . . ."
   SET DEFAU TO &MBACKDRV
```

```
   IF FILE('LAYER.DBF')
     DELE FILE LAYER.DBF
   ENDIF

   * Checking diskspace on Backup drive and backing up Layer ID master files
   SET DEFAU TO &MDRIVE
   USE \PAVEDB\FILES\LAYER
   STORE RECCOUNT() * RECSIZE() TO MSIZE
   MNULL = ""
   DO WHILE MNULL < FIELD(MNUMFIELDS + 1)
      MNUMFIELDS = MNUMFIELDS + 1
   ENDDO
   MHEADER = (32 * MNUMFIELDS) + 34
   MTOTALSIZE = MSIZE + MHEADER + 20
   SET DEFAULT TO &MBACKDRV
   MDISKSPACE = DISKSPACE()
   SET DEFAULT TO &MDRIVE
   DO WHILE MDISKSPACE < MTOTALSIZE
     ? "Not enough space on diskette in drive " + MBACKDRV
     ? "Please replace with another diskette and press any key to continue"
     WAIT " "
     IF READKEY() = 12
        ? "PROCESS ABORTED"
        RETURN
     ENDIF
     SET DEFAULT TO &MBACKDRV
     MDISKSPACE = DISKSPACE()
     SET DEFAULT TO &MDRIVE
   ENDDO
   USE
   IF FILE('\pavedb\files\backup\LAYER.DBF')
     DELE FILE \PAVEDB\FILES\BACKUP\LAYER.DBF
   ENDIF
   SET TALK ON
   COPY FILE \PAVEDB\FILES\LAYER.DBF TO &MBACKDRV\LAYER.DBF
   COPY FILE \PAVEDB\FILES\LAYER.DBF TO \PAVEDB\FILES\BACKUP\LAYER.DBF

   * Backing up Layer ID transaction files i.e. newly entered data
   * to Backup directory
   IF FILE('\PAVEDB\EDITUPDT\BACKUP\LAYR_OLD.DBF')
      SET TALK OFF
      DELE FILE \PAVEDB\EDITUPDT\BACKUP\LAYR_OLD.DBF
      SET TALK ON
   ENDIF
   IF FILE('\PAVEDB\EDITUPDT\BACKUP\LAYR_NEW.DBF')
      RENA \PAVEDB\EDITUPDT\BACKUP\LAYR_NEW.DBF TO
\PAVEDB\EDITUPDT\BACKUP\LAYR_OLD.DBF
   ENDIF
   COPY FILE \PAVEDB\EDITUPDT\LAYR_NEW.DBF TO
\PAVEDB\EDITUPDT\BACKUP\LAYR_NEW.DBF
   CLEAR
   SET TALK OFF
ENDIF
```

```
* backing up LAYTHICK THICKNESS files
USE \PAVEDB\EDITUPDT\LAYT_NEW INDEX \PAVEDB\EDITUPDT\LAYT_NEW
MCOUNT = RECCOUNT()
USE
IF MCOUNT <> 0
   CLEAR
   @ 9,5 SAY "Please wait.  Backing up LAYER THICKNESS Across the Road files . .
."
   SET DEFAU TO &MBACKDRV
   IF FILE('LAYTHICK.DBF')
     DELE FILE LAYTHICK.DBF
   ENDIF

   * Checking diskspace on Backup drive and backing up LAYTHICK ID master files
   SET DEFAU TO &MDRIVE
   USE \PAVEDB\FILES\LAYTHICK
   STORE RECCOUNT() * RECSIZE() TO MSIZE
   MNULL = ""
   DO WHILE MNULL < FIELD(MNUMFIELDS + 1)
      MNUMFIELDS = MNUMFIELDS + 1
   ENDDO
   MHEADER = (32 * MNUMFIELDS) + 34
   MTOTALSIZE = MSIZE + MHEADER + 20
   SET DEFAULT TO &MBACKDRV
   MDISKSPACE = DISKSPACE()
   SET DEFAULT TO &MDRIVE
   DO WHILE MDISKSPACE < MTOTALSIZE
     ? "Not enough space on diskette in drive " + MBACKDRV
     ? "Please replace with another diskette and press any key to continue"
     WAIT " "
     IF READKEY() = 12
        ? "PROCESS ABORTED"
        RETURN
     ENDIF
     SET DEFAULT TO &MBACKDRV
     MDISKSPACE = DISKSPACE()
     SET DEFAULT TO &MDRIVE
   ENDDO
   USE
   IF FILE('\pavedb\files\backup\LAYTHICK.DBF')
     DELE FILE \PAVEDB\FILES\BACKUP\LAYTHICK.DBF
   ENDIF
   SET TALK ON
   COPY FILE \PAVEDB\FILES\LAYTHICK.DBF TO &MBACKDRV\LAYTHICK.DBF
   COPY FILE \PAVEDB\FILES\LAYTHICK.DBF TO \PAVEDB\FILES\BACKUP\LAYTHICK.DBF

   * Backing up LAYTHICK  transaction files i.e. newly entered data
   * to Backup directory
   IF FILE('\PAVEDB\EDITUPDT\BACKUP\LAYT_OLD.DBF')
      SET TALK OFF
      DELE FILE \PAVEDB\EDITUPDT\BACKUP\LAYT_OLD.DBF
      SET TALK ON
   ENDIF
   IF FILE('\PAVEDB\EDITUPDT\BACKUP\LAYT_NEW.DBF')
```

```
      RENA \PAVEDB\EDITUPDT\BACKUP\LAYT_NEW.DBF TO
\PAVEDB\EDITUPDT\BACKUP\LAYT_OLD.DBF
   ENDIF
   COPY FILE \PAVEDB\EDITUPDT\LAYT_NEW.DBF TO
\PAVEDB\EDITUPDT\BACKUP\LAYT_NEW.DBF
   CLEAR
   SET TALK OFF
ENDIF

* backing up GEOMETRIC & SHOULDER files
USE \PAVEDB\EDITUPDT\GEOS_NEW INDEX \PAVEDB\EDITUPDT\GEOS_NEW
MCOUNT = RECCOUNT()
USE
IF MCOUNT <> 0
   CLEAR
   @ 9,5 SAY "Please wait.  Backing up GEOMETRIC & SHOULDER files . . ."
   SET DEFAU TO &MBACKDRV
   IF FILE('GEOSHO.DBF')
     DELE FILE GEOSHO.DBF
   ENDIF

   * Checking diskspace on Backup drive and backing up GEOMETRIC & SHOULDER
master files
   SET DEFAU TO &MDRIVE
   USE \PAVEDB\FILES\GEOSHO
   STORE RECCOUNT() * RECSIZE() TO MSIZE
   MNULL = ""
   DO WHILE MNULL < FIELD(MNUMFIELDS + 1)
      MNUMFIELDS = MNUMFIELDS + 1
   ENDDO
   MHEADER = (32 * MNUMFIELDS) + 34
   MTOTALSIZE = MSIZE + MHEADER + 20
   SET DEFAULT TO &MBACKDRV
   MDISKSPACE = DISKSPACE()
   SET DEFAULT TO &MDRIVE
   DO WHILE MDISKSPACE < MTOTALSIZE
     ? "Not enough space on diskette in drive " + MBACKDRV
     ? "Please replace with another diskette and press any key to continue"
     WAIT " "
     IF READKEY() = 12
        ? "PROCESS ABORTED"
        RETURN
     ENDIF
     SET DEFAULT TO &MBACKDRV
     MDISKSPACE = DISKSPACE()
     SET DEFAULT TO &MDRIVE
   ENDDO
   USE
   IF FILE('\pavedb\files\backup\GEOSHO.DBF')
     DELE FILE \PAVEDB\FILES\BACKUP\GEOSHO.DBF
     SET TALK ON
   ENDIF
   COPY FILE \PAVEDB\FILES\GEOSHO.DBF TO &MBACKDRV\GEOSHO.DBF
   COPY FILE \PAVEDB\FILES\GEOSHO.DBF TO \PAVEDB\FILES\BACKUP\GEOSHO.DBF
```

```
   * Backing up GEOSHO  transaction files i.e. newly entered data
   * to drive Backup directory
   IF FILE('\PAVEDB\EDITUPDT\BACKUP\GEOS_OLD.DBF')
      SET TALK OFF
      DELE FILE \PAVEDB\EDITUPDT\BACKUP\GEOS_OLD.DBF
      SET TALK ON
  ENDIF
   IF FILE('\PAVEDB\EDITUPDT\BACKUP\GEOS_NEW.DBF')
      RENA \PAVEDB\EDITUPDT\BACKUP\GEOS_NEW.DBF TO
\PAVEDB\EDITUPDT\BACKUP\GEOS_OLD.DBF
   ENDIF
   COPY FILE \PAVEDB\EDITUPDT\GEOS_NEW.DBF TO
\PAVEDB\EDITUPDT\BACKUP\GEOS_NEW.DBF
   CLEAR
   SET TALK OFF
ENDIF

* backing up SURFACE THICKNESS files
USE \PAVEDB\EDITUPDT\SURF_NEW INDEX \PAVEDB\EDITUPDT\SURF_NEW
MCOUNT = RECCOUNT()
USE
IF MCOUNT <> 0
   CLEAR
   @ 9,5 SAY "Please wait.  Backing up SURFACE files . . ."
   SET DEFAU TO &MBACKDRV
   IF FILE('SURFACE.DBF')
     DELE FILE SURFACE.DBF
   ENDIF

   * Checking diskspace on Backup drive and backing up SURFACE master files
   SET DEFAU TO &MDRIVE
   USE \PAVEDB\FILES\SURFACE
   STORE RECCOUNT() * RECSIZE() TO MSIZE
   MNULL = ""
   DO WHILE MNULL < FIELD(MNUMFIELDS + 1)
      MNUMFIELDS = MNUMFIELDS + 1
   ENDDO
   MHEADER = (32 * MNUMFIELDS) + 34
   MTOTALSIZE = MSIZE + MHEADER + 20
   SET DEFAULT TO &MBACKDRV
   MDISKSPACE = DISKSPACE()
   SET DEFAULT TO &MDRIVE
   DO WHILE MDISKSPACE < MTOTALSIZE
     ? "Not enough space on diskette in drive " + MBACKDRV
     ? "Please replace with another diskette and press any key to continue"
     WAIT " "
     IF READKEY() = 12
        ? "PROCESS ABORTED"
        RETURN
     ENDIF
     SET DEFAULT TO &MBACKDRV
     MDISKSPACE = DISKSPACE()
     SET DEFAULT TO &MDRIVE
```

```
   ENDDO
   USE
   IF FILE('\pavedb\files\backup\SURFACE.DBF')
     DELE FILE \PAVEDB\FILES\BACKUP\SURFACE.DBF
   ENDIF
   SET TALK ON
   COPY FILE \PAVEDB\FILES\SURFACE.DBF TO &MBACKDRV\SURFACE.DBF
   COPY FILE \PAVEDB\FILES\SURFACE.DBF TO \PAVEDB\FILES\BACKUP\SURFACE.DBF

   * Backing up SURFACE  transaction files i.e. newly entered data
   * to drive Backup directory
   IF FILE('\PAVEDB\EDITUPDT\BACKUP\SURF_OLD.DBF')
      SET TALK OFF
      DELE FILE \PAVEDB\EDITUPDT\BACKUP\SURF_OLD.DBF
      SET TALK ON
   ENDIF
   IF FILE('\PAVEDB\EDITUPDT\BACKUP\SURF_NEW.DBF')
      RENA \PAVEDB\EDITUPDT\BACKUP\SURF_NEW.DBF TO
\PAVEDB\EDITUPDT\BACKUP\SURF_OLD.DBF
   ENDIF
   COPY FILE \PAVEDB\EDITUPDT\SURF_NEW.DBF TO
\PAVEDB\EDITUPDT\BACKUP\SURF_NEW.DBF
   CLEAR
   SET TALK OFF
ENDIF

* backing up SUBGRADE THICKNESS files
USE \PAVEDB\EDITUPDT\SUBG_NEW INDEX \PAVEDB\EDITUPDT\SUBG_NEW
MCOUNT = RECCOUNT()
USE
IF MCOUNT <> 0
   CLEAR
   @ 9,5 SAY "Please wait.  Backing up SUBGRADE files . . ."
   SET TALK OFF
   SET DEFAU TO &MBACKDRV
   IF FILE('SUBGRADE.DBF')
     DELE FILE SUBGRADE.DBF
   ENDIF

   * Checking diskspace on Backup drive and backing up SUBGRADE master files
   SET DEFAU TO &MDRIVE
   USE \PAVEDB\FILES\SUBGRADE
   STORE RECCOUNT() * RECSIZE() TO MSIZE
   MNULL = ""
   DO WHILE MNULL < FIELD(MNUMFIELDS + 1)
      MNUMFIELDS = MNUMFIELDS + 1
   ENDDO
   MHEADER = (32 * MNUMFIELDS) + 34
   MTOTALSIZE = MSIZE + MHEADER + 20
   SET DEFAULT TO &MBACKDRV
   MDISKSPACE = DISKSPACE()
   SET DEFAULT TO &MDRIVE
   DO WHILE MDISKSPACE < MTOTALSIZE
     ? "Not enough space on diskette in drive " + MBACKDRV
```

```
      ? "Please replace with another diskette and press any key to continue"
      WAIT " "
      IF READKEY() = 12
         ? "PROCESS ABORTED"
         RETURN
      ENDIF
      SET DEFAULT TO &MBACKDRV
      MDISKSPACE = DISKSPACE()
      SET DEFAULT TO &MDRIVE
   ENDDO
   USE
   IF FILE('\pavedb\files\backup\SUBGRADE.DBF')
     DELE FILE \PAVEDB\FILES\BACKUP\SUBGRADE.DBF
   ENDIF
   SET TALK ON
   COPY FILE \PAVEDB\FILES\SUBGRADE.DBF TO &MBACKDRV\SUBGRADE.DBF
   COPY FILE \PAVEDB\FILES\SUBGRADE.DBF TO \PAVEDB\FILES\BACKUP\SUBGRADE.DBF

   * Backing up SUBGRADE  transaction files i.e. newly entered data
   * to drive Backup directory
   IF FILE('\PAVEDB\EDITUPDT\BACKUP\SUBG_OLD.DBF')
      SET TALK OFF
      DELE FILE \PAVEDB\EDITUPDT\BACKUP\SUBG_OLD.DBF
      SET TALK ON
   ENDIF
   IF FILE('\PAVEDB\EDITUPDT\BACKUP\SUBG_NEW.DBF')
      RENA \PAVEDB\EDITUPDT\BACKUP\SUBG_NEW.DBF TO
\PAVEDB\EDITUPDT\BACKUP\SUBG_OLD.DBF
   ENDIF
   COPY FILE \PAVEDB\EDITUPDT\SUBG_NEW.DBF TO
\PAVEDB\EDITUPDT\BACKUP\SUBG_NEW.DBF
   CLEAR
   SET TALK OFF
ENDIF

SET ESCAPE OFF
SET TALK OFF
SET ECHO OFF
SET TYPE TO 20
DO \PAVEDB\EDITUPDT\INV_UPDT
RETURN
```

```
*
* SUBSYSTEM:        EDIT & UPDATE SUBSYSTEM
* PROGRAM NAME:     INV_UPDT.PRG              03/23/88
* CALLED FROM:      INV_BKUP.PRG
* PROJECT 2456 -    TEXAS FLEXIBLE PAVEMENT DATABASE CONVERSION
* TAMU/TTI
* AUTHOR:           TREVOR X. PEREIRA
* PURPOSE:          TO TO CHECK NEW DATA ENTERED IN EACH OF THE 5 INVENTORY FILES
*
* LIST OF PROGRAMS CALLED FROM THE FOLLOWING PROCEDURES:
*      LAYRCHEK.PRG - LAYRCHK3.PRG, LAYRSVER.PRG, SIDCHEK1.PRG, LAYDESC.PRG
*      LAYTCHEK.PRG - LAYTCHK1.PRG, LAYTCHK2.PRG, LAYTCHK3.PRG, SIDCHEK2.PRG
*      GEOSCHEK.PRG - GEOSCHK1.PRG, GEOSCHK2.PRG, GEOXCHK3.PRG, SIDCHEK3.PRG
*      SUBGCHEK.PRG - SUBGCHK1.PRG, SUBGCHK2.PRG, SUBGCHK3.PRG, SIDCHEK4.PRG
*      SURFCHEK.PRG - SURFCHK1.PRG, SURFCHK2.PRG, SURFCHK3.PRG, SIDCHEK5.PRG
*
*

SET ALTE TO \PAVEDB\EDITUPDT\ERRORS
SET ALTE OFF

*
* Checking LOCATION File
*
USE \PAVEDB\EDITUPDT\LOCN_NEW INDEX \PAVEDB\EDITUPDT\LOCN_NEW
MCOUNT = RECCOUNT()
USE
IF MCOUNT <> 0
     * Running the edit/check programs
     CLEAR
     @ 5,5 say "The new LOCATION data you entered is"
     @ 6,5 SAY "being checked for errors."
     USE \PAVEDB\EDITUPDT\LOCN_NEW INDEX \PAVEDB\EDITUPDT\LOCN_NEW
     DO WHILE .NOT. EOF()
       REPLACE ERRORCHAR WITH .F.
       SKIP
     ENDDO
     USE
     DO \PAVEDB\EDITUPDT\LOCNCHEK

     CLEAR
     @ 5, 5 SAY "Please Wait.  Updating Master Database . . . ."
     * Appending data to the Master database if there are no errors
     SELE 1
       USE \PAVEDB\EDITUPDT\LOCN_NEW INDEX \PAVEDB\EDITUPDT\LOCN_NEW
     SELE 2
       USE \PAVEDB\FILES\LOCATION INDEX \PAVEDB\INDEXES\LOCSID
     SELE 1
     DO WHILE .NOT. EOF()
       IF .NOT. ERRORCHAR
```

```
            SELE 2
            APPEND BLANK
            REPLACE SID_NO WITH A->SID_NO
            REPLACE HWYDIST WITH A->HWYDIST
            REPLACE CNTYNUM WITH A->CNTYNUM
            REPLACE CONTROL  WITH A->CONTROL
            REPLACE SECTION  WITH A->SECTION
            REPLACE HWYPREFX WITH A->HWYPREFX
            REPLACE HWYNUM  WITH A->HWYNUM
            REPLACE HWYSUFFX WITH A->HWYSUFFX
            REPLACE BEGMPNT WITH A->BEGMPNT
            REPLACE ENDMPNT WITH A->ENDMPNT
            REPLACE MPNTIMO WITH A->MPNTIMO
            REPLACE MPNTYR WITH A->MPNTYR
            REPLACE LANEID WITH A->LANEID
            REPLACE BEGMPST WITH A->BEGMPST
            REPLACE BDISSIGN WITH A->BDISSIGN
            REPLACE BMPSTDIS WITH A->BMPSTDIS
            REPLACE ENDMPST WITH A->ENDMPST
            REPLACE EDISSIGN WITH A->EDISSIGN
            REPLACE EMPSTDIS WITH A->EMPSTDIS
            REPLACE FUNCLAS WITH A->FUNCLAS
            REPLACE HPMSSAM WITH A->HPMSSAM
            REPLACE HPMSSEC WITH A->HPMSSEC
            REPLACE INACTMO WITH A->INACTMO
            REPLACE INACTYR WITH A->INACTYR
            REPLACE PREVSID WITH A->PREVSID
            REPLACE NEXTSID WITH A->NEXTSID
            REPLACE ACTVFLAG WITH A->ACTVFLAG
            REPLACE NUMLANES WITH A->NUMLANES
            REPLACE COMMENT WITH A->COMMENT
            SELE 1
            DELE
          ENDIF
          SKIP
        ENDDO
        SELE 1
        PACK
ENDIF
SELE 1
USE
SELE 2
USE

*
* Checking layer id file
*
USE \PAVEDB\EDITUPDT\LAYR_NEW INDEX \PAVEDB\EDITUPDT\LAYR_NEW
MCOUNT = RECCOUNT()
USE
IF MCOUNT <> 0
   @ 5,5 say "The new LAYER IDENTIFICATION data you entered is"
   @ 6,5 SAY "being checked for errors."
   * Running the edit/check programs
```

```
USE \PAVEDB\EDITUPDT\LAYR_NEW INDE \PAVEDB\EDITUPDT\LAYR_NEW
DO WHILE .NOT. EOF()
  REPLACE ERRORCHAR WITH .F.
  SKIP
ENDDO
USE
SET PROC TO \PAVEDB\EDITUPDT\LAYRCHEK
DO LAYRCHK3
DO LAYRSVER
DO SIDCHEK1
DO LOCCHEK1
DO LAYDESC
CLEAR
@ 5, 5 SAY "Please Wait.  Updating Master Database . . . ."
* Appending data to the Master database if there are no errors
SELE 1
  USE \PAVEDB\EDITUPDT\LAYR_NEW INDEX \PAVEDB\EDITUPDT\LAYR_NEW
SELE 2
  USE \PAVEDB\FILES\LAYER INDEX \PAVEDB\INDEXES\LAYNDX
SELE 1
DO WHILE .NOT. EOF()
  * If there is any error in any record for a SID number then do not
  * add it to the master file
  MERROR = 0
  MTESTSID = A->SID_NO
  DO WHILE .NOT. EOF() .AND. A->SID_NO = MTESTSID
     IF A->ERRORCHAR
         MERROR = MERROR + 1
     ENDIF
     SKIP
  ENDDO
  IF MERROR = 0
    LOCATE FOR SID_NO = MTESTSID
    DO WHILE .NOT. EOF() .AND. A->SID_NO = MTESTSID
     SELE 2
     APPEND BLANK
     REPLACE SID_NO   WITH A->SID_NO
     REPLACE STRUCNUM WITH A->STRUCNUM
     REPLACE LAYNUM   WITH A->LAYNUM
     REPLACE LAYDESC  WITH A->LAYDESC
     REPLACE CENTITHK  WITH A->CENTITHK
     REPLACE LAYMATCL WITH A->LAYMATCL
     REPLACE JOBCMPYR WITH A->JOBCMPYR
     REPLACE JOBCMPMO WITH A->JOBCMPMO
     REPLACE WIDENLYR WITH A->WIDENLYR
     REPLACE WIDENLMO WITH A->WIDENLMO
     SELE 1
     DELE
     SKIP
    ENDDO
  ENDIF
ENDDO
SELE 1
PACK
```

```
ENDIF
SELE 1
USE
SELE 2
USE

*
* Checking Layer Thickness Across the Road File
*
USE \PAVEDB\EDITUPDT\LAYT_NEW INDEX \PAVEDB\EDITUPDT\LAYT_NEW
MCOUNT = RECCOUNT()
USE
IF MCOUNT <> 0
     @ 5,5 say "The new LAYER THICKNESS ACROSS THE ROAD data you entered is"
     @ 6,5 SAY "being checked for errors."
     * Running the edit/check programs
     USE \PAVEDB\EDITUPDT\LAYT_NEW INDEX \PAVEDB\EDITUPDT\LAYT_NEW
     DO WHILE .NOT. EOF()
       REPLACE ERRORCHAR WITH .F.
       SKIP
     ENDDO
     USE
     SET PROC TO \PAVEDB\EDITUPDT\LAYTCHEK
     DO LAYTCHK1
     DO LAYTCHK3
     DO SIDCHEK2
     CLEAR
     @ 5, 5 SAY "Please Wait.  Updating Master Database . . . ."
     * Appending data to the Master database if there are no errors
     SELE 1
       USE \PAVEDB\EDITUPDT\LAYT_NEW INDEX \PAVEDB\EDITUPDT\LAYT_NEW
     SELE 2
       USE \PAVEDB\FILES\LAYTHICK INDEX \PAVEDB\INDEXES\LAYTNDX
     SELE 1
     DO WHILE .NOT. EOF()
       IF .NOT. ERRORCHAR
          SELE 2
          APPEND BLANK
          REPLACE SID_NO   WITH A->SID_NO
          REPLACE STRUCNUM WITH A->STRUCNUM
          REPLACE LAYNUM   WITH A->LAYNUM
          REPLACE FC3THK   WITH A->FC3THK
          REPLACE FC2THK   WITH A->FC2THK
          REPLACE FC1THK   WITH A->FC1THK
          REPLACE CENTTHK  WITH A->CENTTHK
          REPLACE FC3DIS   WITH A->FC3DIS
          REPLACE FC2DIS   WITH A->FC2DIS
          REPLACE FC1DIS   WITH A->FC1DIS
          SELE 1
          DELE
       ENDIF
       SKIP
     ENDDO
     SELE 1
```

```
     PACK
ENDIF
SELE 1
USE
SELE 2
USE


*
* Checking Geometric & Shoulder Information File
*
USE \PAVEDB\EDITUPDT\GEOS_NEW INDEX \PAVEDB\EDITUPDT\GEOS_NEW
MCOUNT = RECCOUNT()
USE
IF MCOUNT <> 0
     * Running the edit/check programs
     CLEAR
     @ 5,5 say "The new GEOMETRIC & SHOULDER data you entered is"
     @ 6,5 SAY "being checked for errors."
     USE \PAVEDB\EDITUPDT\GEOS_NEW INDEX \PAVEDB\EDITUPDT\GEOS_NEW
     DO WHILE .NOT. EOF()
       REPLACE ERRORCHAR WITH .F.
       SKIP
     ENDDO
     USE
     SET PROC TO \PAVEDB\EDITUPDT\GEOSCHEK
     DO GEOSCHK1
     DO GEOSCHK3
     DO SIDCHEK3
     SET PROC TO

     CLEAR
     @ 5, 5 SAY "Please Wait.  Updating Master Database . . . ."
     * Appending data to the Master database if there are no errors
     SELE 1
       USE
       USE \PAVEDB\EDITUPDT\GEOS_NEW INDEX \PAVEDB\EDITUPDT\GEOS_NEW
     SELE 2
       USE
       USE \PAVEDB\FILES\GEOSHO INDEX \PAVEDB\INDEXES\GEONDX
     SELE 1
     DO WHILE .NOT. EOF()
       IF .NOT. ERRORCHAR
          SELE 2
          APPEND BLANK
          REPLACE SID_NO    WITH A->SID_NO
          REPLACE STRUCNUM WITH A->STRUCNUM
          REPLACE PAVETYP  WITH A->PAVETYP
          REPLACE LANEWID  WITH A->LANEWID
          REPLACE OUTSHOWD WITH A->OUTSHOWD
          REPLACE SHOSFTYP WITH A->SHOSFTYP
          REPLACE SHOBSTYP WITH A->SHOBSTYP
          REPLACE SHOSFTHK WITH A->SHOSFTHK
          REPLACE SHOBSTHK WITH A->SHOBSTHK
```

```
            REPLACE WIDENFLG WITH A->WIDENFLG
            SELE 1
            DELE
          ENDIF
          SKIP
        ENDDO
        SELE 1
        PACK
ENDIF
SELE 1
USE
SELE 2
USE


*
* Checking Surface Information File
*
USE \PAVEDB\EDITUPDT\SURF_NEW INDEX \PAVEDB\EDITUPDT\SURF_NEW
MCOUNT = RECCOUNT()
USE
IF MCOUNT <> 0
     clear
     @ 5,5 say "The new Surface Information File data you entered is"
     @ 6,5 SAY "being checked for errors."

     * Running the edit/check programs
     CLEAR
     USE \PAVEDB\EDITUPDT\SURF_NEW INDEX \PAVEDB\EDITUPDT\SURF_NEW
     DO WHILE .NOT. EOF()
       REPLACE ERRORCHAR WITH .F.
       SKIP
     ENDDO
     USE
     SET PROC TO \PAVEDB\EDITUPDT\SURFCHEK
     DO SURFCHK1
     DO SURFCHK3
     DO SIDCHEK4
     CLEAR
     @ 5, 5 SAY "Please Wait.  Updating Master Database . . . ."
     * Appending error-free SID's to the Master database
     SELE 1
       USE
       USE \PAVEDB\EDITUPDT\SURF_NEW INDEX \PAVEDB\EDITUPDT\SURF_NEW
     SELE 2
       USE
       USE \PAVEDB\FILES\SURFACE INDEX \PAVEDB\INDEXES\SURFNDX
     SELE 1
     DO WHILE .NOT. EOF()
       IF .NOT. A->ERRORCHAR
          SELE 2
          APPEND BLANK
          REPLACE SID_NO   WITH A->SID_NO
          REPLACE STRUCNUM WITH A->STRUCNUM
```

```
            REPLACE LAYNUM   WITH A->LAYNUM
            REPLACE AGAPPLRT WITH A->AGAPPLRT
            REPLACE ADMXTYP  WITH A->ADMXTYP
            REPLACE ADMXPER  WITH A->ADMXPER
            REPLACE ASAPPLRT WITH A->ASAPPLRT
            SELE 1
            DELE
          ENDIF
          SKIP
        ENDDO
        SELE 1
        PACK
ENDIF
SELE 1
USE
SELE 2
USE


*
* Checking Subgrade File
*
USE \PAVEDB\EDITUPDT\SUBG_NEW INDEX \PAVEDB\EDITUPDT\SUBG_NEW
MCOUNT = RECCOUNT()
USE
IF MCOUNT <> 0
     clear
     CLEAR TYPE
     @ 5,5 say "The new SUBGRADE FILE data you entered is"
     @ 6,5 SAY "being checked for errors."

     * Running the edit/check programs
     CLEAR
     USE \PAVEDB\EDITUPDT\SUBG_NEW INDEX \PAVEDB\EDITUPDT\SUBG_NEW
     DO WHILE .NOT. EOF()
       REPLACE ERRORCHAR WITH .F.
       SKIP
     ENDDO
     USE
     SET PROC TO \PAVEDB\EDITUPDT\SUBGCHEK
     DO SUBGCHK1
     DO SUBGCHK3
     DO SIDCHEK5

     CLEAR
     @ 5, 5 SAY "Please Wait.  Updating Master Database . . . ."
     * Appending data to the Master database if there are no errors
     SELE 1
       USE
       USE \PAVEDB\EDITUPDT\SUBG_NEW INDEX \PAVEDB\EDITUPDT\SUBG_NEW
     SELE 2
       USE
       USE \PAVEDB\FILES\SUBGRADE INDEX \PAVEDB\INDEXES\SUBGNDX
     SELE 1
```

```
      DO WHILE .NOT. EOF()
        IF .NOT. ERRORCHAR
           SELE 2
           APPEND BLANK
           REPLACE SID_NO   WITH A->SID_NO
           REPLACE STRUCNUM WITH A->STRUCNUM
           REPLACE LAYNUM   WITH A->LAYNUM
           REPLACE PPSV200  WITH A->PPSV200
           REPLACE PLASTIX  WITH A->PLASTIX
           REPLACE LIQLIM   WITH A->LIQLIM
           REPLACE TXTRIAXL WITH A->TXTRIAXL
           REPLACE PERMIX   WITH A->PERMIX

           SELE 1
           DELE
        ENDIF
        SKIP
      ENDDO
      SELE 1
      PACK
ENDIF
SELE 1
USE
SELE 2
USE
SELE 1

CLOSE ALTE
SET PRINT ON
? "           LIST OF ERRORS IN NEWLY ENTERED INVENTORY DATA FILES"
? "           ----------------------------------------------------"
TYPE  \PAVEDB\EDITUPDT\ERRORS.TXT
SET PRINT OFF
RETURN
```

```
*
* SUBSYSTEM:        EDIT & UPDATE
* PROGRAM NAME:     LOCNCHEK.PRG              06/17/88
* CALLED FROM:      INV_UPDT.PRG
* PROJECT 2456 -    TEXAS FLEXIBLE PAVEMENT DATABASE CONVERSION
* TAMU/TTI
* AUTHOR:           TREVOR X. PEREIRA
* PURPOSE:          CHECKS LOCATION DATA
*


SET TALK OFF
CLEAR
SELE 1
USE \PAVEDB\EDITUPDT\LOCN_NEW INDE \PAVEDB\EDITUPDT\LOCN_NEW
SELE 2
USE \PAVEDB\FILES\LOCATION INDE \PAVEDB\INDEXES\LOCSID
M_ERROR = "LOCATION FILE - INCORRECT DATA ENTERED"
SELE 1

MERRORCNT = 0
SET ALTE ON
? M_ERROR
DO WHILE .NOT. EOF()
   MSID_NO = SID_NO
   * calculates the correct Sid Number
   STORE 0 TO VAR1, VAR2, VAR3, COMPARE
   VAR1 = INT(MSID_NO/1000)
   VAR2 = MOD(INT(MSID_NO/100),10)
   VAR2 = VAR2 * 2
   VAR3 = MOD(INT(MSID_NO/10),10)
   VAR3 = VAR3 * 3
   VAR4 = MOD(VAR1,10)+VAR2+VAR3
   COMPARE = MOD(VAR4,10)

   * compares calculated Sid number with Sid number entered
   IF COMPARE <> MOD(MSID_NO,10)
     MERRORCNT = MERRORCNT + 1
     DISPLAY FIELD SID_NO OFF
     ? "SID NUMBER IS NOT VALID"
     ? " "
   ENDIF
   * verifies that Sid number has been entered
   IF MSID_NO = 0
     MERRORCNT = MERRORCNT + 1
     DISPLAY FIELD SID_NO OFF
     ? "     SID NUMBER IS NOT VALID"
     ? " "
   ENDIF
   SELE 2
```

```
SEEK MSID_NO
IF FOUND()
    MERRORCNT = MERRORCNT + 1
    DISPLAY FIELD SID_NO OFF
    ? "     SID Number is already present in Location file"
    ? " "
ENDIF
SELE 1
IF HWYDIST < 0 .OR. HWYDIST > 27
    DISPLAY FIELDS SID_NO, HWYDIST  OFF
    MERRORCNT = MERRORCNT + 1
ENDIF
IF CNTYNUM < 0 .OR. CNTYNUM > 270
    DISPLAY FIELDS SID_NO, CNTYNUM OFF
    MERRORCNT = MERRORCNT + 1
ENDIF
DO CASE
  CASE HWYPREFX = "FM"
  CASE HWYPREFX = "SH"
  CASE HWYPREFX = "IH"
  CASE HWYPREFX = "US"
  CASE HWYPREFX = "SP"
  CASE HWYPREFX = "LP"
  CASE HWYPREFX = "PR"
  OTHERWISE
    DISPLAY FIELDS SID_NO, HWYPREFX OFF
    MERRORCNT = MERRORCNT + 1
ENDCASE
IF HWYNUM = 0
    DISPLAY FIELDS SID_NO, HWYNUM OFF
    MERRORCNT = MERRORCNT + 1
ENDIF
IF BDISSIGN = "+" .OR. BDISSIGN = "-"
ELSE
    DISPLAY FIELDS SID_NO, BDISSIGN OFF
    MERRORCNT = MERRORCNT + 1
ENDIF
IF EDISSIGN = "+" .OR. EDISSIGN = "-"
ELSE
    DISPLAY FIELDS SID_NO, EDISSIGN OFF
    MERRORCNT = MERRORCNT + 1
ENDIF
IF LANEID = "R" .OR. LANEID = "L"
ELSE
    DISPLAY FIELDS SID_NO, LANEID OFF
    MERRORCNT = MERRORCNT + 1
ENDIF
IF ENDMPNT = 0 .AND. BEGMPNT = 0
    DISPLAY FIELDS SID_NO, BEGMPNT, ENDMPNT OFF
    MERRORCNT = MERRORCNT + 1
ENDIF
IF MPNIMO < 0 .OR. MPNIMO > 12
    DISPLAY FIELDS SID_NO, MPNIMO OFF
    MERRORCNT = MERRORCNT + 1
```

```
   ENDIF
   IF NUMLANES < 1 .OR. NUMLANES > 22
       DISPLAY FIELDS SID_NO, NUMLANES OFF
       MERRORCNT = MERRORCNT + 1
   ENDIF
   IF ACTVFLAG
       IF INACTYR <> 0
          DISPLAY FIELDS SID_NO, INACTYR OFF
          ? "    Inactive Flag and Inactive YEAR are inconsistent"
          ? " "
          ? " "
          MERRORCNT = MERRORCNT + 1
       ENDIF
       IF INACIMO <> 0
          DISPLAY FIELDS SID_NO, INACIMO OFF
          ? "    Inactive Flag and Inactive MONTH are inconsistent"
          ? " "
          ? " "
          MERRORCNT = MERRORCNT + 1
       ENDIF
       IF NEXTSID <> 0
          DISPLAY FIELDS SID_NO, NEXTSID OFF
          ? "    Cannot point to Another SID number when present SID number is
active"
          ? " "
          ? " "
          MERRORCNT = MERRORCNT + 1
       ENDIF
   ENDIF
   IF PREVSID <> 0
       MPREVSID = PREVSID
       MBEGMPST = BEGMPST
       MBDISSIGN = BDISSIGN
       MBMPSTDIS = BMPSTDIS
       MENDMPST = ENDMPST
       MEDISSIGN = EDISSIGN
       MEMPSTDIS = EMPSTDIS
       SELE 2
       SEEK MPREVSID
       IF .NOT. FOUND()
          SELE 1
          DISPLAY FIELD SID_NO, PREVSID
          ?  "   SID Number NOT present in location file"
          ? " "
          ? " "
          MERRORCNT = MERRORCNT + 1
       ELSE
         IF MBEGMPST <> BEGMPST .OR. MBDISSIGN <> BDISSIGN .OR. MBMPSTDIS <>
BMPSTDIS
           SELE 1
           DISPLAY FIELD SID_NO, PREVSID OFF
           ?  "   SID Number and PREVIOUS SID Beginning Mile Post do not match"
           ? " "
           ? " "
```

```
            MERRORCNT = MERRORCNT + 1
          ENDIF
          SELE 2
          IF MENDMPST <> ENDMPST .OR. MEDISSIGN <> EDISSIGN .OR. MEMPSTDIS <>
EMPSTDIS
            SELE 1
            DISPLAY FIELD SID_NO, PREVSID OFF
            ? "   SID Number and PREVIOUS SID Beginning Mile Post do not match"
            ? " "
            ? " "
            MERRORCNT = MERRORCNT + 1
          ENDIF
        ENDIF
        SELE 1
    ENDIF
    IF CONTROL = 0
        DISPLAY FIELDS SID_NO, CONTROL OFF
        MERRORCNT = MERRORCNT + 1
    ENDIF
    IF SECTION = 0
        DISPLAY FIELDS SID_NO, SECTION OFF
        MERRORCNT = MERRORCNT + 1
    ENDIF
    IF FUNCLAS < 1 .OR. FUNCLAS > 7
        DISPLAY FIELDS SID_NO, FUNCLAS OFF
        MERRORCNT = MERRORCNT + 1
    ENDIF
    IF ENDMPST = 0
        DISPLAY FIELDS SID_NO, ENDMPST OFF
        MERRORCNT = MERRORCNT + 1
    ENDIF
    IF EMPSTDIS < 0
        DISPLAY FIELDS SID_NO, EMPSTDIS OFF
        MERRORCNT = MERRORCNT + 1
    ENDIF
    IF BMPSTDIS < 0
        DISPLAY FIELDS SID_NO, BMPSTDIS OFF
        MERRORCNT = MERRORCNT + 1
    ENDIF
    IF MPNTYR < 20 .OR. MPNTYR > 99
        DISPLAY FIELDS SID_NO, HWYNUM OFF
        MERRORCNT = MERRORCNT + 1
    ENDIF
    IF .NOT. ACTVFLAG
        IF INACTYR < 72 .OR. INACTYR > 99
           DISPLAY FIELDS SID_NO, INACTYR OFF
           MERRORCNT = MERRORCNT + 1
        ENDIF
        IF INACTMO < 0 .OR. INACTMO > 12
           DISPLAY FIELDS SID_NO, INACTMO OFF
           MERRORCNT = MERRORCNT + 1
        ENDIF
    ENDIF
    IF MERRORCNT <> 0
```

```
        REPLACE ERRORCHAR WITH .T.
        MERRORCNT = 0
    ENDIF
    SKIP
ENDDO
SELE 1
USE
SELE 2
USE
* reset parameters
CLEAR
?? CHR(12)
SET ALTE OFF
RETURN
```

```
*
* SUBSYSTEM:      EDIT & UPDATE
* PROGRAM NAME:   LAYRCHEK.PRG        06/08/88
* CALLED FROM:    INV_UPDT.PRG
* PROJECT 2456 - TEXAS FLEXIBLE PAVEMENT DATABASE SYSTEM
* TAMU/TTI
* AUTHOR:         TREVOR X. PEREIRA
* PURPOSE:        LIST OF PROCEDURES TO CHECK FOR ERRORS IN THE LAYER FILE
*


*****************
* PROCEDURE LAYER DESCRIPTION
*****************
PROC LAYDESC

* set parameters
SET TALK OFF
SET ECHO OFF
CLEAR

USE \PAVEDB\EDITUPDT\LAYR_NEW INDEX \PAVEDB\EDITUPDT\LAYR_NEW
M_ERROR  = "LAYER ID FILE - INCORRECT LAYER DESCRIPTION OR MATERIAL
CLASSIFICATION"
M_ERROR2 =
"-----------------------------------------------------------------"
SET ALTE ON
? M_ERROR
? M_ERROR2
* check layer description with layer material classification
DO WHILE .NOT. EOF()
   DO CASE
      CASE LAYDESC = 7
          IF LAYMATCL < 41 .OR. LAYMATCL > 45
             DISPLAY FIELDS SID_NO, STRUCNUM, LAYNUM, LAYDESC, LAYMATCL OFF
             REPLACE ERRORCHAR WITH .T.
          ENDIF

      CASE LAYDESC = 6
          IF LAYMATCL < 21 .OR. LAYMATCL > 33
             IF LAYMATCL <> 17
                DISPLAY FIELDS SID_NO, STRUCNUM, LAYNUM, LAYDESC, LAYMATCL OFF
                REPLACE ERRORCHAR WITH .T.
             ENDIF
          ENDIF

      CASE LAYDESC = 5
          IF LAYMATCL < 21 .OR. LAYMATCL > 27
             IF LAYMATCL <> 17
                DISPLAY FIELDS SID_NO, STRUCNUM, LAYNUM, LAYDESC, LAYMATCL OFF
```

```
            REPLACE ERRORCHAR WITH .T.
         ENDIF
      ENDIF

   CASE LAYDESC = 4
      IF LAYMATCL < 1 .OR. LAYMATCL > 4
         DISPLAY FIELDS SID_NO, STRUCNUM, LAYNUM, LAYDESC, LAYMATCL OFF
         REPLACE ERRORCHAR WITH .T.
      ENDIF

   CASE LAYDESC = 3
      IF LAYMATCL < 1 .OR. LAYMATCL > 17
         DISPLAY FIELDS SID_NO, STRUCNUM, LAYNUM, LAYDESC, LAYMATCL OFF
         REPLACE ERRORCHAR WITH .T.
      ENDIF

   CASE LAYDESC = 2
      IF LAYMATCL <> 11
         DISPLAY FIELDS SID_NO, STRUCNUM, LAYNUM, LAYDESC, LAYMATCL OFF
         REPLACE ERRORCHAR WITH .T.
      ENDIF

   CASE LAYDESC = 1
      IF LAYMATCL < 1 .OR. LAYMATCL > 16
         DISPLAY FIELDS SID_NO, STRUCNUM, LAYNUM, LAYDESC, LAYMATCL OFF
         REPLACE ERRORCHAR WITH .T.
      ENDIF

   CASE LAYDESC = 10
      IF LAYMATCL < 5 .OR. LAYMATCL > 7
         DISPLAY FIELDS SID_NO, STRUCNUM, LAYNUM, LAYDESC, LAYMATCL OFF
         REPLACE ERRORCHAR WITH .T.
      ENDIF

   CASE LAYDESC = 12
      IF LAYMATCL < 12 .OR. LAYMATCL > 15
         DISPLAY FIELDS SID_NO, STRUCNUM, LAYNUM, LAYDESC, LAYMATCL OFF
         REPLACE ERRORCHAR WITH .T.
      ENDIF

   CASE LAYDESC = 14
      IF LAYMATCL <> 27
         DISPLAY FIELDS SID_NO, STRUCNUM, LAYNUM, LAYDESC, LAYMATCL OFF
         REPLACE ERRORCHAR WITH .T.
      ENDIF
ENDCASE
SKIP
ENDDO
USE
* reset parameters
? CHR(12)
CLEAR
SET ALTE OFF
RETURN
```

```
*********************
* PROCEDURE VERIFY LAYER NUMBERS
*********************
PROC LAYRSVER
*
* VERIFIES THAT THE LAYER NUMBERS ARE IN CONSECUTIVE
* ORDER IN THE TRANSACTION FILE AND THAT NONE ARE MISSING
*

CLEAR
SELE 1
USE \PAVEDB\FILES\LAYER INDEX \PAVEDB\INDEXES\LAYNDX
SELE 2
USE \PAVEDB\EDITUPDT\LAYR_NEW INDE \PAVEDB\EDITUPDT\LAYR_NEW

M_ERROR  = "LAYER ID FILE - MISSING  OR  INCORRECT  LAYER  NUMBERS"
M_ERROR2 = "-----------------------------------------------------"

SET ALTE ON
? M_ERROR
? M_ERROR2
DO WHILE .NOT. EOF()
   MSID = SID_NO

   SELE 1
   SEEK STR(MSID,4)
   IF FOUND()
     DO WHILE .NOT. EOF() .AND. SID_NO <> MSID
       SKIP
     ENDDO
     SKIP -1
     MLAYNUM = LAYNUM + 1
   ELSE
     MLAYNUM = 1
   ENDIF

   * checks for consecutive sid numbers
   SELE 2
   DO WHILE .NOT. EOF() .AND. B->SID_NO = MSID
       IF LAYNUM <> MLAYNUM
          IF LAYDESC = 13 .OR. LAYDESC = 12
              SKIP
              STORE LAYNUM TO MLAYNUM
              IF MSID <> SID_NO
                  SKIP -1
              ENDIF
          ELSE
              DISPLAY SID_NO, STRUCNUM, LAYNUM OFF
              REPLACE ERRORCHAR WITH .T.
              DO WHILE .NOT. EOF() .AND. MSID = SID_NO
                SKIP
                REPLACE ERRORCHAR WITH .T.
```

```
            ENDDO
            SKIP -1
         ENDIF
      ENDIF
      MLAYNUM = MLAYNUM + 1
      IF .NOT. EOF()
         SKIP
      ENDIF
   ENDDO
ENDDO
USE
* resets parameters
N = 0
DO WHILE N < 6
   ? CHR(13)
   N = N + 1
ENDDO
SET ALTE OFF
clear
RETURN


*********************
* PROCEDURE LAYER OUT OF RANGE CHECK
*********************
PROC LAYRCHK3
*
* TO LIST OUT OF RANGE DATA IN THE LAYER IDENTIFICATION
* TRANSACTION FILE
*

* set parameters
SET TALK OFF
SET ECHO OFF

USE \PAVEDB\EDITUPDT\LAYR_NEW INDEX \PAVEDB\EDITUPDT\LAYR_NEW

M_ERRORCK = 0
M_ERROR  = "LAYER ID FILE - OUT  OF  RANGE  DATA"
M_ERROR2 = "-----------------------------------"

SET ALTE ON
? M_ERROR
? M_ERROR2

* check the data
DO WHILE .NOT. EOF()
    IF STRUCNUM < 1 .OR. STRUCNUM > 9
       DISPLAY FIELDS SID_NO, STRUCNUM, LAYNUM OFF
       M_ERRORCK = M_ERRORCK + 1
    ENDIF
    IF LAYNUM < 1 .OR. LAYNUM > 13
       DISPLAY FIELDS SID_NO, STRUCNUM, LAYNUM OFF
       M_ERRORCK = M_ERRORCK + 1
```

```
    ENDIF
    IF LAYDESC < 1 .OR. LAYDESC > 14
       DISPLAY FIELDS SID_NO, STRUCNUM, LAYNUM, LAYDESC OFF
       M_ERRORCK = M_ERRORCK + 1
    ENDIF
    IF CENTTHK < .1 .OR. CENTTHK > 25
        IF LAYNUM <> 1 .AND. CENTTHK = 0
        ELSE
          DISPLAY FIELDS SID_NO, STRUCNUM, LAYNUM, CENTTHK OFF
          M_ERRORCK = M_ERRORCK + 1
        ENDIF
    ENDIF
    IF LAYMATCL < 1 .OR. LAYMATCL > 45
       DISPLAY FIELDS SID_NO, STRUCNUM, LAYNUM, LAYMATCL OFF
       M_ERRORCK = M_ERRORCK + 1
    ENDIF
    IF JOBCMPYR < 20 .OR. JOBCMPYR > 99
       DISPLAY FIELDS SID_NO, STRUCNUM, LAYNUM, JOBCMPYR OFF
       M_ERRORCK = M_ERRORCK + 1
    ENDIF
    IF JOBCMPMO < 1 .OR. JOBCMPMO > 12
       DISPLAY FIELDS SID_NO, STRUCNUM, LAYNUM, JOBCMPMO OFF
       M_ERRORCK = M_ERRORCK + 1
    ENDIF
    IF WIDENLYR < 28 .OR. WIDENLYR > 99
       IF WIDENLYR <> 0
           DISPLAY FIELDS SID_NO, STRUCNUM, LAYNUM, WIDENLYR OFF
           M_ERRORCK = M_ERRORCK + 1
       ENDIF
    ENDIF
    IF WIDENLMO < 1 .OR. WIDENLMO > 12
       IF WIDENLMO <> 0
            DISPLAY FIELDS SID_NO, STRUCNUM, LAYNUM, WIDENLMO OFF
            M_ERRORCK = M_ERRORCK + 1
       ENDIF
    ENDIF

    IF M_ERRORCK <> 0
       REPLACE ERRORCHAR WITH .T.
       M_ERRORCK = 0
    ENDIF
    SKIP
ENDDO

USE
* reset parameters
N = 0
DO WHILE N < 6
   ? CHR(13)
   N = N + 1
ENDDO
CLEAR
SET ALTE OFF
RETURN
```

```
*******************
* PROCEDURE CHECK SID NUMBER
*******************
PROC SIDCHEK1
*
* LIST OUT SID NUMBERS THAT ARE INCORRECT
*

* set parameters
set talk off
set echo off
clear

USE \PAVEDB\EDITUPDT\LAYR_NEW INDEX \PAVEDB\EDITUPDT\LAYR_NEW
M_ERROR  = "LAYER ID FILE  -  INCORRECT  SID  NUMBERS"
M_ERROR2 = "----------------------------------------"
SET ALTE ON
? M_ERROR
? M_ERROR2

* initialize variables
STORE 0 TO VAR1, VAR2, VAR3, VAR4, COMPARE

* calculate correct Sid Numnber check digit
DO WHILE  .NOT. EOF()
    VAR1 = INT(SID_NO/1000)
    VAR2 = MOD(INT(SID_NO/100),10)
    VAR2 = VAR2 * 2
    VAR3 = MOD(INT(SID_NO/10),10)
    VAR3 = VAR3 * 3
    VAR4 = MOD(VAR1,10)+VAR2+VAR3
    COMPARE = MOD(VAR4,10)

    * compare actual check digit with calculated check digit
    IF COMPARE <> MOD(SID_NO,10)
        DISPLAY SID_NO, STRUCNUM, LAYNUM OFF
        REPLACE ERRORCHAR WITH .T.
    ENDIF
    IF SID_NO = 0
        DISPLAY SID_NO, STRUCNUM, LAYNUM OFF
        REPLACE ERRORCHAR WITH .T.
    ENDIF
    SKIP
ENDDO
USE
* reset parameters
N = 0
DO WHILE N < 6
   ? CHR(13)
   N = N + 1
ENDDO
SET ALTE OFF
```

```
USE
CLEAR
RETURN


*******************
* PROCEDURE LOCATION CHECK
*******************
PROC LOCCHEK1
*
* LIST OUT LAYER RECORDS THAT DO NOT HAVE A LOCATION RECORD
*

* set parameters
set talk off
set echo off
clear

SELE 1
USE \PAVEDB\EDITUPDT\LAYR_NEW INDEX \PAVEDB\EDITUPDT\LAYR_NEW
SELE 2
USE \PAVEDB\FILES\LOCATION INDE \PAVEDB\INDEXES\LOCSID

M_ERROR  = "LAYER ID FILE - ENTERED SID NUMBERS THAT DO NOT HAVE A LOCATION
RECORD"
M_ERROR2 =
"-------------------------------------------------------------------------"

SET ALTE ON
? M_ERROR
? M_ERROR2
SELE 1
DO WHILE  .NOT. EOF()
    MSID_NO = A->SID_NO
    SELE 2
    SEEK MSID_NO
    IF .NOT. FOUND()
        SELE 1
        DISPLAY SID_NO, STRUCNUM, LAYNUM OFF
        REPLACE ERRORCHAR WITH .T.
    ENDIF
    SELE 1
    SKIP
ENDDO
SELE 1
USE
SELE 2
USE

* reset parameters
N = 0
DO WHILE N < 6
   ? CHR(13)
   N = N + 1
```

```
ENDDO
SET ALTE OFF
CLEAR
RETURN
```

```
*
* SUBSYSTEM:      EDIT & UPDATE INVENTORY FILES
* PROGRAM NAME:   LAYTCHEK.PRG       06/08/88
* CALLED FROM:    INV_UPDT.PRG
* PROJECT 2456 - TEXAS FLEXIBLE PAVEMENT DATABASE CONVERSION
* TAMU/TTI
* AUTHOR:         TREVOR X. PEREIRA
* PURPOSE:        TO LIST ERRORS INT THE LAYER THICKNESS ACROSS THE ROAD FILE
*

*********************
* PROCEDURE CHECK LAYER THICKNESS FOR RECORDS NOT IN LAYER FILE
*********************
PROC LAYTCHK1
* set parameters
Set talk oFF
set echo oFF
CLEAR

* print out page heading
SET ALTE ON
MTITLE  = "LAYER THICKNESS FILE - LIST OF NEW RECORDS ENTERED NOT PRESENT IN
LAYER ID FILE"
MTITLE2 =
"-------------------------------------------------------------------------------"
? MTITLE
? MTITLE2

* assign files to different work areas
SELECT 1
   USE \PAVEDB\FILES\LAYER INDEX \PAVEDB\INDEXES\LAYNDX
SELECT 2
   USE \PAVEDB\EDITUPDT\LAYT_NEW INDEX \PAVEDB\EDITUPDT\LAYT_NEW

DO WHILE .NOT. EOF()
   MLAYSID = SID_NO
   MLAYSTRC = STRUCNUM
   MLAYNUM = LAYNUM

   * find the laythick file record in layer file
   SELECT 1
   seek str(MLAYSID,4)+str(MLAYSTRC,2)+str(MLAYNUM,2)
   IF .NOT. FOUND()
      SELECT 2
      DISPLAY FIELDS SID_NO, STRUCNUM, LAYNUM OFF
      REPLACE ERRORCHAR WITH .T.
   ENDIF
   SELE 2
   SKIP
```

```
ENDDO

* reset parameters
N = 0
DO WHILE N < 6
   ? CHR(13)
   N = N + 1
ENDDO
CLEAR
SET ALTE OFF
SELE 1
USE
SELE 2
USE
RETURN


**********************
* PROCEDURE TO CHECK OUT OF RANGE DATA
**********************
PROC LAYTCHK3
*
* TO LIST ALL OUT OF RANGE DATA IN THE LAYER THICKNESS
* ACROSS THE ROAD FILE
*

* set parameters
SET TALK OFF
SET ECHO OFF

USE \PAVEDB\EDITUPDT\LAYT_NEW INDEX \PAVEDB\EDITUPDT\LAYT_NEW
GOTO TOP
M_ERROR = 0

* print out the page heading
M_TITLE  = "LAYER THICKNESS FILES - OUT OF RANGE CHECK"
M_TITLE2 = "----------------------------------------"
SET ALTE ON
? M_TITLE
? M_TITLE2

* check the ranges
DO WHILE .NOT. EOF()
    IF STRUCNUM < 1 .OR. STRUCNUM > 9
       DISPLAY FIELDS SID_NO, STRUCNUM, LAYNUM, STRUCNUM OFF
       M_ERROR = M_ERROR + 1
    ENDIF
    IF LAYNUM < 1 .OR. LAYNUM > 13
       DISPLAY FIELDS SID_NO, STRUCNUM, LAYNUM, LAYNUM OFF
       M_ERROR = M_ERROR + 1
    ENDIF
    IF FC3THK < 0 .OR. FC3THK > 25
       DISPLAY FIELDS SID_NO, STRUCNUM, LAYNUM, FC3THK OFF
       M_ERROR = M_ERROR + 1
```

```
    ENDIF
    IF FC2THK < 0 .OR. FC2THK > 25
       DISPLAY SID_NO, STRUCNUM, LAYNUM, FC2THK  OFF
       M_ERROR = M_ERROR + 1
    ENDIF
    IF FC1THK < .1 .OR. FC1THK > 25
       DISPLAY SID_NO, STRUCNUM, LAYNUM, FC1THK  OFF
       M_ERROR = M_ERROR + 1
    ENDIF
    IF CENTITHK < .1 .OR. CENTITHK > 25
       DISPLAY SID_NO, STRUCNUM, LAYNUM, CENTITHK  OFF
       M_ERROR = M_ERROR + 1
    ENDIF
    IF FC1DIS < .01 .OR. FC1DIS > 99
       DISPLAY SID_NO, STRUCNUM, LAYNUM, FC1DIS  OFF
       M_ERROR = M_ERROR + 1
    ENDIF
    IF FC2DIS < FC1DIS
       IF FC2DIS <> 0
          DISPLAY SID_NO, STRUCNUM, LAYNUM, FC2DIS  OFF
          M_ERROR = M_ERROR + 1
       ENDIF
    ENDIF
    IF FC3DIS < FC2DIS
       IF FC3DIS <> 0
          DISPLAY SID_NO, STRUCNUM, LAYNUM, FC3DIS  OFF
          M_ERROR = M_ERROR + 1
       ENDIF
    ENDIF
    IF M_ERROR <> 0
       REPL ERRORCHAR WITH .T.
       M_ERROR = 0
    ENDIF
    SKIP
ENDDO

* reset parameters
N = 0
DO WHILE N < 6
   ? CHR(13)
   N = N + 1
ENDDO
CLEAR
SET ALTE OFF
USE
RETURN



**********************
* PROCEDURE TO CHECK THE SID NUMBER
**********************
PROC SIDCHEK2
*
```

```
* TO CHECK FOR INVALID SID NUMBERS
*

* set parameters
set talk off
set echo off
CLEAR

USE \PAVEDB\EDITUPDT\LAYT_NEW INDEX \PAVEDB\EDITUPDT\LAYT_NEW

* set up page heading
M_TITLE1 = "LAYER THICKNESS FILE - SID NUMBER CHECK"
M_TITLE2 = "----------------------------------------"
SET ALTE ON
? M_TITLE1
? M_TITLE2
STORE 0 TO VAR1, VAR2, VAR3, VAR4, COMPARE

* calculate check digit
DO WHILE  .NOT. EOF()
    VAR1 = INT(SID_NO/1000)
    VAR2 = MOD(INT(SID_NO/100),10)
    VAR2 = VAR2 * 2
    VAR3 = MOD(INT(SID_NO/10),10)
    VAR3 = VAR3 * 3
    VAR4 = MOD(VAR1,10)+VAR2+VAR3
    COMPARE = MOD(VAR4,10)

    * compare calculated check digit with actual check digit
    IF COMPARE <> MOD(SID_NO,10)
        DISPLAY SID_NO OFF
        REPL ERRORCHAR WITH .T.
        MSID = SID_NO
        DO WHILE SID_NO = MSID .OR. (.NOT. EOF() )
          SKIP
        ENDDO
        SKIP -1
    ENDIF
    IF SID_NO = 0
        DISPLAY SID_NO, STRUCNUM, LAYNUM OFF
        REPL ERRORCHAR WITH .T.
    ENDIF
    SKIP
ENDDO

* reset parameters
? CHR(12)
CLEAR
SET ALTE OFF
USE
RETURN


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

```
* PROCEDURE LAYER THICKNESS CHECK 2
**********************
PROC LAYTCHK2
*
* TO LIST LAYER IDENTIFICATION FILE RECORDS THAT ARE NOT
* PRESENT IN LAYER THICKNESS ACROSS THE ROAD FILE
*

* set parameters
SET TALK OFF
SET ECHO OFF
CLEAR

M_TITLE  = "LAYER THICKNESS FILE - LIST OF ALL RECORDS IN LAYER ID FILE THAT ARE
NOT"
M_TITLE2 = "          PRESENT IN THE LAYER THICKNESS FILE"
M_TITLE3 =
"_____________________________________________________________________________"
SET ALTE ON
? M_TITLE
? M_TITLE2
? M_TITLE3

* assign files to different work areas
SELECT 1
   USE \PAVEDB\FILES\LAYER INDEX \PAVEDB\INDEXES\LAYNDX
SELECT 2
   USE \PAVEDB\FILES\LAYTHICK INDEX \PAVEDB\INDEXES\LAYTNDX

SELECT 1
DO WHILE .NOT. EOF()
   MSID = SID_NO
   MSTR = STRUCNUM
   MLAY = LAYNUM

   * if layer number is 1, skip to the next record in layer file
   IF LAYNUM = 1
      SKIP
      LOOP
   ENDIF

   * find the layer record in laythick file
   SELECT 2
   SEEK STR(MSID,4)+STR(MSTR,2)+STR(MLAY,2)
   IF .NOT. FOUND()
      SELECT 1
      DISPLAY FIELDS SID_NO, STRUCNUM, LAYNUM OFF
      REPLACE ERRORCHAR WITH .T.
   ENDIF
   SELECT 1
   SKIP
ENDDO

* reset parameters
```

```
N = 0
DO WHILE N < 6
   ? CHR(13)
   N = N + 1
ENDDO
CLEAR
SET ALTE OFF
SELE 1
USE
SELE 2
USE
RETURN
```

```
*
* SUBSYSTEM:      EDIT & UPDATE INVENTORY FILES
* PROGRAM NAME:   GEOSCHEK.PRG       06/07/88
* CALLED FROM:    INV_UPDT.PRG
* PROJECT 2456 - TEXAS FLEXIBLE PAVEMENT DATABASE CONVERSION
* TAMU/TTI
* AUTHOR:         TREVOR X. PEREIRA
* PURPOSE:        PROCEDURES TO LIST NEWLY ENTERED RECORDS THAT ARE
*                 IN ERROR
*

******************
* PROCEDURE GEOSCHK1
******************
PROC GEOSCHK1

* set parameters
set talk oFF
set echo oFF
CLEAR

M_TITLE  = "GEOMEITRIC & SHOULDER FILE - LIST OF NEWLY ENTERED RECORDS NOT IN
LAYER ID FILE"
M_TITLE2 =
"--------------------------------------------------------------------------------"
SET ALTE ON
? M_TITLE
? M_TITLE2

* assign database files to different work areas
SELECT 1
   USE \PAVEDB\FILES\LAYER INDEX \PAVEDB\INDEXES\LAYNDX

SELECT 2
   USE \PAVEDB\EDITUPDT\GEOS_NEW INDEX \PAVEDB\EDITUPDT\GEOS_NEW

SELE 2
DO WHILE .NOT. EOF()
   * store to memory variables
   MSID = SID_NO
   MSTRUC = STRUCNUM

   SELECT 1
   * find the record in layer that equals the sid number and structure
   * number
   SEEK STR(MSID,4)+STR(MSTRUC,2)

   * if record is not found, print the sid number
   IF .NOT. FOUND()
      SELE 2
```

```
      DISPLAY FIELD SID_NO, STRUCNUM OFF
      REPLACE ERRORCHAR WITH .T.
   ENDIF
   SELE 2
   SKIP
ENDDO

* reset parameters
N = 0
DO WHILE N < 6
   ? CHR(13)
   N = N + 1
ENDDO
CLEAR
SET ALTE OFF
SELE 1
USE
SELE 2
USE
return


*********************
* PROCEDURE GEOSCHK3
*********************
PROC GEOSCHK3

*
* TO LIST ALL OUT OF RANGE DATA FOR THE GEOMETRIC AND
* INFORMATION FILE
*

* set parameters
SET TALK OFF
SET ECHO OFF

USE \PAVEDB\EDITUPDT\GEOS_NEW INDEX \PAVEDB\EDITUPDT\GEOS_NEW
M_ERROR = 0

* set up page heading
M_TITLE  = "GEOMETRIC & SHOULDER FILE - OUT OF RANGE CHECK"
M_TITLE2 = "-----------------------------------------------"
SET ALTE ON
? M_TITLE
? M_TITLE2

* check data range
DO WHILE .NOT. EOF()
    IF STRUCNUM < 1 .OR. STRUCNUM > 9
       DISPLAY FIELDS SID_NO, STRUCNUM OFF
       M_ERROR = M_ERROR + 1
    ENDIF
    IF PAVETYP < 1 .OR. PAVETYP > 27
       DISPLAY FIELDS SID_NO, STRUCNUM, PAVETYP OFF
```

```
      M_ERROR = M_ERROR + 1
   ENDIF
   IF LANEWID < 8 .OR. LANEWID > 15
      IF LANEWID <> 0
         DISPLAY FIELDS SID_NO, STRUCNUM, LANEWID OFF
         M_ERROR = M_ERROR + 1
      ENDIF
   ENDIF
   IF OUTSHOWD < 0 .OR. OUTSHOWD > 12
      DISPLAY FIELDS SID_NO, STRUCNUM, OUTSHOWD OFF
      M_ERROR = M_ERROR + 1
   ENDIF
   IF SHOSFTYP < 1 .OR. SHOSFTYP > 6
      DISPLAY FIELDS SID_NO, STRUCNUM, SHOSFTYP OFF
      M_ERROR = M_ERROR + 1
   ENDIF
   IF SHOBSTYP < 21 .OR. SHOBSTYP > 45
      IF SHOBSTYP <> 0
         DISPLAY FIELDS SID_NO, STRUCNUM, SHOBSTYP OFF
         M_ERROR = M_ERROR + 1
      ENDIF
   ENDIF
   IF WIDENFLG < 0 .OR. WIDENFLG > 2
      DISPLAY FIELDS SID_NO, STRUCNUM, WIDENFLG OFF
      M_ERROR = M_ERROR + 1
   ENDIF
   IF SHOSFTHK < 0 .OR. SHOSFTHK > 10.1
      DISPLAY FIELDS SID_NO, STRUCNUM, SHOSFTHK OFF
      M_ERROR = M_ERROR + 1
   ENDIF
   IF M_ERROR <> 0
      REPLACE ERRORCHAR WITH .T.
      M_ERROR = 0
   ENDIF
   SKIP

ENDDO

* reset parameters
N = 0
DO WHILE N < 6
   ? CHR(13)
   N = N + 1
ENDDO
CLEAR
SET ALTE OFF
USE
RETURN


**********
* PROCEDURE GEOSID
**********
PROC SIDCHEK3
```

```
*
* TO LIST OUT ALL INCORRECT SID NUMBERS FOR GEOMETRIC & SHOULDER
*

* SET PARAMETERS
set talk off
set echo off

USE \PAVEDB\EDITUPDT\GEOS_NEW INDEX \PAVEDB\EDITUPDT\GEOS_NEW

* SET PAGE HEADING
M_TITLE  = "GEOMETRIC & SHOULDER FILE - LIST OF INCORRECT SID NUMBERS"
M_TITLE2 = "-------------------------------------------------------"
SET ALTE ON
? M_TITLE
? M_TITLE2

STORE 0 TO VAR1, VAR2, VAR3, VAR4, COMPARE
DO WHILE  .NOT. EOF()
    * calculate the correct check digit for the Sid number
    VAR1 = INT(SID_NO/1000)
    VAR2 = MOD(INT(SID_NO/100),10)
    VAR2 = VAR2 * 2
    VAR3 = MOD(INT(SID_NO/10),10)
    VAR3 = VAR3 * 3
    VAR4 = MOD(VAR1,10)+VAR2+VAR3
    COMPARE = MOD(VAR4,10)

    * compare the entered check digit with the calculated check digit
    IF COMPARE <> MOD(SID_NO,10)
        DISPLAY SID_NO OFF
        REPLACE ERRORCHAR WITH .T.
        MSID = SID_NO
        DO WHILE SID_NO = MSID .OR. (.NOT. EOF() )
          SKIP
        ENDDO
        SKIP -1
    ENDIF
    IF SID_NO = 0
        DISPLAY SID_NO, STRUCNUM OFF
        REPLACE ERRORCHAR WITH .T.
    ENDIF
    SKIP
ENDDO

* reset parameters
? CHR(12)
CLEAR
SET ALTE OFF
USE
RETURN
```

```
*
* SUBSYSTEM:      EDIT & UPDATE INVENTORY FILES
* PROGRAM NAME:   SURFCHEK.PRG      06/08/88
* CALLED FROM:    INV_UPDT.PRG
* PROJECT 2456 - TEXAS FLEXIBLE PAVEMENT DATABASE CONVERSION
* TAMU/TTI
* AUTHOR:         TREVOR X. PEREIRA
* PURPOSE:        LISTS THE SURFACE FILE RECORDS THAT HAVE ERRORS
*

*********************
* PROCEDURE TO CHECK SURFACE RECORDS IN LAYER FILE
*********************
PROC SURFCHK1

* LISTS THE SURFACE FILE RECORDS THAT ARE NOT PRESENT IN
* THE MASTER LAYER IDENTIFICATION FILE
* set parameters
set talk oFF
set echo oFF
CLEAR

M_TITLE  = "SURFACE FILE - LIST OF NEWLY ENTERED RECORDS NOT PRESENT IN LAYER ID
FILE"
M_TITLE2 =
"-------------------------------------------------------------------------------"
SET ALTE ON
? M_TITLE
? M_TITLE2

* assign files to different work areas
SELECT 1
   USE \PAVEDB\FILES\LAYER INDEX \PAVEDB\INDEXES\LAYNDX

SELECT 2
   USE \PAVEDB\EDITUPDT\SURF_NEW INDEX \PAVEDB\EDITUPDT\SURF_NEW

DO WHILE .NOT. EOF()
   * store surface record in memory
   MSURFSID = SID_NO
   MSURFSTR = STRUCNUM
   MSURFLAY = LAYNUM

   * find the layer ID record that matches surface record
   SELECT 1
   SEEK STR(MSURFSID,4)+STR(MSURFSTR,2)+STR(MSURFLAY,2)
   IF .NOT. FOUND()
      SELE 2
      DISPLAY FIELDS SID_NO, STRUCNUM, LAYNUM OFF
```

```
      REPLACE ERRORCHAR WITH .T.
   ELSE
      SELE 2
   ENDIF
   SKIP
ENDDO

* reset parameters
N = 0
DO WHILE N < 6
   ? CHR(13)
   N = N + 1
ENDDO
CLEAR
SET ALTE OFF
SELE 1
USE
SELE 2
USE
RETURN


**********************
* PROCEDURE TO CHECK OUT OF RANGE DATA
**********************
PROC SURFCHK3
*
* TO LIST ALL OUT OF RANGE DATA IN THE SURFACE FILE
*

* set parameters
SET TALK OFF
SET ECHO OFF

* print out page headings
M_TITLE  = "SURFACE FILE - OUT OF RANGE RECORDS"
M_TITLE2 = "-----------------------------------"
M_ERROR = 0
SET ALTE ON
? M_TITLE
? M_TITLE2

USE \PAVEDB\EDITUPDT\SURF_NEW INDEX \PAVEDB\EDITUPDT\SURF_NEW

* check data with ranges
DO WHILE .NOT. EOF()
    IF STRUCNUM < 1 .OR. STRUCNUM > 9
       DISPLAY FIELDS SID_NO, STRUCNUM, LAYNUM, STRUCNUM OFF
       M_ERROR = M_ERROR + 1
    ENDIF
    IF LAYNUM < 1 .OR. LAYNUM > 13
       DISPLAY FIELDS SID_NO, STRUCNUM, LAYNUM, LAYNUM OFF
       M_ERROR = M_ERROR + 1
    ENDIF
```

```
    IF AGAPPLRT < 30 .OR. AGAPPLRT > 200
       IF AGAPPLRT <> 0
           DISPLAY SID_NO, STRUCNUM, LAYNUM, AGAPPLRT OFF
           M_ERROR = M_ERROR + 1
       ENDIF
    ENDIF
    IF ADMXPER < 0 .OR. ADMXPER > 10.2
           DISPLAY SID_NO, STRUCNUM, LAYNUM, ADMXPER OFF
           M_ERROR = M_ERROR + 1
    ENDIF
    IF ASAPPLRT < .1 .OR. ASAPPLRT > .6
       IF ASAPPLRT <> 0
           DISPLAY SID_NO, STRUCNUM, LAYNUM, ASAPPLRT OFF
           M_ERROR = M_ERROR + 1
       ENDIF
    ENDIF
    IF M_ERROR <> 0
       REPLACE ERRORCHAR WITH .T.
       M_ERROR = 0
    ENDIF
    SKIP

ENDDO

* reset parameters
N = 0
DO WHILE N < 6
   ? CHR(13)
   N = N + 1
ENDDO
CLEAR
SET ALTE OFF
USE
RETURN


**********************
* PROCEDURE TO CHECK SID NUMBER
**********************
PROC SIDCHEK4
*
* TO LIST ALL INCORRECT SID NUMBERS FOR SURFACE FILE
*

* set parameters
set talk off
set echo off
CLEAR

USE \PAVEDB\EDITUPDT\SURF_NEW INDEX \PAVEDB\EDITUPDT\SURF_NEW

* print out page heading
M_TITLE  = "SURFACE FILE - INCORRECT SID NUMBER"
M_TITLE2 = "--------------------------------"
```

```
SET ALTE ON
? M_TITLE
? M_TITLE2

* initialize variables
STORE 0 TO VAR1, VAR2, VAR3, VAR4, COMPARE

* calculate check digit of Sid number
DO WHILE  .NOT. EOF()
    VAR1 = INT(SID_NO/1000)
    VAR2 = MOD(INT(SID_NO/100),10)
    VAR2 = VAR2 * 2
    VAR3 = MOD(INT(SID_NO/10),10)
    VAR3 = VAR3 * 3
    VAR4 = MOD(VAR1,10)+VAR2+VAR3
    COMPARE = MOD(VAR4,10)

    * compare calculated check digit with actual check digit
    IF COMPARE <> MOD(SID_NO,10)
        DISPLAY SID_NO, STRUCNUM, LAYNUM OFF
        REPLACE ERRORCHAR WITH .T.
    ENDIF
    IF SID_NO = 0
        DISPLAY SID_NO, STRUCNUM, LAYNUM OFF
        REPLACE ERRORCHAR WITH .T.
    ENDIF
    SKIP
ENDDO


* reset parameters
? CHR(12)
CLEAR
SET ALTE OFF
USE
RETURN
```

```
*
* SUBSYSTEM:      EDIT & UPDATE INVENTORY FILES
* PROGRAM NAME:   SUBGCHEK.PRG       06/08/88
* CALLED FROM:    INV_UPDT.PRG
* PROJECT 2456 - TEXAS FLEXIBLE PAVEMENT DATABASE CONVERSION
* TAMU/TTI
* AUTHOR:         TREVOR X. PEREIRA
* PURPOSE:        PROCEDURES TO LIST THE SUBGRADE FILE RECORDS THAT HAVE ERRORS
*


**********************
* PROCEDURE SUBGCHK1
**********************
PROC SUBGCHK1
*
* TO LIST THE SUBGRADE FILE RECORDS THAT ARE NOT PRESENT
* IN THE MASTER LAYER IDENTIFICATION FILE
*
* set parameters
set talk oFF
set echo oFF
CLEAR

M_TITLE  = " SUBGRADE FILE - LIST OF NEWLY ENTERED RECORDS NOT PRESENT IN LAYER
ID FILE"
M_TITLE2 =
"-------------------------------------------------------------------------------"
SET ALTE ON
? M_TITLE
? M_TITLE2

* assign files to different work areas
SELECT 1
   USE \PAVEDB\FILES\LAYER INDEX \PAVEDB\INDEXES\LAYNDX

SELECT 2
   USE \PAVEDB\EDITUPDT\SUBG_NEW INDEX \PAVEDB\EDITUPDT\SUBG_NEW

DO WHILE .NOT. EOF()
   MSUBGSID = SID_NO
   MSUBGLAY = " 1"
   MSUBGSTR = " 1"

   * find the subgrade sid number in layer identification file
   SELECT 1
   seek str(MSUBGSID,4)+MSUBGLAY+MSUBGSTR
   IF .NOT. FOUND()
      SELECT 2
      DISPLAY FIELD SID_NO, STRUCNUM, LAYNUM OFF
```

```
      REPLACE ERRORCHAR WITH .T.
   ENDIF
   SELECT 2
   SKIP
ENDDO

* reset parameters
N = 0
DO WHILE N < 6
   ? CHR(13)
   N = N + 1
ENDDO
SET ALTE OFF
clear
SELE 1
USE
SELE 2
USE
RETURN


*********************
* PROCEDURE TO CHECK OUT OF RANGE DATA
*********************
PROC SUBGCHK3
*
* TO LIST OUT ALL DATA THAT IS OUT OF RANGE IN THE SUBGRADE FILE
*

* set parameters
SET TALK OFF
SET ECHO OFF

* print out page heading
M_TITLE  = "SUBGRADE FILE - OUT OF RANGE RECORDS"
M_TITLE2 = "-----------------------------------"
M_ERROR = 0
SET ALTE ON
? M_TITLE
? M_TITLE2

USE \PAVEDB\EDITUPDT\SUBG_NEW INDEX \PAVEDB\EDITUPDT\SUBG_NEW

* do a range check for all fields
DO WHILE .NOT. EOF()
    IF PLASTIX < 0 .OR. PLASTIX > 70
       IF PLASTIX <> 0
           DISPLAY FIELDS SID_NO, STRUCNUM, LAYNUM, PLASTIX OFF
           M_ERROR = M_ERROR + 1
       ENDIF
    ENDIF
    IF LIQLIM < 10 .OR. LIQLIM > 80
       IF LIQLIM <> 0
           DISPLAY FIELDS SID_NO, STRUCNUM, LAYNUM, LIQLIM OFF
```

```
            M_ERROR = M_ERROR + 1
         ENDIF
      ENDIF
      IF TXTRIAXL < 1 .OR. TXTRIAXL > 6
         IF TXTRIAXL <> 0
              DISPLAY FIELDS SID_NO, STRUCNUM, LAYNUM, TXTRIAXL OFF
              M_ERROR = M_ERROR + 1
         ENDIF
      ENDIF
      IF PERMIX < 0 .OR. PERMIX > 10
         DISPLAY SID_NO, STRUCNUM, LAYNUM, PERMIX OFF
         M_ERROR = M_ERROR + 1
      ENDIF

      IF M_ERROR <> 0
         REPLACE ERRORCHAR WITH .T.
         M_ERROR = 0
      ENDIF
      SKIP

ENDDO

* reset parameters
N = 0
DO WHILE N < 6
   ? CHR(13)
   N = N + 1
ENDDO
SET ALTE OFF
clear
SELE 1
USE
SELE 2
USE
RETURN


*******************
* PROCEDURE TO CHECK SID NUMBERS
*******************
PROC SIDCHEK5
*
* LIST OF INCORRECT SID NUMBERS FOR THE SUBGRADE FILE
*

* set parameters
set talk off
set echo off

M_TITLE  = "SUBGRADE FILE - INCORRECT SID NUMBERS"
M_TITLE2 = "-----------------------------------"
SET ALTE ON
? M_TITLE
? M_TITLE2
```

```
USE \PAVEDB\EDITUPDT\SUBG_NEW INDEX \PAVEDB\EDITUPDT\SUBG_NEW

* initialize variables
STORE 0 TO VAR1, VAR2, VAR3, VAR4, COMPARE

* calculate correct Sid number check digit
DO WHILE  .NOT. EOF()
    VAR1 = INT(SID_NO/1000)
    VAR2 = MOD(INT(SID_NO/100),10)
    VAR2 = VAR2 * 2
    VAR3 = MOD(INT(SID_NO/10),10)
    VAR3 = VAR3 * 3
    VAR4 = MOD(VAR1,10)+VAR2+VAR3
    COMPARE = MOD(VAR4,10)

    * compare calculated check digit with actual check digit
    IF COMPARE <> MOD(SID_NO,10)
        DISPLAY SID_NO, STRUCNUM, LAYNUM OFF
        REPLACE ERRORCHAR WITH .T.
    ENDIF
    IF SID_NO = 0
        DISPLAY SID_NO, STRUCNUM, LAYNUM OFF
        REPLACE ERRORCHAR WITH .T.
    ENDIF
    SKIP
ENDDO

* reset parameters
? CHR(12)
SET ALTE OFF
clear
USE
RETURN
```

TEXAS FLEXIBLE PAVEMENT DATABASE

VOLUME II. PROGRAMMER'S MANUAL

By

Rebecca Yette, Trevor Pereira and Victor Wong

Research Report
456-1F Volume II cont.

on

Research Study Number 2-8-86-456
Texas Flexible Pavement Database

Sponsored By
Texas State Department of Highways & Public Transportation

In Cooperation with
Federal Highway Administration

August 1988

Texas Transportation Institute
Texas A&M University System
College Station, Texas

METRIC (SI*) CONVERSION FACTORS

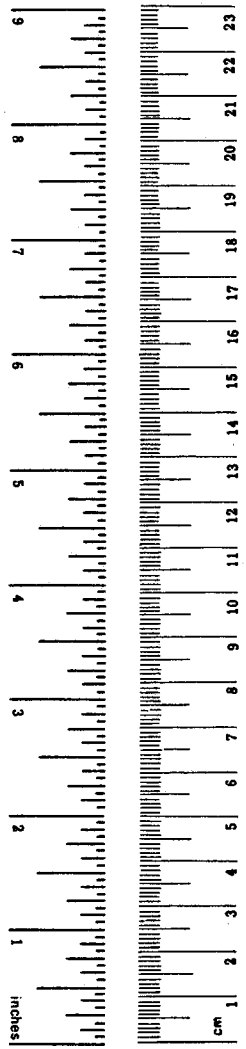
APPROXIMATE CONVERSIONS TO SI UNITS

Symbol	When You Know	Multiply By	To Find	Symbol
LENGTH				
in	inches	2.54	centimetres	cm
ft	feet	0.3048	metres	m
yd	yards	0.914	metres	m
mi	miles	1.61	kilometres	km
AREA				
in^2	square inches	645.2	centimetres squared	cm^2
ft^2	square feet	0.0929	metres squared	m^2
yd^2	square yards	0.836	metres squared	m^2
mi^2	square miles	2.59	kilometres squared	km^2
ac	acres	0.395	hectares	ha
MASS (weight)				
oz	ounces	28.35	grams	g
lb	pounds	0.454	kilograms	kg
T	short tons (2000 lb)	0.907	megagrams	Mg
VOLUME				
fl oz	fluid ounces	29.57	millilitres	mL
gal	gallons	3.785	litres	L
ft^3	cubic feet	0.0328	metres cubed	m^3
yd^3	cubic yards	0.0765	metres cubed	m^3

NOTE: Volumes greater than 1000 L shall be shown in m^3.

TEMPERATURE (exact)

°F	Fahrenheit temperature	5/9 (after subtracting 32)	Celsius temperature	°C

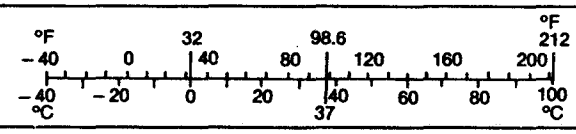


APPROXIMATE CONVERSIONS TO SI UNITS

Symbol	When You Know	Multiply By	To Find	Symbol
LENGTH				
mm	millimetres	0.039	inches	in
m	metres	3.28	feet	ft
m	metres	1.09	yards	yd
km	kilometres	0.621	miles	mi
AREA				
mm^2	millimetres squared	0.0016	square inches	in^2
m^2	metres squared	10.764	square feet	ft^2
km^2	kilometres squared	0.39	square miles	mi^2
ha	hectores (10 000 m^2)	2.53	acres	ac
MASS (weight)				
g	grams	0.0353	ounces	oz
kg	kilograms	2.205	pounds	lb
Mg	megagrams (1 000 kg)	1.103	short tons	T
VOLUME				
mL	millilitres	0.034	fluid ounces	fl oz
L	litres	0.264	gallons	gal
m^3	metres cubed	35.315	cubic feet	ft^3
m^3	metres cubed	1.308	cubic yards	yd^3

TEMPERATURE (exact)

°C	Celsius temperature	9/5 (then add 32)	Fahrenheit temperature	°F



These factors conform to the requirement of FHWA Order 5190.1A.

* SI is the symbol for the International System of Measurements

CONTINUATION OF REPORT 456-1F Vol. II

Narrative on Changing Inventory Data

The Edit & Update program (EDITUPDT.PRG) calls CHNGLAYR.PRG to change the Inventory data. The inventory files include Location, Layer Identification, Layer Thickness Across the Road, Geometric and Shoulder, Surface and Subgrade. The user must make a backup of the master files first. In order to do this, CHNGLAYR.PRG calls the backup program CHNGBKUP.PRG.

The backup program backs up the master files to floppy disk as well as to the hard disk. It also creates 6 temporary dBASE files for the user to make changes to (LOCNCHNG.DBF, LAYRCHNG.DBF, LAYTCHNG.DBF, GEOSCHNG.DBF, SURFCHNG.DBF and SUBGCHNG.DBF).

After the changes have been completed, the Edit/Check programs must be run to flag any errors. The Edit/Check programs include CHEKLOCN.PRG, CHEKLAYR.PRG, CHEKLAYT.PRG, CHEKGEOS.PRG, CHEKSURF.PRG and CHEKSUBG.PRG. If there are any errors present in the files, an error listing is printed out and the user can edit the data again to remove the errors. If no errors are present, the temporary files become the master files with the new changes in them. The original master files are deleted.

The programs and the temporary dBASE files for this section (CHANGE Inventory Data) are in the subdirectory \PAVEDB\EDITUPDT.

The inventory change process is illustrated in figures 16 through 18. Figure 16 depicts the change process on a global level, Figure 17 illustrates the high level program flow logic, and figure 18 charts the programs, procedures, and input and output files used int he inventory change process.

Inventory Data - Change Process

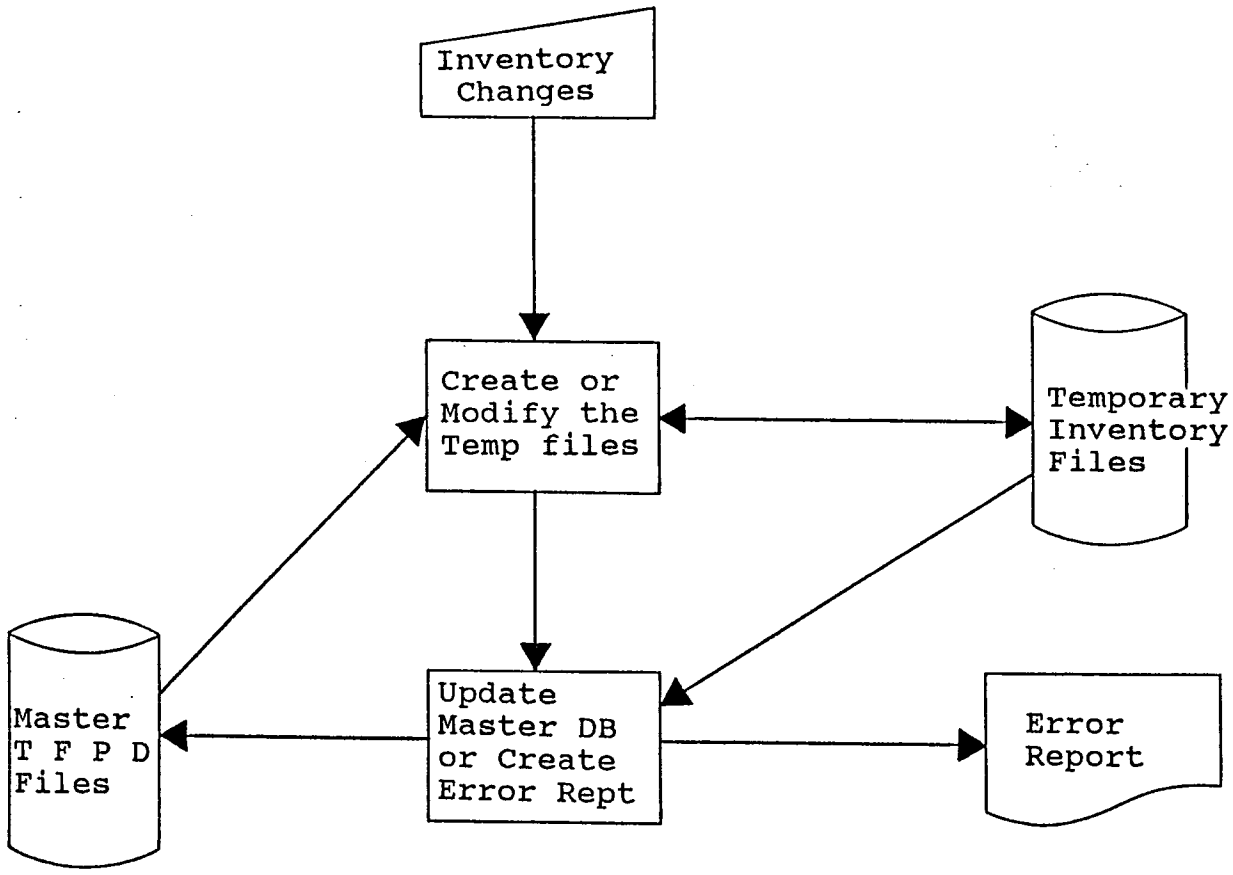


FIGURE 16

Inventory Data - Change Process
Program Flow Chart

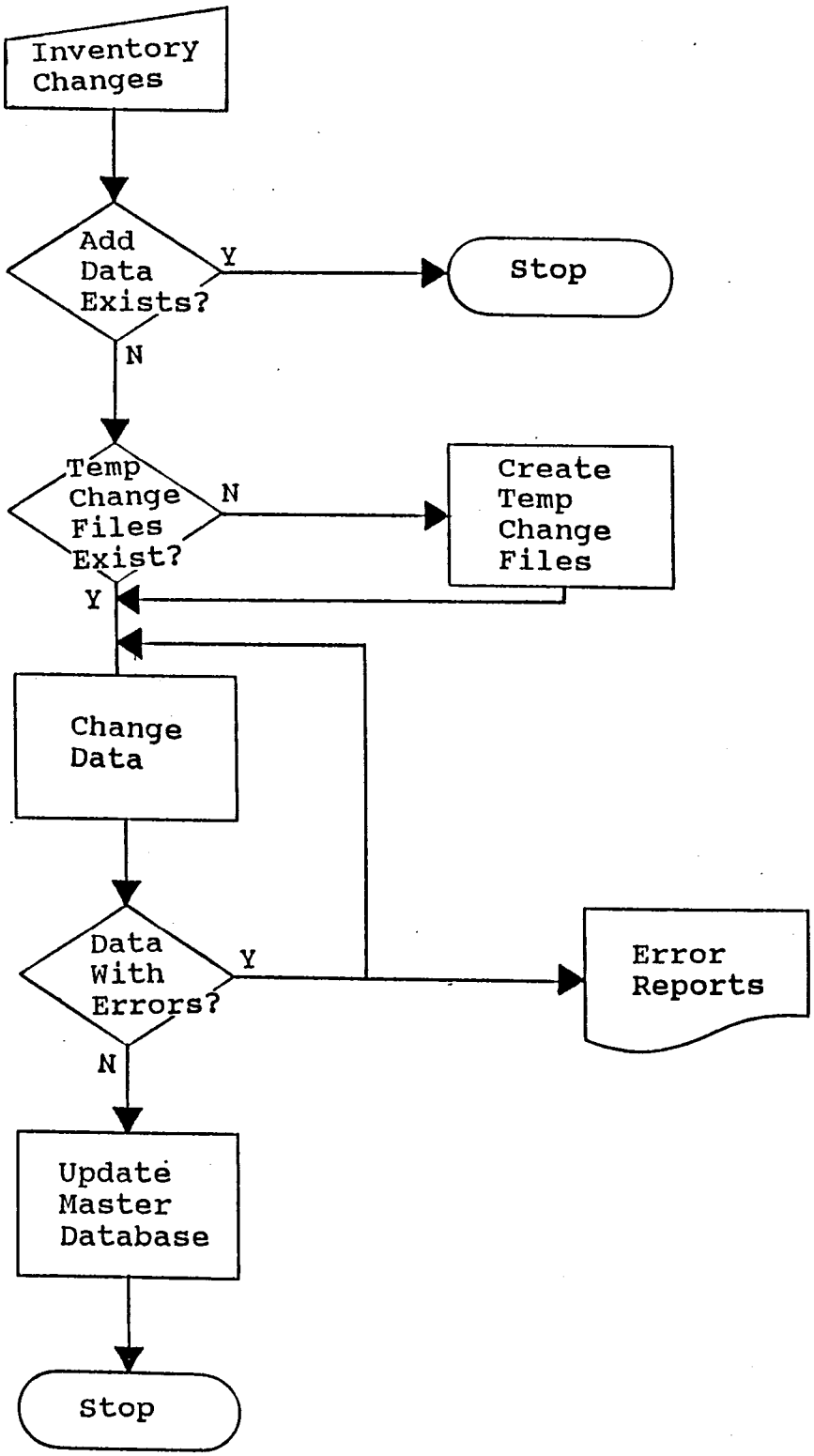


FIGURE 17

INVENTORY DATA - CHANGE PROCESS
PROGRAM FLOW DIAGRAM (Continued)

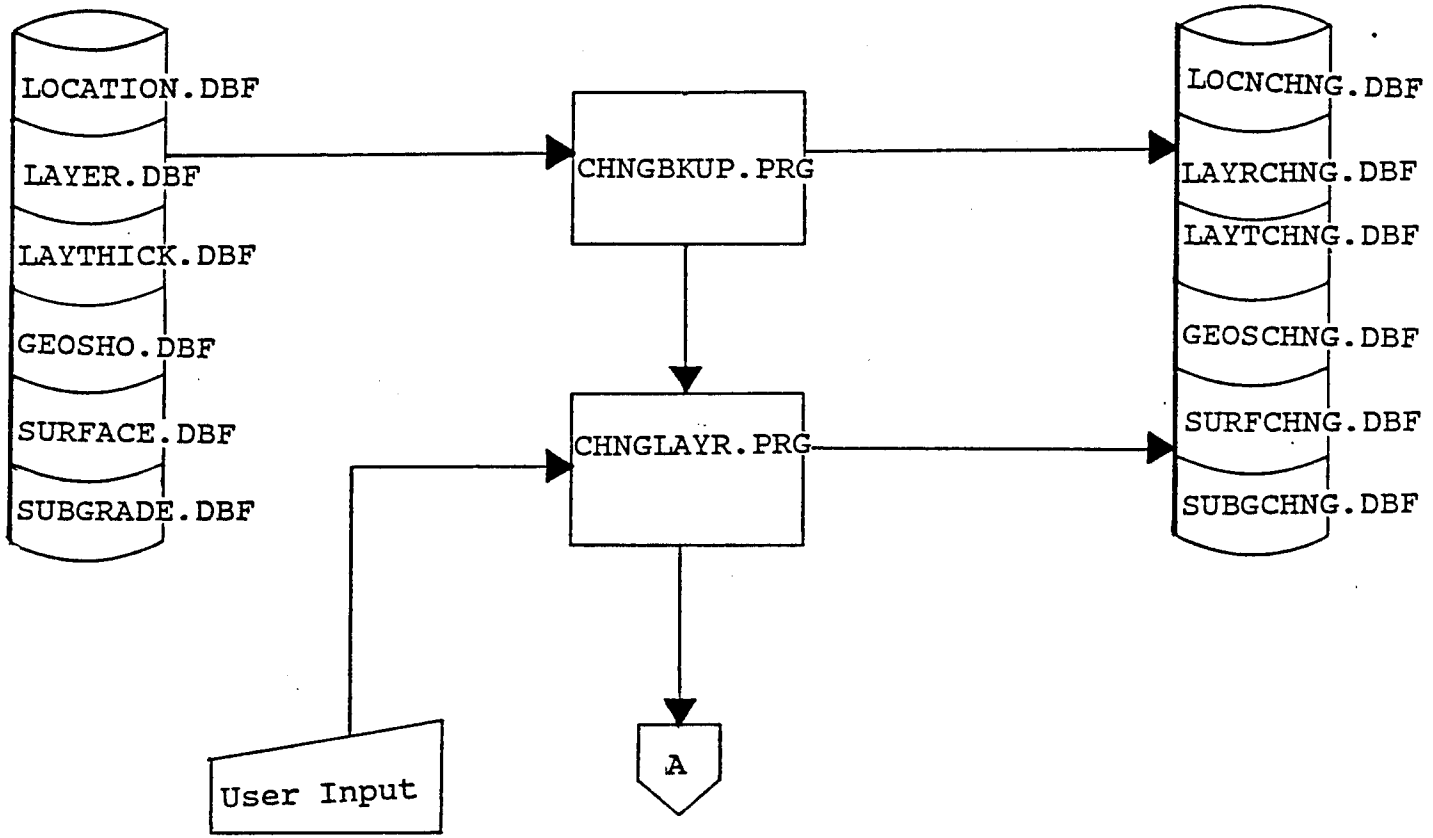


FIGURE 18

INVENTORY DATA - CHANGE PROCESS
PROGRAM FLOW DIAGRAM

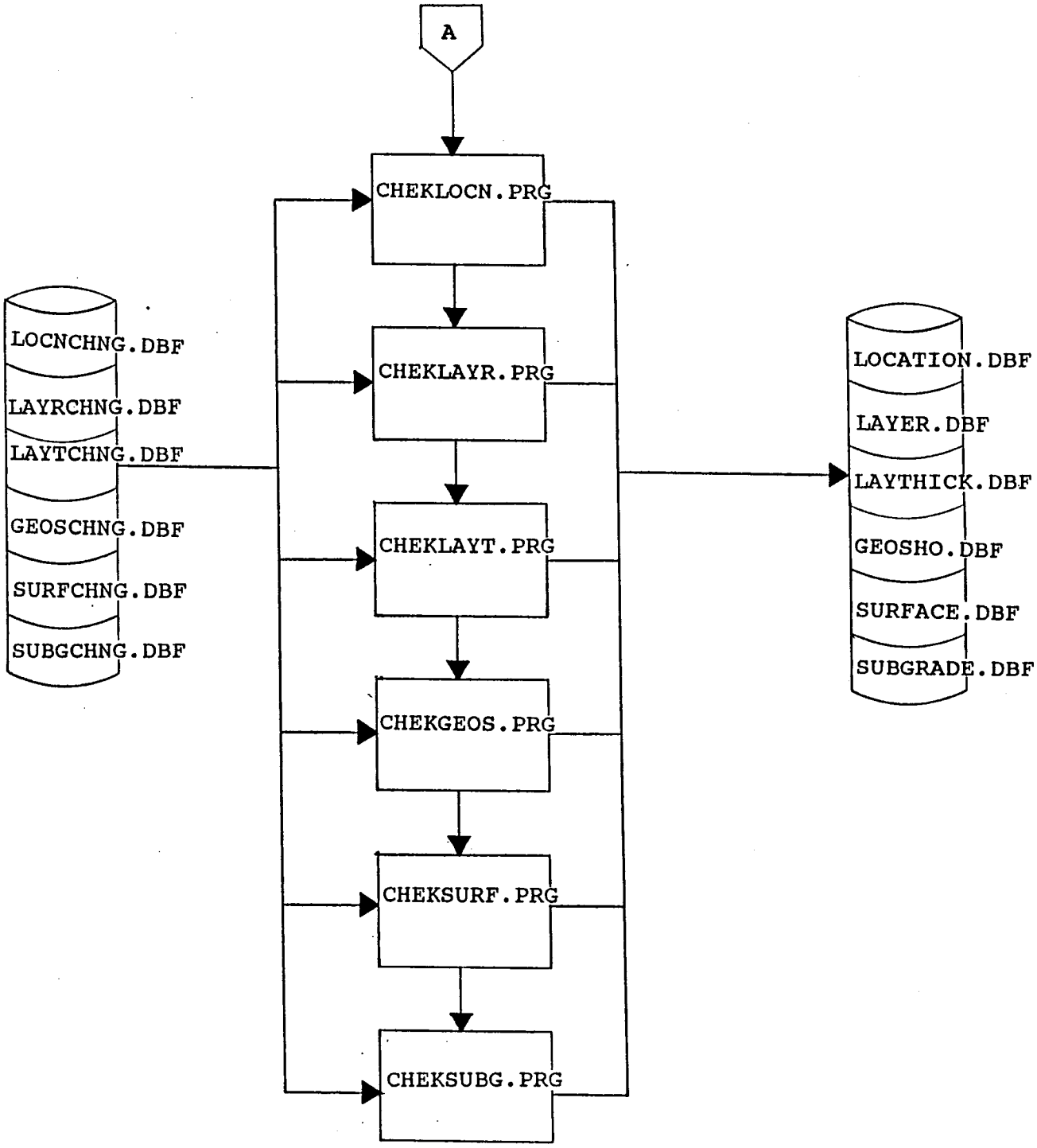


FIGURE 18 (Continued)

PROGRAM SPECIFICATION

Program Name: CHNGLAYR.PRG

Purpose: To display the CHANGE Inventory data menu, Change the Data and to call the programs that Edit & Check the data.

Procedures\Edits:

The following are the procedures:

1) Backup the master files using CHNGBKUP.PRG.
2) Run the Edit/Check procedures to flag errors in the Changed data (CHEKLOCN.PRG, CHEKLAYR.PRG, CHEKLAYT.PRG, CHEKGEOS.PRG, CHEKSURF.PRG and CHEKSUBG.PRG).

Input\Output Files:

	Files	Indices
Temporary files -	LOCNCHNG.DBF	LOCNCHNG.NDX
	LAYRCHNG.DBF	LAYRCHNG.NDX
	LAYTCHNG.DBF	LAYTCHNG.NDX
	GEOSCHNG.DBF	GEOSCHNG.NDX
	SURFCHNG.DBF	SURFCHNG.NDX
	SUBGCHNG.DBF	SUBGCHNG.NDX
Master Files -	LOCATION.DBF	LOCSID.NDX
	LAYER.DBF	LAYNDX.NDX
	LAYTHICK.DBF	LAYTNDX.NDX
	GEOSHO.DBF	GEONDX.NDX
	SURFACE.DBF	SURFNDX.NDX
	SUBGRADE.DBF	SUBGNDX.NDX

Programs Called (See Program Flow Diagram):

dBASE programs	- CHNGBKUP.PRG
	- CHEKGEOS.PRG
	- CHEKLAYR.PRG
	- CHEKLAYT.PRG
	- CHEKLOCN.PRG
	- CHEKSUBG.PRG
	- CHEKSURF.PRG
Screen Format Programs	- GEOSCHNG.FMT
	- LAYRCHNG.FMT
	- LAYTCHNG.FMT
	- LOCNCHNG.FMT
	- SUBGCHNG.FMT
	- SURFCHNG.FMT

CHANGE INVENTORY SCREENS

The following screens are produced by CHNGLAYR.PRG and the screen format files except for the next menu (Edit & Update menu 3.0) which is produced by EDITUPDT.PRG.

```
        TEXAS FLEXIBLE PAVEMENT DATABASE          3.0
                Edit & Update



     1 - Pavement Condition Data
     2 - Inventory Data
     3 - Traffic Data
     4 - Tables

                                  OPTION ===>
```

Choice 2 Inventory Data asks the user if he wants to ADD or CHANGE data to the inventory files. If he wants to CHANGE, CHNGLAYR.PRG program is run which produces the next screen (CHANGE Inventory Data 3.2.C).

```
        TEXAS FLEXIBLE PAVEMENT DATABASE          3.2.C
                EDIT & UPDATE
            CHANGE Inventory Data


     1 - Location
     2 - Layer ID
     3 - Geometric & Shoulder
     4 - Surface
     5 - Subgrade
     6 - Layer Thickness Across The Road

     K - Check the Data Changed

                                  OPTION ===>
```

Choice 1 Location displays the first screen on this page to let the user change Location data.
Choice 2 Layer ID displays the second screen on this page to let the user change add Layer data.

TEXAS FLEXIBLE PAVEMENT DATABASE
EDIT & UPDATE - CHANGE Inventory
Location File

SID Number 13 District 1 County 92

Highway Ident. US 82 Control/Section 45/ 4

Mile Post 22 + 0 TO 24 + 0 Lane Identification R

Mile Point 22.000 TO 24.000 Mile Point Date 6/75

HPMS Sample Number HPMS Section Subdivision 0

Functional Classification 0 Number of Lanes 1

Active ? T Inactive Date 0/ 0 Previous SID 0 Next SID 0

Comment

To exit & keep changes, press CTRL + End keys simultaneously

TEXAS FLEXIBLE PAVEMENT DATABASE
EDIT & UPDATE - CHANGE Inventory
Layer Identification

Sid Number 13

Structure Number	Layr No.	Layer Desc.	Center Thick	Material Type Class.	Date Compltd Job Mnth	Date Compltd Job Year	Widened Mnth	Widened Year
1	1	7	0.00	44	6	31	0	0

To exit & keep changes, press CTRL + End keys simultaneously

Choice 3 Geometric & Shoulder displays the first screen on this page to modify data.
Choice 4 Surface displays the second screen on this page to modify data.

TEXAS FLEXIBLE PAVEMENT DATABASE
EDIT & UPDATE - CHANGE Inventory
Geometric & Shoulder Information

Sid Number 13
Structure Number 1

Type of Pavement (See TTI Codes)	34
Lane Width (Feet)	12
Outside Shoulder Width (Feet)	0
Shoulder Surface Type	1
Shoulder Base Type (See Base Type Code, Table A.6)	0
Shoulder Surface Thickness (Inches)	0.0
Shoulder Base Thickness (Inches)	0.00
Widened Flag (0-2)	1

To exit & keep changes, press CTRL + End keys simultaneously

TEXAS FLEXIBLE PAVEMENT DATABASE
EDIT & UPDATE - CHANGE Inventory
Surface Layer

SID NUMBER 13

Structure Number	Layer Number	Aggregate Application Rate	Type Admixture	Percent Admixture (Mean Asphalt Content)	Asphalt Application Rate
1	3	0 (S.Y./C.Y.)	AC	5.70	0.00 (GAL/S.Y.)

To exit & keep changes, press CTRL + End keys simultaneously

Choice 5 Subgrade displays the first screen on this page to enter data into.

Choice 6 Layer Thickness Across The Road displays the second screen on this page to enter data into.

TEXAS FLEXIBLE PAVEMENT DATABASE
EDIT & UPDATE - CHANGE Inventory
Subgrade File

SID NUMBER 13

Percent Passing No. 200 Sieve	88.8
Texas Triaxial Class	5.3
Liquid Limit	64.5
Plasticity Index	40.4
Permeability Index	0.23

To exit & keep changes, press CTRL + End keys simultaneously

TEXAS FLEXIBLE PAVEMENT DATABASE
EDIT & UPDATE - CHANGE Inventory
Layer Thickness Across The Road

SID NUMBER 13

Structure Number	Layer Number	Thickness - From Center				Distance From Center		
		3rd Pos	2nd Pos	1st Pos	Center	3rd Pos	2nd Pos	1st Pos
1	2	10.00	10.00	6.00	6.00	12.0	9.0	5.0
		(In Inches)				(In Feet)		

To exit & keep changes, press CTRL + End keys simultaneously

PROGRAM LISTING

```
* GEOMETRIC & SHOULDER INFORMATION SCREEN FORMAT FILE - GEOSCHNG.FMT

@  0,  0  CLEAR
@  4, 22  SAY "TEXAS FLEXIBLE PAVEMENT DATABASE"
@  5, 22  SAY "EDIT & UPDATE - CHANGE Inventory"
@  6, 22  SAY "Geometric & Shoulder Information"
@  8, 56  SAY "Sid Number "
@  8, 67  SAY  GEOSCHNG->SID_NO
@  9, 50  SAY "Structure Number"
@  9, 69  SAY  GEOSCHNG->STRUCNUM PICTURE "99"
@ 11,  8  SAY "Type of Pavement (See TTI Codes)"
@ 11, 61  GET  GEOSCHNG->PAVETYP PICTURE "99" RANGE 1,27
@ 12,  8  SAY "Lane Width (Feet)"
@ 12, 61  GET  GEOSCHNG->LANEWID PICTURE "99" RANGE 8,15
@ 13,  8  SAY "Outside Shoulder Width (Feet)"
@ 13, 61  GET  GEOSCHNG->OUTSHOWD PICTURE "99" RANGE 0,15
@ 14,  8  SAY "Shoulder Surface Type"
@ 14, 61  GET  GEOSCHNG->SHOSFTYP PICTURE "99" RANGE 1,6
@ 15,  8  SAY "Shoulder Base Type (See Base Type Code, Table A.6)"
@ 15, 61  GET  GEOSCHNG->SHOBSTYP PICTURE "99" RANGE 21,45
@ 16,  8  SAY "Shoulder Surface Thickness (Inches)"
@ 16, 59  GET  GEOSCHNG->SHOSFTHK PICTURE "99.9" RANGE 0.0,10.1
@ 17,  8  SAY "Shoulder  Base Thickness (Inches)"
@ 17, 58  GET  GEOSCHNG->SHOBSTHK PICTURE "99.99"
@ 18,  8  SAY  "Widened Flag (0-2)"
@ 18, 62  GET  GEOSCHNG->WIDENFLG   PICTURE "9" RANGE 0,2
@  3,  3  TO 20, 74    DOUBLE
@ 22, 10  SAY "To exit & keep changes, press  CTRL + End  keys simultaneously"
```

PROGRAM LISTING

```
* LAYER IDENTIFICATION SCREEN FORMAT FILE - LAYRCHNG.FMT

@  0,  0  CLEAR
@  3, 23  SAY "TEXAS FLEXIBLE PAVEMENT DATABASE"
@  4, 23  SAY "EDIT & UPDATE - CHANGE Inventory"
@  5, 23  SAY "     Layer Identification"
@  7, 55  SAY "Sid Number "
@  7, 66  SAY  LAYRCHNG->SID_NO PICTURE "9999"
@ 10, 54  SAY "Date"
@ 11,  4  SAY "                                    Material  Job Compltd   Widened"
@ 12,  4  SAY "Structure  Layr   Layer  Center   Type     ----------
---------"
@ 13,  4  SAY " Number     No.   Desc.  Thick   Class.    Mnth   Year   Mnth
Year"
@ 14,  4  SAY "---------  -----  -----  ------  --------  ----   ----   ----
----"
@ 16,  8  SAY  LAYRCHNG->STRUCNUM PICTURE "9"
@ 16, 17  SAY  LAYRCHNG->LAYNUM   PICTURE "99"
@ 16, 24  GET  LAYRCHNG->LAYDESC  PICTURE "99" RANGE 1,14
@ 16, 30  GET  LAYRCHNG->CENTITHK  PICTURE "99.99" RANGE 0,25.0
@ 16, 40  GET  LAYRCHNG->LAYMATCL PICTURE "99" RANGE 01,45
@ 16, 48  GET  LAYRCHNG->JOBCMPMO PICTURE "99" RANGE 1,12
@ 16, 55  GET  LAYRCHNG->JOBCMPYR PICTURE "99" RANGE 20,99
@ 16, 62  GET  LAYRCHNG->WIDENLMO PICTURE "99" RANGE 0,12
@ 16, 67  GET  LAYRCHNG->WIDENLYR PICTURE "99" RANGE 0,99
@  2,  2  TO 18, 74    DOUBLE
@ 22, 10  SAY "To exit & keep changes, press  CTRL + End  keys simultaneously"
```

```
* LAYER THICKNESS ACROSS THE ROAD SCREEN FORMAT FILE - LAYTCHNG.FMT

@  0, 0   CLEAR
@  4, 23  SAY "TEXAS FLEXIBLE PAVEMENT DATABASE"
@  5, 23  SAY "EDIT & UPDATE - CHANGE Inventory"
@  6, 23  SAY "Layer Thickness Across The Road"
@  8, 56  SAY "SID NUMBER "
@  8, 67  SAY  LAYTCHNG->SID_NO PICTURE "9999"
@ 10, 25  SAY "Thickness - From Center       Distance From Center"
@ 11,  3  SAY "Structure  Layer   ----------------------------
----------------------"
@ 12,  4  SAY "Number    Number  3rd Pos 2nd Pos 1st Pos Center  3rd Pos 2nd Pos
1st Pos"
@ 13,  3  SAY "---------  ------  ------- ------- ------- ------  ------- -------
-------"
@ 14,  6  SAY  LAYTCHNG->STRUCNUM  PICTURE "99"
@ 14, 16  SAY  LAYTCHNG->LAYNUM    PICTURE "99"
@ 14, 24  GET  LAYTCHNG->FC3THK    PICTURE "99.99" RANGE 0.0,25.0
@ 14, 32  GET  LAYTCHNG->FC2THK    PICTURE "99.99" RANGE 0.0,25.0
@ 14, 40  GET  LAYTCHNG->FC1THK    PICTURE "99.99" RANGE 0.1,25.0
@ 14, 47  GET  LAYTCHNG->CENTTHK   PICTURE "99.99" RANGE 0.1,25.0
@ 14, 56  GET  LAYTCHNG->FC3DIS    PICTURE "99.9" RANGE 0.0,99.0
@ 14, 64  GET  LAYTCHNG->FC2DIS    PICTURE "99.9" RANGE 0.0,99.0
@ 14, 72  GET  LAYTCHNG->FC1DIS    PICTURE "99.9" RANGE 0.0,99.0
@ 15, 34  SAY "(In Inches)"
@ 15, 62  SAY "(In Feet)"
@  2,  0  TO 17, 79    DOUBLE
@ 22, 10  SAY "To exit & keep changes, press  CTRL + End  keys simultaneously"
```

```
*  LOCATION INFORMATION SCREEN FORMAT FILE - LOCNCHNG.FMT

@  2, 23  SAY "TEXAS FLEXIBLE PAVEMENT DATABASE"
@  3, 23  SAY "EDIT & UPDATE - CHANGE Inventory"
@  4, 23  SAY "        Location File"
@  6,  6  SAY "SID Number"
@  6, 18  SAY  LOCNCHNG->SID_NO PICTURE "9999"
@  6, 44  SAY "District"
@  6, 53  GET  LOCNCHNG->HWYDIST PICTURE "99" RANGE 0,99
@  6, 59  SAY "County"
@  6, 66  GET  LOCNCHNG->CNTYNUM PICTURE "999" RANGE 0,999
@  8,  6  SAY "Highway Ident."
@  8, 21  GET  LOCNCHNG->HWYPREFX
@  8, 24  GET  LOCNCHNG->HWYNUM PICTURE "9999"
@  8, 29  GET  LOCNCHNG->HWYSUFFX
@  8, 44  SAY "Control/Section"
@  8, 60  GET  LOCNCHNG->CONTROL PICTURE "9999"
@  8, 64  SAY "/"
@  8, 65  GET  LOCNCHNG->SECTION PICTURE "99"
@ 10,  6  SAY "Mile Post"
@ 10, 17  GET  LOCNCHNG->BEGMPST PICTURE "999"
@ 10, 21  GET  LOCNCHNG->BDISSIGN
@ 10, 23  GET  LOCNCHNG->BMPSTDIS PICTURE "99"
@ 10, 26  SAY "TO"
@ 10, 29  GET  LOCNCHNG->ENDMPST PICTURE "999"
@ 10, 33  GET  LOCNCHNG->EDISSIGN
@ 10, 35  GET  LOCNCHNG->EMPSTDIS PICTURE "99"
@ 10, 44  SAY "Lane Identification"
@ 10, 64  GET  LOCNCHNG->LANEID
@ 12,  6  SAY "Mile Point"
@ 12, 17  GET  LOCNCHNG->BEGMPNT PICTURE "99.999"
@ 12, 24  SAY "TO"
@ 12, 27  GET  LOCNCHNG->ENDMPNT PICTURE "99.999"
@ 12, 44  SAY "Mile Point Date"
@ 12, 61  GET  LOCNCHNG->MPNTMO PICTURE "99"  RANGE 0, 12
@ 12, 63  SAY "/"
@ 12, 64  GET  LOCNCHNG->MPNTYR PICTURE "99"
@ 14,  6  SAY "HPMS Sample Number"
@ 14, 25  GET  LOCNCHNG->HPMSSAM
@ 14, 44  SAY "HPMS Section Subdivision"
@ 14, 69  GET  LOCNCHNG->HPMSSEC PICTURE "9"
@ 16,  6  SAY "Functional Classification"
@ 16, 31  GET  LOCNCHNG->FUNCLAS PICTURE "99" RANGE 1, 7
@ 16, 44  SAY "Number of Lanes"
@ 16, 61  GET  LOCNCHNG->NUMLANES PICTURE "99" RANGE 1, 22
@ 18,  6  SAY "Active ?"
@ 18, 15  GET  LOCNCHNG->ACTVFLAG PICTURE "L"
@ 18, 19  SAY "Inactive Date"
@ 18, 33  GET  LOCNCHNG->INACTMO  PICTURE "99"  RANGE 0, 12
@ 18, 35  SAY "/"
```

```
@ 18, 36  GET  LOCNCHNG->INACTYR PICTURE "99"
@ 18, 41  SAY "Previous SID"
@ 18, 54  GET  LOCNCHNG->PREVSID PICTURE "9999"
@ 18, 61  SAY "Next SID"
@ 18, 70  GET  LOCNCHNG->NEXTSID PICTURE "9999"
@ 20,  6  SAY "Comment"
@ 20, 15  GET  LOCNCHNG->COMMENT PICTURE
"XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX"
@  1,  2  TO 21, 77    DOUBLE
@ 22, 10  SAY "To exit & keep changes, press  CTRL + End  keys simultaneously"
```

```
* SUBGRADE SCREEN FORMAT FILE - SUBGCHNG.FMT

@  0,  0  CLEAR
@  4, 23  SAY "TEXAS FLEXIBLE PAVEMENT DATABASE"
@  5, 23  SAY "EDIT & UPDATE - CHANGE Inventory"
@  6, 23  SAY "        Subgrade File"
@  8, 45  SAY "SID NUMBER"
@  8, 56  SAY  SUBGCHNG->SID_NO PICTURE "9999"
@ 11, 19  SAY "Percent Passing No. 200 Sieve"
@ 11, 51  GET  SUBGCHNG->PPSV200 PICTURE "99.9"
@ 12, 19  SAY "Texas Triaxial Class"
@ 12, 52  GET  SUBGCHNG->TXTRIAXL PICTURE "9.9" RANGE 0.0,6.0
@ 13, 19  SAY "Liquid Limit"
@ 13, 51  GET  SUBGCHNG->LIQLIM PICTURE "99.9" RANGE 0.0,80.0
@ 14, 19  SAY "Plasticity Index"
@ 14, 51  GET  SUBGCHNG->PLASTIX  PICTURE "99.9" RANGE 0.0,70.0
@ 15, 19  SAY "Permeability Index"
@ 15, 50  GET  SUBGCHNG->PERMIX PICTURE "99.99" RANGE 0.00,10.00
@  3, 12  TO 17, 65    DOUBLE
@ 22, 10  SAY "To exit & keep changes, press  CTRL + End  keys simultaneously"
```

```
* SURFACE SCREEN FORMAT FILE - SURFCHNG.FMT

@  0,  0  CLEAR
@  4, 20  SAY "   TEXAS FLEXIBLE PAVEMENT DATABASE"
@  5, 20  SAY "   EDIT & UPDATE - CHANGE Inventory"
@  6, 20  SAY "           Surface Layer"
@  8, 57  SAY "SID NUMBER "
@  8, 68  SAY  SURFCHNG->SID_NO PICTURE "9999"
@ 10,  3  SAY "                                                          Percent"
@ 11,  3  SAY "                      Aggregate                          Admixture
Asphalt"
@ 12,  3  SAY "Structure  Layer   Application      Type          (Mean Asphalt
Application"
@ 13,  3  SA
```

```
*
* SUBSYSTEM:      EDIT & UPDATE INVENTORY dBASE III FILES
* PROGRAM NAME:   CHNGLAYR.PRG                06/27/88
* CALLED FROM:    EDITUPDT.PRG
* PROJECT 2456 - TEXAS FLEXIBLE PAVEMENT DATABASE CONVERSION
* TAMU/TTI
* AUTHOR:         TREVOR X. PEREIRA
* PURPOSE:        THIS PROCEDURE CALLS UP THE RESPECTIVE INVENTORY
*                 DATA FILES TO CHANGE THE DATA IN IT.  THE FOLLOWING PROCEDURES
*                 CAN BE FOUND HERE:
*
*

* First check to see if there is any data in the transaction file.  If
* there is data, display error message and go back to editupdt program.
SET ESCAPE OFF
CLEAR
MCOUNT = 0
USE \PAVEDB\EDITUPDT\LOCN_NEW INDEX \PAVEDB\EDITUPDT\LOCN_NEW
MCOUNT = MCOUNT + RECCOUNT()
USE \PAVEDB\EDITUPDT\LAYR_NEW INDEX \PAVEDB\EDITUPDT\LAYR_NEW
MCOUNT = MCOUNT + RECCOUNT()
USE \PAVEDB\EDITUPDT\LAYT_NEW INDEX \PAVEDB\EDITUPDT\LAYT_NEW
MCOUNT = MCOUNT + RECCOUNT()
USE \PAVEDB\EDITUPDT\GEOS_NEW INDEX \PAVEDB\EDITUPDT\GEOS_NEW
MCOUNT = MCOUNT + RECCOUNT()
USE \PAVEDB\EDITUPDT\SURF_NEW INDEX \PAVEDB\EDITUPDT\SURF_NEW
MCOUNT = MCOUNT + RECCOUNT()
USE \PAVEDB\EDITUPDT\SUBG_NEW INDEX \PAVEDB\EDITUPDT\SUBG_NEW
MCOUNT = MCOUNT + RECCOUNT()
MCONTINUE = .T.
IF MCOUNT <> 0
   CLEAR
   @ 4, 7 SAY "Data is present in the Temporary Inventory files.  You have"
   @ 5, 7 SAY "   to remove these records before you can change the data "
   @ 6, 7 SAY "   in the master files."
   @ 8, 7 SAY "To remove the records, "
   @ 9, 7 SAY "   1) Choose choice '3' from the Edit & Update Menu."
   @ 10,7 SAY "   2) Type 'A' to the next question to Add Data"
   @ 11,7 SAY "   3) Type 'E' to Edit the data in the Temporary Files."
   @ 12,7 SAY "   4) Now you can either"
   @ 13,7 SAY "         A) Delete each record by pressing Ctrl U and PgDn"
   @ 14,7 SAY "              OR"
   @ 15,7 SAY "         B) Correct the Data and run the Edit/Check programs"
   @ 16,7 SAY "              i.e. Choice 'K'."
   @ 19,7 SAY "   Press any key to continue . . ."
   WAIT " "
   CLEAR
   RETURN
ENDIF
MCOUNT2 = 0
```

```
IF FILE('\PAVEDB\EDITUPDT\LOCNCHNG.DBF')
     MCOUNT2 = MCOUNT2 + 1
ENDIF
IF FILE('\PAVEDB\EDITUPDT\LOCNCHNG.NDX')
     MCOUNT2 = MCOUNT2 + 1
ENDIF
IF FILE('\PAVEDB\EDITUPDT\LAYRCHNG.DBF')
     MCOUNT2 = MCOUNT2 + 1
ENDIF
IF FILE('\PAVEDB\EDITUPDT\LAYRCHNG.NDX')
     MCOUNT2 = MCOUNT2 + 1
ENDIF
IF FILE('\PAVEDB\EDITUPDT\LAYTCHNG.DBF')
     MCOUNT2 = MCOUNT2 + 1
ENDIF
IF FILE('\PAVEDB\EDITUPDT\LAYTCHNG.NDX')
     MCOUNT2 = MCOUNT2 + 1
ENDIF
IF FILE('\PAVEDB\EDITUPDT\GEOSCHNG.DBF')
     MCOUNT2 = MCOUNT2 + 1
ENDIF
IF FILE('\PAVEDB\EDITUPDT\GEOSCHNG.NDX')
     MCOUNT2 = MCOUNT2 + 1
ENDIF
IF FILE('\PAVEDB\EDITUPDT\SURFCHNG.DBF')
     MCOUNT2 = MCOUNT2 + 1
ENDIF
IF FILE('\PAVEDB\EDITUPDT\SURFCHNG.NDX')
     MCOUNT2 = MCOUNT2 + 1
ENDIF
IF FILE('\PAVEDB\EDITUPDT\SUBGCHNG.DBF')
     MCOUNT2 = MCOUNT2 + 1
ENDIF
IF FILE('\PAVEDB\EDITUPDT\SUBGCHNG.NDX')
     MCOUNT2 = MCOUNT2 + 1
ENDIF
IF MCOUNT2 = 0
   DO \PAVEDB\EDITUPDT\CHNGBKUP
ENDIF
M_OK = 0
CLEAR
* Display the Inventory Change data menu on the screen
DO WHILE MCONTINUE
 STORE " " TO CHNGCOLL
 DO WHILE .NOT. (CHNGCOLL $ '123456K')
   @  3, 21  SAY "TEXAS FLEXIBLE PAVEMENT DATABASE      3.3.C"
   @  4, 21  SAY "        EDIT & UPDATE"
   @  5, 21  SAY "    CHANGE Inventory Data"
   @  8, 15  SAY "1 - Location"
   @  9, 15  SAY "2 - Layer ID"
   @ 10, 15  SAY "3 - Geometric & Shoulder"
   @ 11, 15  SAY "4 - Surface"
   @ 12, 15  SAY "5 - Subgrade"
   @ 13, 15  SAY "6 - Layer Thickness Across The Road"
```

```
  @ 15, 15  SAY "K - Check the Data Changed"
  @ 17, 40  SAY "OPTION ======> " GET CHNGCOLL
  @  2,  9  TO 20, 65    DOUBLE
  READ
  IF READKEY() = 12
     MCONTINUE = .F.
     SET PROC TO
     EXIT
  ENDIF
  IF (CHNGCOLL $ '123456')
     @ 19, 23  SAY "Please enter SID Number ======> " GET MSID_NO PICTURE "9999"
     READ
     CLEAR
  ENDIF
ENDDO
IF MCONTINUE
  CLEAR
  DO CASE
   CASE CHNGCOLL = "1"
     USE \PAVEDB\EDITUPDT\LOCNCHNG INDE \PAVEDB\EDITUPDT\LOCNCHNG
     SEEK MSID_NO
     SET FORMAT TO \PAVEDB\EDITUPDT\LOCNCHNG
     CHANGE
     SET FORMAT TO
   CASE CHNGCOLL = "2"
     USE \PAVEDB\EDITUPDT\LAYRCHNG INDE \PAVEDB\EDITUPDT\LAYRCHNG
     SEEK STR(MSID_NO,4)
     SET FORMAT TO \PAVEDB\EDITUPDT\LAYRCHNG
     CHANGE
     SET FORMAT TO
   CASE CHNGCOLL = "3"
     USE \PAVEDB\EDITUPDT\GEOSCHNG INDE \PAVEDB\EDITUPDT\GEOSCHNG
     SEEK STR(MSID_NO,4)
     SET FORMAT TO \PAVEDB\EDITUPDT\GEOSCHNG
     CHANGE
     SET FORMAT TO
   CASE CHNGCOLL = "4"
     USE \PAVEDB\EDITUPDT\SURFCHNG INDE \PAVEDB\EDITUPDT\SURFCHNG
     SEEK STR(MSID_NO,4)
     SET FORMAT TO \PAVEDB\EDITUPDT\SURFCHNG
     CHANGE
     SET FORMAT TO
   CASE CHNGCOLL = "5"
     USE \PAVEDB\EDITUPDT\SUBGCHNG INDE \PAVEDB\EDITUPDT\SUBGCHNG
     SEEK STR(MSID_NO,4)
     SET FORMAT TO \PAVEDB\EDITUPDT\SUBGCHNG
     CHANGE
     SET FORMAT TO
   CASE CHNGCOLL = "6"
     USE \PAVEDB\EDITUPDT\LAYTCHNG INDE \PAVEDB\EDITUPDT\LAYTCHNG
     SEEK STR(MSID_NO,4)
     SET FORMAT TO \PAVEDB\EDITUPDT\LAYTCHNG
     CHANGE
     SET FORMAT TO
```

```
    CASE CHNGCOLL = "K"
      DO \PAVEDB\EDITUPDT\CHEKLOCN
      DO \PAVEDB\EDITUPDT\CHEKLAYR
      DO \PAVEDB\EDITUPDT\CHEKLAYT
      DO \PAVEDB\EDITUPDT\CHEKGEOS
      DO \PAVEDB\EDITUPDT\CHEKSURF
      DO \PAVEDB\EDITUPDT\CHEKSUBG
      DO \PAVEDB\EDITUPDT\COPYLAYR
   ENDCASE
   CLEAR
  ENDIF
  CLEAR
ENDDO
RETURN
```

```
*
* SUBSYSTEM:        EDIT & UPDATE
* PROGRAM NAME:     CHNGBKUP.PRG              06/21/88
* CALLED FROM:      CHNGLAYR.PRG
* MODIFIED ON:      09/20/88
* PROJECT 2456 -    TEXAS FLEXIBLE PAVEMENT DATABASE CONVERSION
* TAMU/TTI
* AUTHOR:           TREVOR X. PEREIRA
* PURPOSE:          BACK UP MASTER INVENTORY FILES (ORIGINAL FILES)
*

SET TALK OFF
SET ECHO OFF
SET ESCAPE OFF
CLEAR
MBACKDRV = MDRIVE2 + ":"
CLEAR TYPE
SET TYPE TO 0
STORE 0 TO MSIZE, MNUMFIELDS, MHEADER, MTOTALSIZE, MCOUNT
STORE "" TO MNULL
@ 5,5 say "The MASTER files have to be backed up first. "
@ 9, 5 say "Place a FORMATTED DISKETTE in drive " + MBACKDRV + " and close the
drive."
@ 12,5 say "If you DO NOT want to continue, press the ESC key."
@ 13,5 say "Otherwise press any other key to continue."
READ
IF READKEY() = 12
   CLEAR
   MCONTINUE = .F.
   RETURN
ENDIF

*
* Backing up Location Data
*
CLEAR
MOVERWRITE = " "
SET DEFAULT TO &MBACKDRV
DO WHILE FILE('LOCATION.DBF')
  ? "LOCATION FILE exists on drive " + MBACKDRV
  ACCEPT "Overwrite it (Y/N) " TO MOVERWRITE
  IF MOVERWRITE $ "Yy"
     DELE FILE LOCATION.DBF
  ELSE
     CLEAR
     ? "Replace diskette in drive " + MBACKDRV + " with another diskette"
     ? "Then press any key to continue"
     WAIT " "
  ENDIF
ENDDO
```

```
SET DEFAULT TO &MDRIVE

SET TALK OFF
* Checking diskspace on backup drive and backing up LOCATION master files
USE \PAVEDB\FILES\LOCATION
STORE RECCOUNT() * RECSIZE() TO MSIZE
MNULL = ""
DO WHILE MNULL < FIELD(MNUMFIELDS + 1)
   MNUMFIELDS = MNUMFIELDS + 1
ENDDO
MHEADER = (32 * MNUMFIELDS) + 34
MTOTALSIZE = MSIZE + MHEADER + 20
SET DEFAULT TO &MBACKDRV
MDISKSPACE = DISKSPACE()
SET DEFAULT TO &MDRIVE
DO WHILE MDISKSPACE < MTOTALSIZE
  ? "Not enough space on diskette in drive " + MBACKDRV
  ? "Please replace with another diskette and press any key to continue"
  WAIT " "
  IF READKEY() = 12
     ? "PROCESS ABORTED"
     RETURN
  ENDIF
  SET DEFAULT TO &MBACKDRV
  MDISKSPACE = DISKSPACE()
  SET DEFAULT TO &MDRIVE
ENDDO
USE
? "Please wait.  Backing up Location data files . . ."
IF FILE('\pavedb\files\backup\LOCATION.DBF')
  DELE FILE \PAVEDB\FILES\BACKUP\LOCATION.DBF
ENDIF
SET TALK ON
COPY FILE \PAVEDB\FILES\LOCATION.DBF TO &MBACKDRV\LOCATION.DBF
COPY FILE \PAVEDB\FILES\LOCATION.DBF TO \PAVEDB\FILES\BACKUP\LOCATION.DBF
SET TALK OFF

*
* backing up layer id files
*
CLEAR
MOVERWRITE = " "
SET DEFAULT TO &MBACKDRV
DO WHILE FILE('LAYER.DBF')
  ? "LAYER FILE exists on drive " + MBACKDRV
  ACCEPT "Overwrite it (Y/N) " TO MOVERWRITE
  IF MOVERWRITE $ "Yy"
     DELE FILE LAYER.DBF
  ELSE
     CLEAR
     ? "Replace diskette in drive " + MBACKDRV + " with another diskette"
     ? "Then press any key to continue"
     WAIT " "
  ENDIF
```

```
ENDDO
CLEAR
SET DEFAU TO &MDRIVE

* Checking diskspace on backup drive and backing up Layer ID master files
USE \PAVEDB\FILES\LAYER
STORE RECCOUNT() * RECSIZE() TO MSIZE
MNULL = ""
DO WHILE MNULL < FIELD(MNUMFIELDS + 1)
   MNUMFIELDS = MNUMFIELDS + 1
ENDDO
MHEADER = (32 * MNUMFIELDS) + 34
MTOTALSIZE = MSIZE + MHEADER + 20
SET DEFAULT TO &MBACKDRV
MDISKSPACE = DISKSPACE()
SET DEFAULT TO &MDRIVE
DO WHILE MDISKSPACE < MTOTALSIZE
  ? "Not enough space on diskette in drive " + MBACKDRV
  ? "Please replace with another diskette and press any key to continue"
  WAIT " "
  IF READKEY() = 12
     ? "PROCESS ABORTED"
     RETURN
  ENDIF
  SET DEFAULT TO &MBACKDRV
  MDISKSPACE = DISKSPACE()
  SET DEFAULT TO &MDRIVE
ENDDO
clear
? "Please wait.  Backing up Layer ID . . ."
USE
IF FILE('\pavedb\files\backup\LAYER.DBF')
  DELE FILE \PAVEDB\FILES\BACKUP\LAYER.DBF
ENDIF
SET TALK ON
COPY FILE \PAVEDB\FILES\LAYER.DBF TO &MBACKDRV\LAYER.DBF
COPY FILE \PAVEDB\FILES\LAYER.DBF TO \PAVEDB\FILES\BACKUP\LAYER.DBF
SET TALK OFF

*
* backing up LAYTHICK THICKNESS files
*
CLEAR
MOVERWRITE = " "
SET DEFAU TO &MBACKDRV
DO WHILE FILE('LAYTHICK.DBF')
  ? "LAYER THICKNESS FILE exists on drive " + MBACKDRV
  ACCEPT "Overwrite it (Y/N) " TO MOVERWRITE
  IF MOVERWRITE $ "Yy"
     DELE FILE LAYTHICK.DBF
  ELSE
     CLEAR
     ? "Replace diskette in drive " + MBACKDRV + " with another diskette"
     ? "Then press any key to continue"
```

```
    WAIT " "
  ENDIF
ENDDO
SET DEFAU TO &MDRIVE

* Checking diskspace on backup drive and backing up LAYTHICK ID master files
USE \PAVEDB\FILES\LAYTHICK
STORE RECCOUNT() * RECSIZE() TO MSIZE
MNULL = ""
DO WHILE MNULL < FIELD(MNUMFIELDS + 1)
   MNUMFIELDS = MNUMFIELDS + 1
ENDDO
MHEADER = (32 * MNUMFIELDS) + 34
MTOTALSIZE = MSIZE + MHEADER + 20
SET DEFAULT TO &MBACKDRV
MDISKSPACE = DISKSPACE()
SET DEFAULT TO &MDRIVE
DO WHILE MDISKSPACE < MTOTALSIZE
  ? "Not enough space on diskette in drive " + MBACKDRV
  ? "Please replace with another diskette and press any key to continue"
  WAIT " "
  IF READKEY() = 12
     ? "PROCESS ABORTED"
     RETURN
  ENDIF
  SET DEFAULT TO &MBACKDRV
  MDISKSPACE = DISKSPACE()
  SET DEFAULT TO &MDRIVE
ENDDO
? "Please wait.  Backing up LAYER THICKNESS Across the Road files . . ."
USE
IF FILE('\pavedb\files\backup\LAYTHICK.DBF')
  DELE FILE \PAVEDB\FILES\BACKUP\LAYTHICK.DBF
ENDIF
SET TALK ON
COPY FILE \PAVEDB\FILES\LAYTHICK.DBF TO &MBACKDRV\LAYTHICK.DBF
COPY FILE \PAVEDB\FILES\LAYTHICK.DBF TO \PAVEDB\FILES\BACKUP\LAYTHICK.DBF
SET TALK OFF

*
* backing up GEOMETRIC & SHOULDER files
*
CLEAR
MOVERWRITE = " "
SET DEFAU TO &MBACKDRV
DO WHILE FILE('GEOSHO.DBF')
  ? "GEOMETRIC & SHOULDER FILE exists on drive " + MBACKDRV
  ACCEPT "Overwrite it (Y/N) " TO MOVERWRITE
  IF MOVERWRITE $ "Yy"
     DELE FILE GEOSHO.DBF
  ELSE
     CLEAR
     ? "Replace diskette in drive " + MBACKDRV + " with another diskette"
     ? "Then press any key to continue"
```

```
    WAIT " "
  ENDIF
ENDDO
SET DEFAU TO &MDRIVE

* Checking diskspace on backup drive and backing up GEOMETRIC & SHOULDER master
files
USE \PAVEDB\FILES\GEOSHO
STORE RECCOUNT() * RECSIZE() TO MSIZE
MNULL = ""
DO WHILE MNULL < FIELD(MNUMFIELDS + 1)
   MNUMFIELDS = MNUMFIELDS + 1
ENDDO
MHEADER = (32 * MNUMFIELDS) + 34
MTOTALSIZE = MSIZE + MHEADER + 20
SET DEFAULT TO &MBACKDRV
MDISKSPACE = DISKSPACE()
SET DEFAULT TO &MDRIVE
DO WHILE MDISKSPACE < MTOTALSIZE
  ? "Not enough space on diskette in drive " + MBACKDRV
  ? "Please replace with another diskette and press any key to continue"
  WAIT " "
  IF READKEY() = 12
     ? "PROCESS ABORTED"
     RETURN
  ENDIF
  SET DEFAULT TO &MBACKDRV
  MDISKSPACE = DISKSPACE()
  SET DEFAULT TO &MDRIVE
ENDDO
? "Please wait.  Backing up GEOMETRIC & SHOULDER files . . ."
USE
IF FILE('\pavedb\files\backup\GEOSHO.DBF')
  DELE FILE \PAVEDB\FILES\BACKUP\GEOSHO.DBF
ENDIF
SET TALK ON
COPY FILE \PAVEDB\FILES\GEOSHO.DBF TO &MBACKDRV\GEOSHO.DBF
COPY FILE \PAVEDB\FILES\GEOSHO.DBF TO \PAVEDB\FILES\BACKUP\GEOSHO.DBF
SET TALK OFF


*
* backing up SURFACE THICKNESS files
*
CLEAR
MOVERWRITE = " "
SET DEFAULT TO &MBACKDRV
DO WHILE FILE('SURFACE.DBF')
  ? "SURFACE FILE exists on drive " + MBACKDRV
  ACCEPT "Overwrite it (Y/N) " TO MOVERWRITE
  IF MOVERWRITE $ "Yy"
     DELE FILE SURFACE.DBF
  ELSE
     CLEAR
```

```
      ? "Replace diskette in drive " + MBACKDRV + " with another diskette"
      ? "Then press any key to continue"
      WAIT " "
   ENDIF
ENDDO
CLEAR
SET DEFAULT TO &MDRIVE

* Checking diskspace on backup drive and backing up SURFACE master files
USE \PAVEDB\FILES\SURFACE
STORE RECCOUNT() * RECSIZE() TO MSIZE
MNULL = ""
DO WHILE MNULL < FIELD(MNUMFIELDS + 1)
   MNUMFIELDS = MNUMFIELDS + 1
ENDDO
MHEADER = (32 * MNUMFIELDS) + 34
MTOTALSIZE = MSIZE + MHEADER + 20
SET DEFAULT TO &MBACKDRV
MDISKSPACE = DISKSPACE()
SET DEFAULT TO &MDRIVE
DO WHILE MDISKSPACE < MTOTALSIZE
   ? "Not enough space on diskette in drive " + MBACKDRV
   ? "Please replace with another diskette and press any key to continue"
   WAIT " "
   IF READKEY() = 12
      ? "PROCESS ABORTED"
      RETURN
   ENDIF
   SET DEFAULT TO &MBACKDRV
   MDISKSPACE = DISKSPACE()
   SET DEFAULT TO &MDRIVE
ENDDO
USE
CLEAR
? "Please wait.  Backing up SURFACE files . . ."
IF FILE('\pavedb\files\backup\SURFACE.DBF')
   DELE FILE \PAVEDB\FILES\BACKUP\SURFACE.DBF
ENDIF
SET TALK ON
COPY FILE \PAVEDB\FILES\SURFACE.DBF TO &MBACKDRV\SURFACE.DBF
COPY FILE \PAVEDB\FILES\SURFACE.DBF TO \PAVEDB\FILES\BACKUP\SURFACE.DBF
SET TALK OFF

*
* backing up SUBGRADE THICKNESS files
*
CLEAR
MOVERWRITE = " "
SET DEFAU TO &MBACKDRV
DO WHILE FILE('SUBGRADE.DBF')
   ? "SUBGRADE FILE exists on drive " + MBACKDRV
   ACCEPT "Overwrite it (Y/N) " TO MOVERWRITE
   IF MOVERWRITE $ "Yy"
      DELE FILE SUBGRADE.DBF
```

```
  ELSE
     CLEAR
     ? "Replace diskette in drive " + MBACKDRV + " with another diskette"
     ? "Then press any key to continue"
     WAIT " "
  ENDIF
ENDDO
CLEAR
SET TALK OFF
SET DEFAU TO &MDRIVE

* Checking diskspace on backup drive and backing up SUBGRADE master files
USE \PAVEDB\FILES\SUBGRADE
STORE RECCOUNT() * RECSIZE() TO MSIZE
MNULL = ""
DO WHILE MNULL < FIELD(MNUMFIELDS + 1)
   MNUMFIELDS = MNUMFIELDS + 1
ENDDO
MHEADER = (32 * MNUMFIELDS) + 34
MTOTALSIZE = MSIZE + MHEADER + 20
SET DEFAULT TO &MBACKDRV
MDISKSPACE = DISKSPACE()
SET DEFAULT TO &MDRIVE
DO WHILE MDISKSPACE < MTOTALSIZE
  ? "Not enough space on diskette in drive " + MBACKDRV
  ? "Please replace with another diskette and press any key to continue"
  WAIT " "
  IF READKEY() = 12
     ? "PROCESS ABORTED"
     RETURN
  ENDIF
  SET DEFAULT TO &MBACKDRV
  MDISKSPACE = DISKSPACE()
  SET DEFAULT TO &MDRIVE
ENDDO
USE
? "Please wait.  Backing up SUBGRADE files . . ."
IF FILE('\pavedb\files\backup\SUBGRADE.DBF')
  DELE FILE \PAVEDB\FILES\BACKUP\SUBGRADE.DBF
ENDIF
SET TALK ON
COPY FILE \PAVEDB\FILES\SUBGRADE.DBF TO &MBACKDRV\SUBGRADE.DBF
COPY FILE \PAVEDB\FILES\SUBGRADE.DBF TO \PAVEDB\FILES\BACKUP\SUBGRADE.DBF
SET TALK OFF

CLEAR
? "Please wait.  Setting up Files . . . ."
SET TALK ON
SET SAFETY OFF
COPY FILE \PAVEDB\FILES\LOCATION.DBF TO \PAVEDB\EDITUPDT\LOCNCHNG.DBF
COPY FILE \PAVEDB\INDEXES\LOCSID.NDX TO \PAVEDB\EDITUPDT\LOCNCHNG.NDX
COPY FILE \PAVEDB\FILES\LAYER.DBF TO \PAVEDB\EDITUPDT\LAYRCHNG.DBF
COPY FILE \PAVEDB\INDEXES\LAYNDX.NDX TO \PAVEDB\EDITUPDT\LAYRCHNG.NDX
COPY FILE \PAVEDB\FILES\LAYTHICK.DBF TO \PAVEDB\EDITUPDT\LAYTCHNG.DBF
```

```
COPY FILE \PAVEDB\INDEXES\LAYINDX.NDX TO \PAVEDB\EDITUPDT\LAYTCHNG.NDX
COPY FILE \PAVEDB\FILES\GEOSHO.DBF TO \PAVEDB\EDITUPDT\GEOSCHNG.DBF
COPY FILE \PAVEDB\INDEXES\GEONDX.NDX TO \PAVEDB\EDITUPDT\GEOSCHNG.NDX
COPY FILE \PAVEDB\FILES\SURFACE.DBF TO \PAVEDB\EDITUPDT\SURFCHNG.DBF
COPY FILE \PAVEDB\INDEXES\SURFNDX.NDX TO \PAVEDB\EDITUPDT\SURFCHNG.NDX
COPY FILE \PAVEDB\FILES\SUBGRADE.DBF TO \PAVEDB\EDITUPDT\SUBGCHNG.DBF
COPY FILE \PAVEDB\INDEXES\SUBGNDX.NDX TO \PAVEDB\EDITUPDT\SUBGCHNG.NDX
SET SAFETY ON
SET TALK OFF
SET ECHO OFF
SET TYPE TO 20
SET ESCAPE OFF
RETURN
```

```
*
* SUBSYSTEM:      EDIT & UPDATE INVENTORY FILES
* PROGRAM NAME:   CHEKGEOS.PRG       06/21/88
* CALLED FROM:    CHNGLAYR.PRG
* PROJECT 2456 - TEXAS FLEXIBLE PAVEMENT DATABASE CONVERSION
* TAMU/TTI
* AUTHOR:         TREVOR X. PEREIRA
* PURPOSE:        PROGRAM TO LIST ECORDS THAT ARE IN ERROR
*

* set parameters
set talk oFF
set echo oFF
CLEAR
M_ERROR = 0
SET ALTE TO GEOSERR

M_TITLE  = "GEOMETRIC & SHOULDER FILE - LIST OF RECORDS NOT IN LAYER ID FILE"
M_TITLE2 = "------------------------------------------------------------------"
SET ALTE ON
? M_TITLE
? M_TITLE2

* assign database files to different work areas
SELECT 1
   USE \PAVEDB\EDITUPDT\LAYRCHNG INDEX \PAVEDB\EDITUPDT\LAYRCHNG

SELECT 2
   USE \PAVEDB\EDITUPDT\GEOSCHNG INDEX \PAVEDB\EDITUPDT\GEOSCHNG

SELE 2
DO WHILE .NOT. EOF()
   * store to memory variables
   MSID = SID_NO
   MSTRUC = STRUCNUM

   SELECT 1
   * find the record in layer that equals the sid number and structure
   * number
   SEEK STR(MSID,4)+STR(MSTRUC,2)

   * if record is not found, print the sid number
   IF .NOT. FOUND()
      SELE 2
      DISPLAY FIELD SID_NO, STRUCNUM OFF
      M_ERROR = M_ERROR + 1
   ENDIF
   SELE 2
   SKIP
ENDDO
```

```
SELE 1
USE
SELE 2
USE
? " "
? " "
? " "
? " "
? " "
? " "

*
* TO LIST ALL OUT OF RANGE DATA FOR THE GEOMETRIC AND
* INFORMATION FILE
*

USE \PAVEDB\EDITUPDT\GEOSCHNG INDEX \PAVEDB\EDITUPDT\GEOSCHNG

* set up page heading
M_TITLE  = "GEOMETRIC & SHOULDER FILE - OUT OF RANGE RECORDS"
M_TITLE2 = "------------------------------------------------"
? M_TITLE
? M_TITLE2

* check data range
DO WHILE .NOT. EOF()
    IF STRUCNUM < 1 .OR. STRUCNUM > 9
       DISPLAY FIELDS SID_NO, STRUCNUM OFF
       M_ERROR = M_ERROR + 1
    ENDIF
    IF PAVETYP < 1 .OR. PAVETYP > 37
       DISPLAY FIELDS SID_NO, STRUCNUM, PAVETYP OFF
       M_ERROR = M_ERROR + 1
    ENDIF
    IF LANEWID < 8 .OR. LANEWID > 15
       IF LANEWID <> 0
           DISPLAY FIELDS SID_NO, STRUCNUM, LANEWID OFF
           M_ERROR = M_ERROR + 1
       ENDIF
    ENDIF
    IF OUTSHOWD < 0 .OR. OUTSHOWD > 15
       DISPLAY FIELDS SID_NO, STRUCNUM, OUTSHOWD OFF
       M_ERROR = M_ERROR + 1
    ENDIF
    IF SHOSFTYP < 1 .OR. SHOSFTYP > 6
       DISPLAY FIELDS SID_NO, STRUCNUM, SHOSFTYP OFF
       M_ERROR = M_ERROR + 1
    ENDIF
    IF SHOBSTYP < 21 .OR. SHOBSTYP > 49
       IF SHOBSTYP <> 0
           DISPLAY FIELDS SID_NO, STRUCNUM, SHOBSTYP OFF
           M_ERROR = M_ERROR + 1
       ENDIF
    ENDIF
```

```
    IF WIDENFLG < 0 .OR. WIDENFLG > 2
       DISPLAY FIELDS SID_NO, STRUCNUM, WIDENFLG OFF
       M_ERROR = M_ERROR + 1
    ENDIF
    IF SHOSFTHK < 0 .OR. SHOSFTHK > 10.1
       DISPLAY FIELDS SID_NO, STRUCNUM, SHOSFTHK OFF
       M_ERROR = M_ERROR + 1
    ENDIF
    SKIP
ENDDO
CLEAR
? " "
? " "
? " "
? " "
? " "
? " "

*
* TO LIST OUT ALL INCORRECT SID NUMBERS FOR GEOMETRIC & SHOULDER
*

* SET PARAMETERS
set talk off
set echo off

* SET PAGE HEADING
M_TITLE  = "GEOMETRIC & SHOULDER FILE - LIST OF INCORRECT SID NUMBERS"
M_TITLE2 = "-----------------------------------------------------------"
? M_TITLE
? M_TITLE2

STORE 0 TO VAR1, VAR2, VAR3, VAR4, COMPARE
DO WHILE  .NOT. EOF()
    * calculate the correct check digit for the Sid number
    VAR1 = INT(SID_NO/1000)
    VAR2 = MOD(INT(SID_NO/100),10)
    VAR2 = VAR2 * 2
    VAR3 = MOD(INT(SID_NO/10),10)
    VAR3 = VAR3 * 3
    VAR4 = MOD(VAR1,10)+VAR2+VAR3
    COMPARE = MOD(VAR4,10)

    * compare the entered check digit with the calculated check digit
    IF COMPARE <> MOD(SID_NO,10)
        DISPLAY SID_NO, STRUCNUM OFF
        M_ERROR = M_ERROR + 1
    ENDIF
    IF SID_NO = 0
        DISPLAY SID_NO, STRUCNUM OFF
        M_ERROR = M_ERROR + 1
    ENDIF
    SKIP
ENDDO
```

```
* reset parameters
CLEAR
? " "
? " "
? " "
? " "
? " "
? " "

*
* TO LIST THE LAYER IDENTIFICATION RECORDS THAT ARE NOT
* PRESENT IN GEOMETRIC AND SHOULDER
*

* set parameters
CLEAR
M_ERROR1 = "GEOMETRIC & SHOULDER FILE - LIST OF ALL LAYER ID RECORDS"
M_ERROR2 = "    NOT PRESENT IN THE GEPMETRIC & SHOULDER FILE"
M_ERROR3 = "------------------------------------------------------------"
? M_ERROR1
? M_ERROR2
? M_ERROR3

* assign files to different work areas
SELECT 1
   USE \PAVEDB\EDITUPDT\LAYRCHNG INDEX \PAVEDB\EDITUPDT\LAYRCHNG
SELECT 2
   USE \PAVEDB\EDITUPDT\GEOSCHNG INDEX \PAVEDB\EDITUPDT\GEOSCHNG

SELECT 1
DO WHILE .NOT. EOF()
   MSID = SID_NO
   MSTRUC = STRUCNUM
   STRC_COMP = STRUCNUM

   * find the 1st layer identification record in geometric and shoulder
   * file
   SELECT 2
   Seek str(MSID,4)+str(MSTRUC,2)
   IF .NOT. FOUND()
      SELE 1
      DISPLAY FIELDS SID_NO, STRUCNUM OFF
      M_ERROR = M_ERROR + 1
   ENDIF
   SELECT 1
   SKIP

   * skip the rest of the records for the same sid number
   DO WHILE MSID = SID_NO .AND. STRC_COMP = STRUCNUM
      SKIP
   ENDDO
ENDDO
SELE 1
use
```

```
SELE 2
USE
USE \PAVEDB\EDITUPDT\GEOSCHNG INDEX \PAVEDB\EDITUPDT\GEOSCHNG
PACK
USE
CLEAR
? CHR(12)
SET ALTE OFF
CLOSE ALTE
IF M_ERROR <> 0
   M_OK = M_OK + 1
ENDIF
RETURN
```

```
*
* SUBSYSTEM:      EDIT & UPDATE INVENTORY FILES
* PROGRAM NAME:   CHEKLAYR.PRG        06/21/88
* CALLED FROM:    CHNGLAYR.PRG
* PROJECT 2456 - TEXAS FLEXIBLE PAVEMENT DATABASE SYSTEM
* TAMU/TTI
* AUTHOR:         TREVOR X. PEREIRA
* PURPOSE:        CHECKS FOR ERRORS IN THE LAYER FILE
*


* set parameters
SET TALK OFF
SET ECHO OFF
CLEAR
SET ALTE TO LAYRERR
USE \PAVEDB\EDITUPDT\LAYRCHNG INDEX \PAVEDB\EDITUPDT\LAYRCHNG
M_ERROR  = "LAYER ID FILE - INCORRECT LAYER DESCRIPTION OR MATERIAL
CLASSIFICATION"
M_ERROR2 =
"-----------------------------------------------------------------"
SET ALTE ON
? M_ERROR
? M_ERROR2
M_ERRORCK = 0
* check layer description with layer material classification
DO WHILE .NOT. EOF()
   DO CASE
      CASE LAYDESC = 7
          IF LAYMATCL < 41 .OR. LAYMATCL > 45
             DISPLAY FIELDS SID_NO, STRUCNUM, LAYNUM, LAYDESC, LAYMATCL OFF
             M_ERRORCK = M_ERRORCK + 1
          ENDIF
      CASE LAYDESC = 6
          IF LAYMATCL < 21 .OR. LAYMATCL > 33
             IF LAYMATCL <> 17
                DISPLAY FIELDS SID_NO, STRUCNUM, LAYNUM, LAYDESC, LAYMATCL OFF
                M_ERRORCK = M_ERRORCK + 1
             ENDIF
          ENDIF
      CASE LAYDESC = 5
          IF LAYMATCL < 21 .OR. LAYMATCL > 27
             IF LAYMATCL <> 17
                DISPLAY FIELDS SID_NO, STRUCNUM, LAYNUM, LAYDESC, LAYMATCL OFF
                M_ERRORCK = M_ERRORCK + 1
             ENDIF
          ENDIF
      CASE LAYDESC = 4
          IF LAYMATCL < 1 .OR. LAYMATCL > 4
             DISPLAY FIELDS SID_NO, STRUCNUM, LAYNUM, LAYDESC, LAYMATCL OFF
```

```
            M_ERRORCK = M_ERRORCK + 1
         ENDIF
      CASE LAYDESC = 3
         IF LAYMATCL < 1 .OR. LAYMATCL > 17
            DISPLAY FIELDS SID_NO, STRUCNUM, LAYNUM, LAYDESC, LAYMATCL OFF
            M_ERRORCK = M_ERRORCK + 1
         ENDIF
      CASE LAYDESC = 2
         IF LAYMATCL <> 11
            DISPLAY FIELDS SID_NO, STRUCNUM, LAYNUM, LAYDESC, LAYMATCL OFF
            M_ERRORCK = M_ERRORCK + 1
         ENDIF
      CASE LAYDESC = 1
         IF LAYMATCL < 1 .OR. LAYMATCL > 16
            DISPLAY FIELDS SID_NO, STRUCNUM, LAYNUM, LAYDESC, LAYMATCL OFF
            M_ERRORCK = M_ERRORCK + 1
         ENDIF
      CASE LAYDESC = 10
         IF LAYMATCL < 5 .OR. LAYMATCL > 7
            DISPLAY FIELDS SID_NO, STRUCNUM, LAYNUM, LAYDESC, LAYMATCL OFF
            M_ERRORCK = M_ERRORCK + 1
         ENDIF
      CASE LAYDESC = 12
         IF LAYMATCL < 12 .OR. LAYMATCL > 15
            DISPLAY FIELDS SID_NO, STRUCNUM, LAYNUM, LAYDESC, LAYMATCL OFF
            M_ERRORCK = M_ERRORCK + 1
         ENDIF
      CASE LAYDESC = 14
         IF LAYMATCL <> 27
            DISPLAY FIELDS SID_NO, STRUCNUM, LAYNUM, LAYDESC, LAYMATCL OFF
            M_ERRORCK = M_ERRORCK + 1
         ENDIF
   ENDCASE
   SKIP
ENDDO
CLEAR
? " "
? " "
? " "
? " "
? " "
? " "

*
* VERIFIES THAT THE LAYER NUMBERS ARE IN CONSECUTIVE
* ORDER IN THE TRANSACTION FILE AND THAT NONE ARE MISSING
*
GOTO TOP
M_ERROR  = "LAYER ID FILE - MISSING  OR  INCORRECT  LAYER  NUMBERS"
M_ERROR2 = "-------------------------------------------------"
? M_ERROR
? M_ERROR2
MSID_NO = SID_NO
MLAYNUM = 1
```

```
DO WHILE .NOT. EOF()
    * checks for consecutive sid numbers
   DO WHILE .NOT. EOF() .AND. MSID_NO = SID_NO
    IF LAYNUM <> MLAYNUM
        IF LAYDESC = 13 .OR. LAYDESC = 12
            SKIP
            STORE LAYNUM TO MLAYNUM
            IF MSID_NO <> SID_NO
                SKIP -1
            ENDIF
        ELSE
            DISPLAY SID_NO, STRUCNUM, LAYNUM OFF
            M_ERRORCK = M_ERRORCK + 1
            STORE LAYNUM TO MLAYNUM
        ENDIF
    ENDIF
    MLAYNUM = MLAYNUM + 1
    SKIP
   ENDDO
   MSID_NO = SID_NO
   MLAYNUM = 1
ENDDO
CLEAR
? " "
? " "
? " "
? " "
? " "
? " "

*
* TO LIST OUT OF RANGE DATA IN THE LAYER IDENTIFICATION
* TRANSACTION FILE
*
GOTO TOP
M_ERROR  = "LAYER ID FILE - OUT  OF  RANGE  DATA"
M_ERROR2 = "-----------------------------------"
? M_ERROR
? M_ERROR2

* check the data
DO WHILE .NOT. EOF()
    IF STRUCNUM < 1 .OR. STRUCNUM > 9
       DISPLAY FIELDS SID_NO, STRUCNUM, LAYNUM OFF
       M_ERRORCK = M_ERRORCK + 1
    ENDIF
    IF LAYNUM < 1 .OR. LAYNUM > 13
       DISPLAY FIELDS SID_NO, STRUCNUM, LAYNUM OFF
       M_ERRORCK = M_ERRORCK + 1
    ENDIF
    IF LAYDESC < 1 .OR. LAYDESC > 14
       DISPLAY FIELDS SID_NO, STRUCNUM, LAYNUM, LAYDESC OFF
       M_ERRORCK = M_ERRORCK + 1
    ENDIF
```

```
    IF CENTTHK < .1 .OR. CENTTHK > 25
        IF LAYNUM <> 1 .AND. CENTTHK = 0
            DISPLAY FIELDS SID_NO, STRUCNUM, LAYNUM, CENTTHK OFF
            M_ERRORCK = M_ERRORCK + 1
        ENDIF
    ENDIF
    IF LAYMATCL < 1 .OR. LAYMATCL > 45
       DISPLAY FIELDS SID_NO, STRUCNUM, LAYNUM, LAYMATCL OFF
       M_ERRORCK = M_ERRORCK + 1
    ENDIF
    IF JOBCMPYR < 20 .OR. JOBCMPYR > 99
       DISPLAY FIELDS SID_NO, STRUCNUM, LAYNUM, JOBCMPYR OFF
       M_ERRORCK = M_ERRORCK + 1
    ENDIF
    IF JOBCMPMO < 1 .OR. JOBCMPMO > 12
       DISPLAY FIELDS SID_NO, STRUCNUM, LAYNUM, JOBCMPMO OFF
       M_ERRORCK = M_ERRORCK + 1
    ENDIF
    IF WIDENLYR < 28 .OR. WIDENLYR > 99
       IF WIDENLYR <> 0
           DISPLAY FIELDS SID_NO, STRUCNUM, LAYNUM, WIDENLYR OFF
           M_ERRORCK = M_ERRORCK + 1
       ENDIF
    ENDIF
    IF WIDENLMO < 1 .OR. WIDENLMO > 12
       IF WIDENLMO <> 0
            DISPLAY FIELDS SID_NO, STRUCNUM, LAYNUM, WIDENLMO OFF
            M_ERRORCK = M_ERRORCK + 1
       ENDIF
    ENDIF
    SKIP
ENDDO
CLEAR
? " "
? " "
? " "
? " "
? " "
? " "

*
* LIST OUT SID NUMBERS THAT ARE INCORRECT
*

clear
GOTO TOP
M_ERROR  = "LAYER ID FILE  -  INCORRECT  SID  NUMBERS"
M_ERROR2 = "----------------------------------------"
? M_ERROR
? M_ERROR2

* initialize variables
STORE 0 TO VAR1, VAR2, VAR3, VAR4, COMPARE
```

```
* calculate correct Sid Numnber check digit
DO WHILE  .NOT. EOF()
    VAR1 = INT(SID_NO/1000)
    VAR2 = MOD(INT(SID_NO/100),10)
    VAR2 = VAR2 * 2
    VAR3 = MOD(INT(SID_NO/10),10)
    VAR3 = VAR3 * 3
    VAR4 = MOD(VAR1,10)+VAR2+VAR3
    COMPARE = MOD(VAR4,10)

    * compare actual check digit with calculated check digit
    IF COMPARE <> MOD(SID_NO,10)
        DISPLAY SID_NO, STRUCNUM, LAYNUM OFF
        M_ERRORCK = M_ERRORCK + 1
    ENDIF
    IF SID_NO = 0
        DISPLAY SID_NO, STRUCNUM, LAYNUM OFF
        M_ERRORCK = M_ERRORCK + 1
    ENDIF
    SKIP
ENDDO
? " "
? " "
? " "
? " "
? " "
? " "

*
* LIST OUT LAYER RECORDS THAT DO NOT HAVE A LOCATION RECORD
*
clear
GOTO TOP
SELE 2
USE \PAVEDB\EDITUPDT\LOCNCHNG INDE \PAVEDB\EDITUPDT\LOCNCHNG
M_ERROR  = "LAYER ID FILE - SID NUMBERS THAT DO NOT HAVE A LOCATION RECORD"
M_ERROR2 = "-------------------------------------------------------------"
SELE 1
? M_ERROR
? M_ERROR2
DO WHILE  .NOT. EOF()
    MSID_NO = A->SID_NO
    SELE 2
    SEEK MSID_NO
    IF .NOT. FOUND()
        SELE 1
        DISPLAY SID_NO, STRUCNUM, LAYNUM OFF
        M_ERRORCK = M_ERRORCK + 1
    ENDIF
    SELE 1
    DO WHILE A->SID_NO = MSID_NO .OR. (.NOT. EOF() )
       SKIP
    ENDDO
ENDDO
```

```
sele 1
use
sele 2
use
USE \PAVEDB\EDITUPDT\LAYRCHNG INDEX \PAVEDB\EDITUPDT\LAYRCHNG
PACK
USE
? CHR(12)
SET ALTE OFF
CLOSE ALTE
IF M_ERRORCK <> 0
    M_OK = M_OK + 1
ENDIF
CLEAR
RETURN
```

```
*
* SUBSYSTEM:      EDIT & UPDATE INVENTORY FILES
* PROGRAM NAME:   CHEKLAYT.PRG       06/21/88
* CALLED FROM:    CHNGLAYR.PRG
* PROJECT 2456 - TEXAS FLEXIBLE PAVEMENT DATABASE CONVERSION
* TAMU/TTI
* AUTHOR:         TREVOR X. PEREIRA
* PURPOSE:        TO LIST ERRORS IN THE LAYER THICKNESS ACROSS THE ROAD FILE
*

* set parameters
Set talk oFF
set echo oFF
CLEAR

* print out page heading
SET ALTE TO LAYTERR
SET ALTE ON
MTITLE  = "LAYER THICKNESS FILE - LIST OF RECORDS NOT PRESENT IN LAYER ID FILE"
MTITLE2 = "-------------------------------------------------------------------"
? MTITLE
? MTITLE2
* assign files to different work areas
SELECT 1
   USE \PAVEDB\EDITUPDT\LAYRCHNG INDEX \PAVEDB\EDITUPDT\LAYRCHNG
SELECT 2
   USE \PAVEDB\EDITUPDT\LAYTCHNG INDEX \PAVEDB\EDITUPDT\LAYTCHNG

M_ERROR = 0
DO WHILE .NOT. EOF()
   MLAYSID = SID_NO
   MLAYSTRC = STRUCNUM
   MLAYNUM = LAYNUM

   * find the laythick file record in layer file
   SELECT 1
   seek str(MLAYSID,4)+str(MLAYSTRC,2)+str(MLAYNUM,2)
   IF .NOT. FOUND()
      SELECT 2
      DISPLAY FIELDS SID_NO, STRUCNUM, LAYNUM OFF
      M_ERROR = M_ERROR + 1
   ENDIF
   SELE 2
   SKIP
ENDDO
SELE 1
USE
SELE 2
USE
CLEAR
```

```
? " "
? " "
? " "
? " "
? " "
? " "

*
* TO LIST ALL OUT OF RANGE DATA IN THE LAYER THICKNESS
* ACROSS THE ROAD FILE
*

USE \PAVEDB\EDITUPDT\LAYTCHNG INDEX \PAVEDB\EDITUPDT\LAYTCHNG
GOTO TOP

* print out the page heading
M_TITLE =  "LAYER THICKNESS FILES - OUT OF RANGE RECORDS"
M_TITLE2 = "----------------------------------------"
? M_TITLE
? M_TITLE2

* check the ranges
DO WHILE .NOT. EOF()
    IF STRUCNUM < 1 .OR. STRUCNUM > 9
       DISPLAY FIELDS SID_NO, STRUCNUM, LAYNUM, STRUCNUM OFF
       M_ERROR = M_ERROR + 1
    ENDIF
    IF LAYNUM < 1 .OR. LAYNUM > 13
       DISPLAY FIELDS SID_NO, STRUCNUM, LAYNUM, LAYNUM OFF
       M_ERROR = M_ERROR + 1
    ENDIF
    IF FC3THK < 0 .OR. FC3THK > 25
       DISPLAY FIELDS SID_NO, STRUCNUM, LAYNUM, FC3THK OFF
       M_ERROR = M_ERROR + 1
    ENDIF
    IF FC2THK < 0 .OR. FC2THK > 25
       DISPLAY SID_NO, STRUCNUM, LAYNUM, FC2THK  OFF
       M_ERROR = M_ERROR + 1
    ENDIF
    IF FC1THK < .1 .OR. FC1THK > 25
       DISPLAY SID_NO, STRUCNUM, LAYNUM, FC1THK  OFF
       M_ERROR = M_ERROR + 1
    ENDIF
    IF CENTITHK < .1 .OR. CENTITHK > 25
       DISPLAY SID_NO, STRUCNUM, LAYNUM, CENTITHK  OFF
       M_ERROR = M_ERROR + 1
    ENDIF
    IF FC1DIS < .01 .OR. FC1DIS > 99
       DISPLAY SID_NO, STRUCNUM, LAYNUM, FC1DIS  OFF
       M_ERROR = M_ERROR + 1
    ENDIF
    IF FC2DIS < FC1DIS
       IF FC2DIS <> 0
          DISPLAY SID_NO, STRUCNUM, LAYNUM, FC2DIS  OFF
```

```
            M_ERROR = M_ERROR + 1
         ENDIF
      ENDIF
      IF FC3DIS < FC2DIS
         IF FC3DIS <> 0
            DISPLAY SID_NO, STRUCNUM, LAYNUM, FC3DIS  OFF
            M_ERROR = M_ERROR + 1
         ENDIF
      ENDIF
      SKIP
ENDDO
CLEAR
? " "
? " "
? " "
? " "
? " "
? " "

*
* TO CHECK FOR INVALID SID NUMBERS
*
GOTO TOP
* set up page heading
M_TITTLE1 = "LAYER THICKNESS FILE - INCORRECT SID NUMBERS"
M_TITTLE2 = "----------------------------------------"
? M_TITTLE1
? M_TITTLE2
STORE 0 TO VAR1, VAR2, VAR3, VAR4, COMPARE

* calculate check digit
DO WHILE  .NOT. EOF()
    VAR1 = INT(SID_NO/1000)
    VAR2 = MOD(INT(SID_NO/100),10)
    VAR2 = VAR2 * 2
    VAR3 = MOD(INT(SID_NO/10),10)
    VAR3 = VAR3 * 3
    VAR4 = MOD(VAR1,10)+VAR2+VAR3
    COMPARE = MOD(VAR4,10)

    * compare calculated check digit with actual check digit
    IF COMPARE <> MOD(SID_NO,10)
        DISPLAY SID_NO, STRUCNUM, LAYNUM OFF
        M_ERROR = M_ERROR + 1
    ENDIF
    IF SID_NO = 0
        DISPLAY SID_NO, STRUCNUM, LAYNUM OFF
        M_ERROR = M_ERROR + 1
    ENDIF
    SKIP
ENDDO
* reset parameters
CLEAR
? " "
```

```
? " "
? " "
? " "
? " "
? " "

*
* TO LIST LAYER IDENTIFICATION FILE RECORDS THAT ARE NOT
* PRESENT IN LAYER THICKNESS ACROSS THE ROAD FILE
*

M_TITLE  = "LAYER THICKNESS FILE - LIST OF ALL RECORDS IN MASTER LAYER ID FILE"
M_TITLE2 = "        THAT ARE NOT PRESENT IN THE LAYER THICKNESS FILE"
M_TITLE3 = "------------------------------------------------------------------"
? M_TITLE
? M_TITLE2
? M_TITLE3

* assign files to different work areas
SELECT 1
   USE \PAVEDB\EDITUPDT\LAYRCHNG INDEX \PAVEDB\EDITUPDT\LAYRCHNG
SELECT 2
   USE \PAVEDB\EDITUPDT\LAYTCHNG INDEX \PAVEDB\EDITUPDT\LAYTCHNG

SELECT 1
DO WHILE .NOT. EOF()
   MSID = SID_NO
   MSTR = STRUCNUM
   MLAY = LAYNUM

   * if layer number is 1, skip to the next record in layer file
   IF LAYNUM = 1
      SKIP
      LOOP
   ENDIF

   * find the layer record in laythick file
   SELECT 2
   SEEK STR(MSID,4)+STR(MSTR,2)+STR(MLAY,2)
   IF .NOT. FOUND()
      SELECT 1
      DISPLAY FIELDS SID_NO, STRUCNUM, LAYNUM OFF
      M_ERROR = M_ERROR + 1
   ENDIF
   SELECT 1
   SKIP
ENDDO

CLEAR
? CHR(12)
SET ALTE OFF
SELE 1
use
SELE 2
```

```
USE
USE \PAVEDB\EDITUPDT\LAYTCHNG INDEX \PAVEDB\EDITUPDT\LAYTCHNG
PACK
USE
CLOSE ALTE
IF M_ERROR <> 0
   M_OK = M_OK + 1
ENDIF
RETURN
```

```
*
* SUBSYSTEM:        EDIT & UPDATE
* PROGRAM NAME:     CHEKLOCN.PRG              06/21/88
* CALLED FROM:      CHNGLAYR.PRG
* PROJECT 2456 -    TEXAS FLEXIBLE PAVEMENT DATABASE CONVERSION
* TAMU/TTI
* AUTHOR:           TREVOR X. PEREIRA
* PURPOSE:          CHECKS THE TEMPORARY LOCATION FILE FOR ERRORS AND
*                   THEN MAKES IT THE MASTER LOCATION FILE
*


SET TALK OFF
CLEAR
USE \PAVEDB\EDITUPDT\LOCNCHNG INDE \PAVEDB\EDITUPDT\LOCNCHNG
M_ERROR  = "LOCATION FILE - INCORRECT DATA ENTERED"
M_ERROR2 = "----------------------------------------"
SET ALTE TO LOCNERR
SET ALTE ON
MERRORCNT = 0
? M_ERROR
? M_ERROR2
DO WHILE .NOT. EOF()
   MSID_NO = SID_NO
   * calculates the correct Sid Number
   STORE 0 TO VAR1, VAR2, VAR3, COMPARE
   VAR1 = INT(MSID_NO/1000)
   VAR2 = MOD(INT(MSID_NO/100),10)
   VAR2 = VAR2 * 2
   VAR3 = MOD(INT(MSID_NO/10),10)
   VAR3 = VAR3 * 3
   VAR4 = MOD(VAR1,10)+VAR2+VAR3
   COMPARE = MOD(VAR4,10)

   * compares calculated Sid number with Sid number entered
   IF COMPARE <> MOD(MSID_NO,10)
     MERRORCNT = MERRORCNT + 1
     DISPLAY FIELD SID_NO OFF
     ? "SID NUMBER IS NOT VALID"
     ? " "
     ? " "
   ENDIF
   * verifies that Sid number has been entered
   IF MSID_NO = 0
     MERRORCNT = MERRORCNT + 1
     DISPLAY FIELD SID_NO OFF
     ? "     SID NUMBER IS NOT VALID"
     ? " "
     ? " "
   ENDIF
```

```
IF HWYDIST < 0 .OR. HWYDIST > 27
    DISPLAY FIELDS SID_NO, HWYDIST  OFF
    MERRORCNT = MERRORCNT + 1
ENDIF
IF CNTYNUM < 0 .OR. CNTYNUM > 270
    DISPLAY FIELDS SID_NO, CNTYNUM OFF
    MERRORCNT = MERRORCNT + 1
ENDIF
DO CASE
  CASE HWYPREFX = "FM"
  CASE HWYPREFX = "SH"
  CASE HWYPREFX = "IH"
  CASE HWYPREFX = "US"
  CASE HWYPREFX = "SP"
  CASE HWYPREFX = "LP"
  CASE HWYPREFX = "PR"
  OTHERWISE
    DISPLAY FIELDS SID_NO, HWYPREFX OFF
    MERRORCNT = MERRORCNT + 1
ENDCASE
IF HWYNUM = 0
    DISPLAY FIELDS SID_NO, HWYNUM OFF
    MERRORCNT = MERRORCNT + 1
ENDIF
IF BDISSIGN = "+" .OR. BDISSIGN = "-"
ELSE
    DISPLAY FIELDS SID_NO, BDISSIGN OFF
    MERRORCNT = MERRORCNT + 1
ENDIF
IF EDISSIGN = "+" .OR. EDISSIGN = "-"
ELSE
    DISPLAY FIELDS SID_NO, EDISSIGN OFF
    MERRORCNT = MERRORCNT + 1
ENDIF
IF LANEID = "R" .OR. LANEID = "L"
ELSE
    DISPLAY FIELDS SID_NO, LANEID OFF
    MERRORCNT = MERRORCNT + 1
ENDIF
IF BEGMPNT = 0 .AND. ENDMPNT = 0
    DISPLAY FIELDS SID_NO, BEGMPNT, ENDMPNT OFF
    MERRORCNT = MERRORCNT + 1
ENDIF
IF MPNTMO < 0 .OR. MPNTMO > 12
    DISPLAY FIELDS SID_NO, MPNTMO OFF
    MERRORCNT = MERRORCNT + 1
ENDIF
IF NUMLANES < 1 .OR. NUMLANES > 22
    DISPLAY FIELDS SID_NO, NUMLANES OFF
    MERRORCNT = MERRORCNT + 1
ENDIF
IF PREVSID <> 0
  MPREVSID = PREVSID
  MBEGMPST = BEGMPST
```

```
      MBDISSIGN = BDISSIGN
      MBMPSTDIS = BMPSTDIS
      MENDMPST = ENDMPST
      MEDISSIGN = EDISSIGN
      MEMPSTDIS = EMPSTDIS
      SEEK MPREVSID
      IF .NOT. FOUND()
        SEEK MSID_NO
        DISPLAY FIELD SID_NO, PREVSID OFF
        ? "   SID Number NOT present in location file"
        ? " "
        ? " "
        MERRORCNT = MERRORCNT + 1
      ELSE
        IF MBEGMPST <> BEGMPST .OR. MBDISSIGN <> BDISSIGN .OR. MBMPSTDIS <>
BMPSTDIS
          SEEK MSID_NO
          DISPLAY FIELD SID_NO, PREVSID OFF
          ? "   SID Number and PREVIOUS SID Beginning Mile Post do not match"
          ? " "
          ? " "
          MERRORCNT = MERRORCNT + 1
        ENDIF
        SEEK MPREVSID
        IF MENDMPST <> ENDMPST .OR. MEDISSIGN <> EDISSIGN .OR. MEMPSTDIS <>
EMPSTDIS
          SEEK MSID_NO
          DISPLAY FIELD SID_NO, PREVSID OFF
          ? "   SID Number and PREVIOUS SID Beginning Mile Post do not match"
          ? " "
          ? " "
          MERRORCNT = MERRORCNT + 1
        ENDIF
      ENDIF
      SEEK MSID_NO
    ENDIF
    IF NEXTSID <> 0
      MNEXTSID = NEXTSID
      SEEK MNEXTSID
      IF .NOT. FOUND()
        SEEK MSID_NO
        DISPLAY FIELD SID_NO, NEXTSID OFF
        ? "   SID Number NOT present in location file"
        ? " "
        ? " "
        MERRORCNT = MERRORCNT + 1
      ENDIF
      SEEK MSID_NO
    ENDIF
    IF CONTROL = 0
        DISPLAY FIELDS SID_NO, CONTROL OFF
        MERRORCNT = MERRORCNT + 1
    ENDIF
    IF SECTION = 0
```

```
        DISPLAY FIELDS SID_NO, SECTION OFF
        MERRORCNT = MERRORCNT + 1
    ENDIF
    IF FUNCLAS < 0 .OR. FUNCLAS > 7
        DISPLAY FIELDS SID_NO, FUNCLAS OFF
        MERRORCNT = MERRORCNT + 1
    ENDIF
    IF ENDMPST = 0
        DISPLAY FIELDS SID_NO, ENDMPST OFF
        MERRORCNT = MERRORCNT + 1
    ENDIF
    IF EMPSTDIS < 0
        DISPLAY FIELDS SID_NO, EMPSTDIS OFF
        MERRORCNT = MERRORCNT + 1
    ENDIF
    IF BMPSTDIS < 0
        DISPLAY FIELDS SID_NO, BMPSTDIS OFF
        MERRORCNT = MERRORCNT + 1
    ENDIF
    IF MPNTYR < 20 .OR. MPNTYR > 99
        DISPLAY FIELDS SID_NO, HWYNUM OFF
        MERRORCNT = MERRORCNT + 1
    ENDIF
    IF .NOT. ACTVFLAG
        IF INACTYR < 50 .OR. INACTYR > 99
           DISPLAY FIELDS SID_NO, INACTYR OFF
           MERRORCNT = MERRORCNT + 1
        ENDIF
        IF INACIMO < 0 .OR. INACIMO > 12
           DISPLAY FIELDS SID_NO, INACIMO OFF
           MERRORCNT = MERRORCNT + 1
        ENDIF
    ENDIF
    IF ACTVFLAG
        IF INACTYR <> 0
           DISPLAY FIELDS SID_NO, INACTYR OFF
           ? "    Inactive Flag and Inactive YEAR are inconsistent"
           ? " "
           ? " "
           MERRORCNT = MERRORCNT + 1
        ENDIF
        IF INACIMO <> 0
           DISPLAY FIELDS SID_NO, INACIMO OFF
           ? "    Inactive Flag and Inactive MONTH are inconsistent"
           ? " "
           ? " "
           MERRORCNT = MERRORCNT + 1
        ENDIF
        IF NEXTSID <> 0
           DISPLAY FIELDS SID_NO, NEXTSID OFF
           ? "    Cannot point to Another SID number when present SID number is
active"
           ? " "
           ? " "
```

```
            MERRORCNT = MERRORCNT + 1
         ENDIF
      ENDIF
      SKIP
ENDDO
use
USE \PAVEDB\EDITUPDT\LOCNCHNG INDE \PAVEDB\EDITUPDT\LOCNCHNG
PACK
USE
* reset parameters
CLEAR
? CHR(12)
SET ALITE OFF
CLOSE ALITE
IF MERRORCNT <> 0
   M_OK = M_OK + 1
ENDIF
RETURN
```

```
*
* SUBSYSTEM:      EDIT & UPDATE INVENTORY FILES
* PROGRAM NAME:   CHEKSUBG.PRG      06/21/88
* CALLED FROM:    CHNGLAYR.PRG
* PROJECT 2456 - TEXAS FLEXIBLE PAVEMENT DATABASE CONVERSION
* TAMU/TTI
* AUTHOR:         TREVOR X. PEREIRA
* PURPOSE:        TO LIST THE SUBGRADE FILE RECORDS THAT HAVE ERRORS
*

*
* TO LIST THE SUBGRADE FILE RECORDS THAT ARE NOT PRESENT
* IN THE MASTER LAYER IDENTIFICATION FILE
*
* set parameters
set talk off
set echo off
SET ALTE TO SUBGERR
CLEAR
M_ERROR = 0

M_TITLE  = " SUBGRADE FILE - LIST OF RECORDS NOT PRESENT IN LAYER ID FILE"
M_TITLE2 = "------------------------------------------------------------"
SET ALTE ON
? M_TITLE
? M_TITLE2

* assign files to different work areas
SELECT 1
   USE \PAVEDB\EDITUPDT\LAYRCHNG INDEX \PAVEDB\EDITUPDT\LAYRCHNG

SELECT 2
   USE \PAVEDB\EDITUPDT\SUBGCHNG INDEX \PAVEDB\EDITUPDT\SUBGCHNG

DO WHILE .NOT. EOF()
   MSUBGSID = SID_NO
   MSUBGLAY = " 1"
   MSUBGSTR = " 1"

   * find the subgrade sid number in layer identification file
   SELECT 1
   seek str(MSUBGSID,4)+MSUBGLAY+MSUBGSTR
   IF .NOT. FOUND()
      SELECT 2
      DISPLAY FIELD SID_NO, STRUCNUM, LAYNUM OFF
      M_ERROR = M_ERROR + 1
   ENDIF
   SELECT 2
   SKIP
ENDDO
```

```
SELE 1
USE
SELE 2
USE
? " "
? " "
? " "
? " "
? " "
? " "

*
* TO LIST OUT ALL DATA THAT IS OUT OF RANGE IN THE SUBGRADE FILE
*

* print out page heading
M_TITLE  = "SUBGRADE FILE - OUT OF RANGE RECORDS"
M_TITLE2 = "-----------------------------------"
? M_TITLE
? M_TITLE2

USE \PAVEDB\EDITUPDT\SUBGCHNG INDEX \PAVEDB\EDITUPDT\SUBGCHNG

* do a range check for all fields
DO WHILE .NOT. EOF()
    IF PLASTIX < 0 .OR. PLASTIX > 70
       IF PLASTIX <> 0
           DISPLAY FIELDS SID_NO, STRUCNUM, LAYNUM, PLASTIX OFF
           M_ERROR = M_ERROR + 1
       ENDIF
    ENDIF
    IF LIQLIM < 10 .OR. LIQLIM > 80
       IF LIQLIM <> 0
           DISPLAY FIELDS SID_NO, STRUCNUM, LAYNUM, LIQLIM OFF
           M_ERROR = M_ERROR + 1
       ENDIF
    ENDIF
    IF TXTRIAXL < 1 .OR. TXTRIAXL > 6
       IF TXTRIAXL <> 0
            DISPLAY FIELDS SID_NO, STRUCNUM, LAYNUM, TXTRIAXL OFF
            M_ERROR = M_ERROR + 1
       ENDIF
    ENDIF
    IF PERMIX < 0 .OR. PERMIX > 10
       DISPLAY SID_NO, STRUCNUM, LAYNUM, PERMIX OFF
       M_ERROR = M_ERROR + 1
    ENDIF
    SKIP
ENDDO
CLEAR
? " "
? " "
? " "
? " "
```

```
? " "
? " "

*
* LIST OF INCORRECT SID NUMBERS FOR THE SUBGRADE FILE
*
M_TITLE  = "SUBGRADE FILE - INCORRECT SID NUMBERS"
M_TITLE2 = "------------------------------------"
? M_TITLE
? M_TITLE2

USE \PAVEDB\EDITUPDT\SUBGCHNG INDEX \PAVEDB\EDITUPDT\SUBGCHNG

* initialize variables
STORE 0 TO VAR1, VAR2, VAR3, VAR4, COMPARE

* calculate correct Sid number check digit
DO WHILE  .NOT. EOF()
    VAR1 = INT(SID_NO/1000)
    VAR2 = MOD(INT(SID_NO/100),10)
    VAR2 = VAR2 * 2
    VAR3 = MOD(INT(SID_NO/10),10)
    VAR3 = VAR3 * 3
    VAR4 = MOD(VAR1,10)+VAR2+VAR3
    COMPARE = MOD(VAR4,10)

    * compare calculated check digit with actual check digit
    IF COMPARE <> MOD(SID_NO,10)
        DISPLAY SID_NO, STRUCNUM, LAYNUM OFF
        M_ERROR = M_ERROR + 1
    ENDIF
    IF SID_NO = 0
        DISPLAY SID_NO, STRUCNUM, LAYNUM OFF
        M_ERROR = M_ERROR + 1
    ENDIF
    SKIP
ENDDO
USE
* reset parameters
CLEAR
? " "
? " "
? " "
? " "
? " "
? " "

*
* TO LIST ALL LAYER IDENTIFICATION RECORDS NOT PRESENT IN THE SUBGRADE FILE
*
CLEAR
* print out the page heading
M_TITLE  = "SUBGRADE FILE - LIST OF ALL LAYER ID RECORDS NOT"
M_TITLE2 = "       PRESENT IN THE SUBGRADE FILE"
```

```
M_TITLE3 = "-----------------------------------------------"
? M_TITLE
? M_TITLE2
? M_TITLE3

* assign files to different work areas
SELECT 1
   USE \PAVEDB\EDITUPDT\LAYRCHNG INDEX \PAVEDB\EDITUPDT\LAYRCHNG
SELECT 2
   USE \PAVEDB\EDITUPDT\SUBGCHNG INDEX \PAVEDB\EDITUPDT\SUBGCHNG

SELECT 1
MSID = SID_NO
MSTRUC = " 1"
MLAYNUM = " 1"
DO WHILE .NOT. EOF()
   SELECT 2
   seek str(MSID,4)+MSTRUC+MLAYNUM
   IF .NOT. FOUND()
      SELECT 1
      DISPLAY FIELD SID_NO, STRUCNUM, LAYNUM OFF
      M_ERROR = M_ERROR + 1
   ENDIF
   SELECT 1
   DO WHILE .NOT. EOF() .AND. SID_NO = MSID
     SKIP
   ENDDO
   MSID = SID_NO
ENDDO
SELE 1
use
SELE 2
USE
USE \PAVEDB\EDITUPDT\SUBGCHNG INDEX \PAVEDB\EDITUPDT\SUBGCHNG
PACK
USE
CLEAR
? CHR(12)
SET ALTE OFF
CLOSE ALTE
IF M_ERROR <> 0
   M_OK = M_OK + 1
ENDIF
RETURN
```

```
*
* SUBSYSTEM:      EDIT & UPDATE INVENTORY FILES
* PROGRAM NAME:   CHEKSURF.PRG       06/23/88
* CALLED FROM:    CHNGLAYR.PRG
* PROJECT 2456 - TEXAS FLEXIBLE PAVEMENT DATABASE CONVERSION
* TAMU/TTI
* AUTHOR:         TREVOR X. PEREIRA
* PURPOSE:        LISTS THE SURFACE FILE RECORDS THAT HAVE ERRORS
*

*
* LISTS THE SURFACE FILE RECORDS THAT ARE NOT PRESENT IN
* THE MASTER LAYER IDENTIFICATION FILE
*
* set parameters
set talk oFF
set echo oFF
SET ALTE TO SURFERR
CLEAR
M_ERROR = 0

M_TITLE  = "SURFACE FILE - LIST OF RECORDS NOT PRESENT IN LAYER ID FILE"
M_TITLE2 = "-----------------------------------------------------"
SET ALTE ON
? M_TITLE
? M_TITLE2

* assign files to different work areas
SELECT 1
   USE \PAVEDB\EDITUPDT\LAYRCHNG INDEX \PAVEDB\EDITUPDT\LAYRCHNG

SELECT 2
   USE \PAVEDB\EDITUPDT\SURFCHNG INDEX \PAVEDB\EDITUPDT\SURFCHNG

DO WHILE .NOT. EOF()
   * store surface record in memory
   MSURFSID = SID_NO
   MSURFSTR = STRUCNUM
   MSURFLAY = LAYNUM

   * find the layer ID record that matches surface record
   SELECT 1
   SEEK STR(MSURFSID,4)+STR(MSURFSTR,2)+STR(MSURFLAY,2)
   IF .NOT. FOUND()
      SELE 2
      DISPLAY FIELDS SID_NO, STRUCNUM, LAYNUM OFF
      M_ERROR = M_ERROR + 1
   ELSE
      SELE 2
   ENDIF
```

```
   SKIP
ENDDO
SELE 1
USE
SELE 2
USE
? " "
? " "
? " "
? " "
? " "
? " "

*
* TO LIST ALL OUT OF RANGE DATA IN THE SURFACE FILE
*

* print out page headings
M_TITLE  = "SURFACE FILE - OUT OF RANGE RECORDS"
M_TITLE2 = "-----------------------------------"
? M_TITLE
? M_TITLE2

USE \PAVEDB\EDITUPDT\SURFCHNG INDEX \PAVEDB\EDITUPDT\SURFCHNG

* check data with ranges
DO WHILE .NOT. EOF()
    IF STRUCNUM < 1 .OR. STRUCNUM > 9
       DISPLAY FIELDS SID_NO, STRUCNUM, LAYNUM, STRUCNUM OFF
       M_ERROR = M_ERROR + 1
    ENDIF
    IF LAYNUM < 1 .OR. LAYNUM > 13
       DISPLAY FIELDS SID_NO, STRUCNUM, LAYNUM, LAYNUM OFF
       M_ERROR = M_ERROR + 1
    ENDIF
    IF AGAPPLRT < 30 .OR. AGAPPLRT > 200
       IF AGAPPLRT <> 0
           DISPLAY SID_NO, STRUCNUM, LAYNUM, AGAPPLRT OFF
           M_ERROR = M_ERROR + 1
       ENDIF
    ENDIF
    IF ADMXPER < 0 .OR. ADMXPER > 10.2
           DISPLAY SID_NO, STRUCNUM, LAYNUM, ADMXPER OFF
           M_ERROR = M_ERROR + 1
    ENDIF
    IF ASAPPLRT < .1 .OR. ASAPPLRT > .6
       IF ASAPPLRT <> 0
           DISPLAY SID_NO, STRUCNUM, LAYNUM, ASAPPLRT OFF
           M_ERROR = M_ERROR + 1
       ENDIF
    ENDIF
    SKIP
ENDDO
? " "
```

```
? " "
? " "
? " "
? " "
? " "

*
* TO LIST ALL INCORRECT SID NUMBERS FOR SURFACE FILE
*
CLEAR
* print out page heading
M_TITLE  = "SURFACE FILE - INCORRECT SID NUMBERS"
M_TITLE2 = "-----------------------------------"
? M_TITLE
? M_TITLE2

* initialize variables
STORE 0 TO VAR1, VAR2, VAR3, VAR4, COMPARE

* calculate check digit of Sid number
DO WHILE  .NOT. EOF()
    VAR1 = INT(SID_NO/1000)
    VAR2 = MOD(INT(SID_NO/100),10)
    VAR2 = VAR2 * 2
    VAR3 = MOD(INT(SID_NO/10),10)
    VAR3 = VAR3 * 3
    VAR4 = MOD(VAR1,10)+VAR2+VAR3
    COMPARE = MOD(VAR4,10)

    * compare calculated check digit with actual check digit
    IF COMPARE <> MOD(SID_NO,10)
        DISPLAY SID_NO, STRUCNUM, LAYNUM OFF
        M_ERROR = M_ERROR + 1
    ENDIF
    IF SID_NO = 0
        DISPLAY SID_NO, STRUCNUM, LAYNUM OFF
        M_ERROR = M_ERROR + 1
    ENDIF
    SKIP
ENDDO
USE
? " "
? " "
? " "
? " "
? " "
? " "

*
* TO LIST ALL LAYER IDENTIFICATION RECORDS THAT ARE NOT
* PRESENT IN THE SURFACE FILE
*
CLEAR
* print out page heading
```

```
M_TITLE  = "SURFACE FILE - LIST OF ALL LAYER ID RECORDS NOT"
M_TITLE2 = "       PRESENT IN THE SURFACE FILE"
M_TITLE3 = "-----------------------------------------------"
? M_TITLE
? M_TITLE2
? M_TITLE3

* assign files to different work areas
SELECT 1
   USE \PAVEDB\EDITUPDT\LAYRCHNG INDEX \PAVEDB\EDITUPDT\LAYRCHNG
SELECT 2
   USE \PAVEDB\EDITUPDT\SURFCHNG INDEX \PAVEDB\EDITUPDT\SURFCHNG

SELECT 1
DO WHILE .NOT. EOF()
   STORE .T. TO GOTIT

   * find the surface layers in Layer ID file
   DO WHILE GOTIT
      MSID = SID_NO
      MSTR = STRUCNUM
      MLAY = LAYNUM
      IF LAYMATCL < 17
         GOTIT = .F.
         MDESC = LAYMATCL
      ELSE
         SKIP
      ENDIF
   ENDDO

   * find the Layer ID record in Surface file
   SELECT 2
   SEEK STR(MSID,4)+STR(MSTR,2)+STR(MLAY,2)
   IF .NOT. FOUND()
      SELECT 1
      DISPLAY FIELDS SID_NO, STRUCNUM, LAYNUM OFF
      M_ERROR = M_ERROR + 1
   ENDIF
   SELECT 1
   SKIP
ENDDO
SELE 1
use
SELE 2
USE
USE \PAVEDB\EDITUPDT\SURFCHNG INDEX \PAVEDB\EDITUPDT\SURFCHNG
PACK
USE
CLEAR
? CHR(12)
SET ALTE OFF
CLOSE ALTE
IF M_ERROR <> 0
   M_OK = M_OK + 1
```

```
ENDIF
RETURN
```

Section 4: Traffic Data Update

Traffic Tape to Disk

PROGRAM NARRATIVE

This program is for use with a 6250 BPI Cipher GRC CacheTape tape drive system manufactured by Overland Data Incorporated (ODI). It utilizes the ODI DEPOT2 program to copy a tape file to a disk file. The DEPOT2 program is documented in Chapter 4 of the ODI manual. STLOG.PAS (a subprogram of TRAFUPD.PRG) uses the disk file as an input file. The diskfile is stored in the subdirectory \PAVEDB\FILES.

PROGRAM SPECIFICATION

PROGRAM NAME: READTRAF.BAT

PURPOSE: To copy the Traffic data from a tape to a disk file using the ODI DEPOT2 program.

INPUT FILE: The annual Roadway Inventory Tape File (RIFILE) obtained from the Texas State Department of Highways and Public Transportation.

OUTPUT FILE: TLOG.DAT - Annual Roadway Inventory Disk File.

PROGRAMS CALLED:

DEPOT2
READTRAF.CMD - command file for DEPOT2

PROGRAM LISTING

```
rem readtraf.bat issues the depot2 command which calls
rem the cmd file to read the annual Roadway Inventory tape obtained
rem from SDHPT
DEPOT2 /c READTRAF.CMD
```

PROGRAM LISTING

```
; READTRAF.CMD, a command file for the ODI program DEPOT2.
; Use as:  DEPOT2 /c READTRAF.CMD
; Instuctions on making out this command file are in Chapter 4
;    of the ODI manual pages 4-26 to 4-28.
; Change D:\PAVEDB\FILES\TLOG.DAT to the name of your output disk file,
;    if needed.
; Change "/r nn" record length if needed.
; Change "/s nnnn" blocksize if needed.
; "/m 1" skips past the tape label.
; "/tvM" translates from EBCDIC, Verbose explanations, reads till End-
;    Of-File mark at the end of the data.
/n D:\PAVEDB\FILES\TLOG.DAT /r 263 /s 263 /m 1 /tvM
```

Traffic Update

PROGRAM NARRATIVE

The traffic update program allows the user to update the Traffic File in the Texas Flexible Pavement Database. The source of new data for this process is the annual Roadway Inventory Data File (RIFILE) which is produced by D-10 of the Texas State Department of Highways and Public Transportation. The RIFILE data is copied from a tape to a disk file by a stand alone batch program (READTRAF.BAT). The program will extract relevant traffic information from the RIFILE file and through some intermediate data processing update the Traffic Database. Due to the normally large amount of data, this update procedure is time-consuming. It would thus make sense to perform it only once a year. The traffic update programs are stored in the subdirector \PAVEDB\EDITUPDT\TRAFFIC. Other than the master traffic file, the files used by traffic update are stored in \PAVEDB\EDITUPDT\TRAFFIC.

TRAFFIC UPDATE - PROGRAM FLOW DIAGRAM

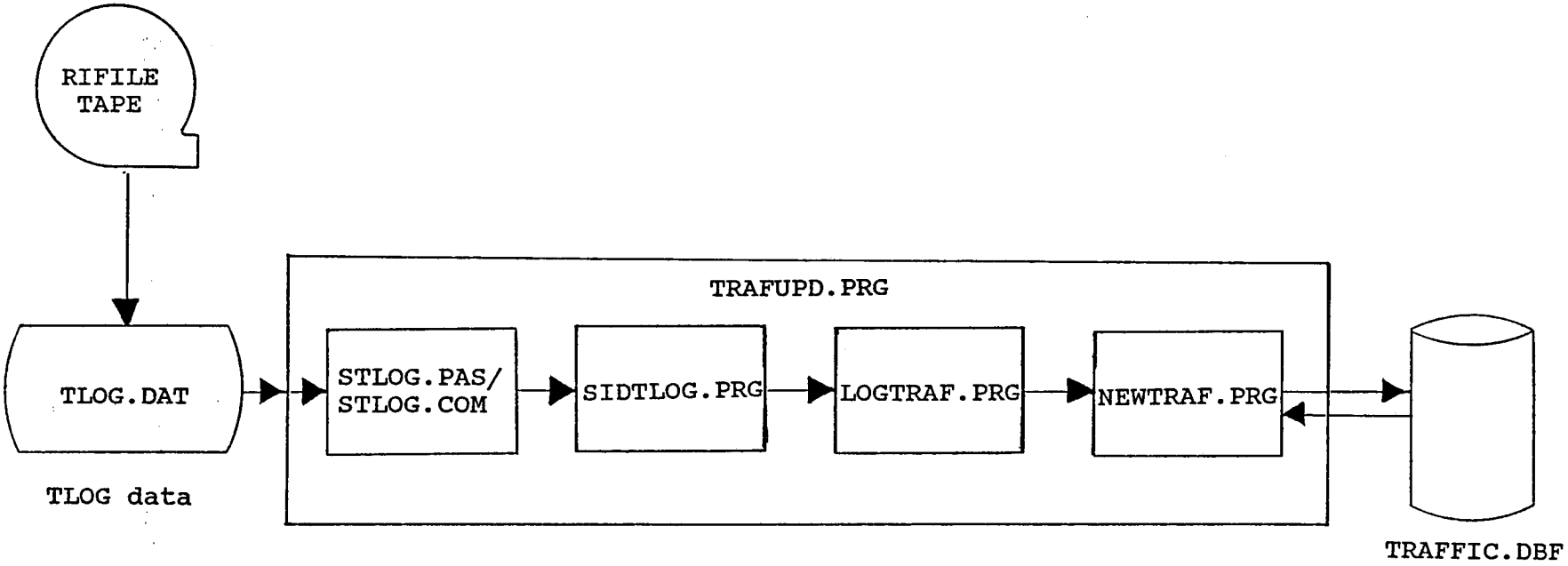


FIGURE 19

PROGRAM SPECIFICATION

PROGRAM NAME: TRAFUPD.PRG

PURPOSE: To use annual Roadway Information File (RIFILE) data to update the Texas Flexible Pavement Database (TRAFFIC.DBF) Traffic File.

PROCEDURES/EDITS:

1. The traffic data is received on a tape. It is copied _from the tape to a disk file and converted from EBCDIC to ASCII format. The data is stored as a record per line in the new disk file. This process is performed by READTRAF.BAT and is not part of the menu system.

2. The Traffic File contains yearly information for each section identification number. The fields include annual average daily traffic (one-way), annual 18KIP, and percentage of trucks.

INPUT FILES: 1. Annual Roadway Inventory Disk File (TLOGxx.DAT where xx stands for the last 2 digits of the RIFILE year)

I/O FILES: 1. Traffic database file (TRAFFIC.DBF)

PROGRAMS CALLED:

STLOG.PAS
SIDTLOG.PRG
LOGTRAF.PRG
NEWTRAF.PRG

NOTE:

1) Where TLOG is used in the following documentation it is referring to the RIFILE.
2) Program Flow Diagrams, Program Specifications, related file layouts, and the program listings are provided in the remainder of this section for each of the called programs.

Traffic Update
TLOG File Record Layout

Roadway Information (RIFILE) File Layout

LABEL	TYPE	FIELD LENGTH
SUB FILE NUMBER	CHAR	1
FILE NUMBER	CHAR	1
THD DISTRICT	NUM	2
THD COUNTY	NUM	3
SHD CONTROL NUMBER	CHAR	4
SHD SECTION NUMBER	CHAR	2
SHD CONTROL/SECTION NUMBER	CHAR	6
BEGINNING MILEPOINT	NUM	5.3
ENDING MILEPOINT	NUM	5.3
LENGTH OF SECTION	NUM	5.3
DESIGNATED HIGHWAY SYSTEM	CHAR	2
DESIGNATED HIGHWAY NUMBER	CHAR	4
HIGHWAY STATUS	CHAR	1
RIGHT-OF-WAY WIDTH (IN FEET)	CHAR	3
HIGHWAY DESIGN TYPE	CHAR	1
ROAD BED WIDTH (IN FEET)	CHAR	3
BASE TYPE	CHAR	1
SURFACE WIDTH (IN FEET)	CHAR	3
COMBINATION OF TWO OR MORE SURFACE TYPES	CHAR	1
SURFACE TYPE	CHAR	2
NUMBER OF LANES	CHAR	2
SHOULDERS TYPE	CHAR	2
DATE OF LAST MILEPOST CHANGE (YYMM)	CHAR	4
YEAR OF LAST MILEPOST CHANGE	CHAR	2
MONTH OF LAST MILEPOST CHANGE	CHAR	2
ADMINISTRATIVE SYSTEM	CHAR	2
TYPE RESERVATION SECTION LOCATED IN	CHAR	2
DESIGNATED WAY FEDERAL AID CODE	CHAR	2
1982 FUNCTIONAL CLASSIFICATION	CHAR	1
URBAN-RURAL FUNCTIONAL CLASSIFICATION	CHAR	1
CONNECTION LINK CODE (FUNCTIONAL CLASS)	CHAR	1
MAINTENANCE SECTION NUMBER	CHAR	2
MAINTENANCE CLASS CODE	CHAR	2
RESTRICTED LOAD LIMIT	CHAR	2
SPECIAL SYSTEMS	CHAR	2
TRUCKS OR COMMERCIAL VEHICLES	CHAR	1
CITY NUMBER	NUM	5
URBAN AREA NUMBER	NUM	5
HWY PERFORMANCE MONITORING SYST SECTION	CHAR	12
YEAR OF CURRENT AADT	NUM	2
ANNUAL AVG DAILY TRAFFIC, CURRENT YEAR	NUM	6
AADT FOR 1 YEAR PRIOR TO CURRENT YEAR	NUM	6
AADT FOR 2 YEARS PRIOR TO CURRENT YEAR	NUM	6
AADT FOR 3 YEARS PRIOR TO CURRENT YEAR	NUM	6
AADT FOR 4 YEARS PRIOR TO CURRENT YEAR	NUM	6

LABEL	TYPE	FIELD LENGTH
AADT FOR 5 YEARS PRIOR TO CURRENT YEAR	NUM	6
AADT FOR 6 YEARS PRIOR TO CURRENT YEAR	NUM	6
AADT FOR 7 YEARS PRIOR TO CURRENT YEAR	NUM	6
AADT FOR 8 YEARS PRIOR TO CURRENT YEAR	NUM	6
AADT FOR 9 YEARS PRIOR TO CURRENT YEAR	NUM	6
DAILY VEHICLE MILES FOR CURRENT YEAR	NUM	8
DESIGN YEAR	NUM	2
YEARLY ADT INCREASE FACTOR	NUM	4.2
ESTIMATED AADT FOR DESIGN YEAR	NUM	6
ESTIMATED DESIGN YEAR DAILY VEH. MILES	NUM	8
DESIGN HOUR'S % OF THE 24 HR AADT	NUM	3.1
DIRECTIONAL DISTRIBUTION IN DHV	NUM	2
% TRUCKS IN AADT	NUM	3.1
% TRUCKS IN DHV	NUM	3.1
AVG 10 HEAVIEST WHEEL LOADS (IN 100 LBS.)	NUM	3
% TANDEM AXLES IN ATHWLD w FIELD	NUM	2
# OF WHEEL LOADS 1/2 8000 LBS	NUM	5
18K FLEX. PAVEMENT ANALYSIS (IN 1000'S)	NUM	5
18K RIGID PAVEMENT ANALYSIS (IN 1000'S)	NUM	5
DESIGN HOUR VOLUME	NUM	5
CONTROL-SECTION LENGTH (NEAREST .001)	NUM	5.3
PRIORITY HIGHWAY SYSTEM	CHAR	2
PRIORITY HIGHWAY NUMBER	NUM	4
CONTROL-SECTION LENGTH (NEAREST .1)	NUM	3.1
CONTROL-SECTION VEHICLE MILES	NUM	16

```
* File name:     TRAFUPD.PRG
* Written by:    Victor Wong
* Created on:    April 8, 1988
* Last updated:  April 8, 1988
* Purpose:       To use new RI2-TLOG data to update Texas Flexible Pavement
*                Traffic Database.

* warn user of this time-consuming process
CLEAR
RES = " "
@ 10,16 SAY "This process will update the TRAFFIC database by"
@ 11,16 SAY "retrieving new data from RI2-TLOG file.  Because"
@ 12,16 SAY "of the size of the data file,  this process will"
@ 13,16 SAY "take a very long time (at least over night)!!    "
@ 15,16 SAY "Do you still want to proceed? (Y/N) " GET RES
READ

IF RES $ "yY"
   * extract traffic information from RI2-TLOG data file
   RUN \PAVEDB\EDITUPDT\TRAFFIC\STLOG

   * order simplified RI2-TLOG data by section identification number
   DO \PAVEDB\EDITUPDT\TRAFFIC\SIDTLOG

   * calculate the Traffic information from the TLOG data
   DO \PAVEDB\EDITUPDT\TRAFFIC\LOGTRAF

   * update the old Traffic database by adding TLOG Traffic data
   DO \PAVEDB\EDITUPDT\TRAFFIC\NEWTRAF
ENDIF

RETURN
```

SUBPROGRAM STLOG.PAS - PROGRAM FLOW DIAGRAM

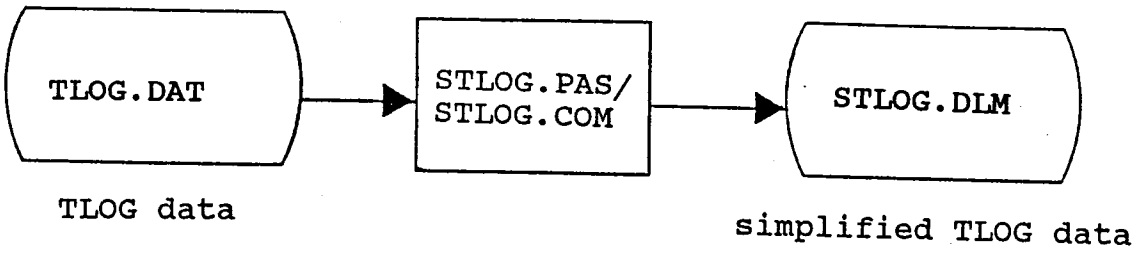


FIGURE 20

Traffic Update
Subprogram 1: STLOG

PROGRAM SPECIFICATIONS

Program Name: STLOG.PAS

Purpose: To simplify the TLOG data by extracting only fields that are required to calculate the values for the Traffic Database File.

Procedures/Edit:

1. The subprogram produces a simlified TLOG data file (STLOG.DLM) from the complete data set (TLOGxx.DAT). The extracted data is stored in a blank-delimited format.

Input Files(s):

1. TLOG data file (TLOG.DAT).

Output Files(s):

1. Simplified TLOG data file (STLOG.DLM).

Traffic Update
Subprogram 1: STLOG

STLOG Delimited File Record Layout

File Name: STLOG.DLM

Description	Size/Type
Control Section number	4N
Beginning Mile Point	6.3N
Ending Mile Point	6.3N
Year of Current AADT	2N
Annual Average Daily Traffic, Current Year (AADT)	6N
AADT Increase Factor (x 0.01)	4N
Estimated AADT for Design Year	6N
Percent Trucks in AADT (x 0.1)	3N
Estimated 18K Flexible Pavement (x 1000)	5N

PROGRAM LISTING

```
{* File name:       STLOG.PAS
 * Program name:    stlog
 * Project 2456:    Texas Flexible Pavement Database Conversion
 * TAMU/TTI
 * Written by:      Victor Wong
 * Purpose:         This program simplifies the data from TLOG
 *                  file.  The only data extracted is  Control
 *                  Section number, Beginning Mile Point (BMP),
 *                  Ending Mile Point (EMP), Current year, Annual
 *                  Average Daily Traffic data (AADT), Design
 *                  year, Average Daily Traffic Increase Factor
 *                  (ADT_INCR_FACTOR), Estimated Average Daily
 *                  Traffic (EST_ADT), Percentage of traffic is
 *                  Trucks (PERCENT_TRUCK), and Estimated 18KIP
 *                  (EST_18K).
 * Input File :     TLOG text data file
 * Output File:     STLOG.DEL
 * NOTE:  This procedure only works for TLOG data file AFTER 1985.
}

{>>> PROGRAM STLOG <<<}
program stlog;

   {>>> variables declarations <<<}
   var
      in_file, out_file: text;       {input & output files}
      skip_junk_1      : string[7];
      cntl_section_no  : string[6];  {control section number}
      bmp_1, emp_1     : string[2];  {beginning & ending    }
      bmp_2, emp_2     : string[3];  {mile points           }
      skip_junk_2      : string[96];
      current_yr       : string[2];  {current yr of adt data}
                                     {adt data now & 9 yrs back}
      aadt             : array [1..10] of string[6];
      skip_junk_3      : string[8];
      design_yr        : string[2];  {design year}
      adt_incr_factor  : string[4];  {act increment factor}
      est_adt          : string[6];  {estimsted adt}
      skip_junk_4      : string[13];
      percent_truck    : string[3];  {percent truck}
      skip_junk_5      : string[13];
      est_18K          : string[5];  {estimated 18k}
      count            : integer;
      ok:              boolean;
      krec,
      rec:             integer;


   {>>> beginning of STLOG.PAS <<<}
```

```
begin
   Clrscr;
   gotoxy (20,10);
   write ('<< STLOG running ... >>');
   gotoxy (20,11);
   write ('   Simplifying the TLOG data.');

   assign (in_file, '\pavedb\tlog.dat');
   reset (in_file);

   assign (out_file, '\pavedb\editupdt\traffic\stlog.dlm');
   rewrite (out_file);

   krec:= 0;
   rec:= 0;

   gotoxy (20,15);
   write ('Extracting data from TLOG data file...');
   gotoxy (20,16);
   write (krec:3,',',rec:3,' records extracted');

   {--- do record by record until end of file for in_file---}
   while not eof (in_file) do
      begin
         {read in control section number}
         read (in_file, skip_junk_1,cntl_section_no);

         {read the rest of the info   }
         {if cnl_section_no is present}
         if (cntl_section_no <> '000000')
            and (cntl_section_no <> '     0')
            and (cntl_section_no <> '      ')
         then
            begin
               read (in_file, bmp_1,bmp_2);
               read (in_file, emp_1, emp_2);
               read (in_file, skip_junk_2, current_yr);

               {if current year is non-zero then continue}
               if current_yr <> '00' then
                  begin
                     for count:= 1 to 10 do
                        read (in_file, aadt[count]);
                     read (in_file, skip_junk_3, design_yr);
                     read (in_file, adt_incr_factor, est_adt);
                     read (in_file, skip_junk_4, percent_truck);
                     readln (in_file, skip_junk_5, est_18K);

                     {write out info}
                     write (out_file, cntl_section_no,' ',bmp_1,'.');
                     write (out_file, bmp_2,' ',emp_1,'.',emp_2);
                     if (design_yr <= '99') and (design_yr > '80') then
                        write (out_file, ' 19',design_yr)
                     else
```

```
                        write (out_file, ' 20',design_yr);
                     if (current_yr <= '99') and (current_yr > '80') then
                        write (out_file, ' 19',current_yr)
                     else
                        write (out_file, ' 20',current_yr);
                     write (out_file, ' ',aadt[1]);
                     write (out_file, ' ',adt_incr_factor,' ',est_adt);
                     writeln (out_file, ' ',percent_truck,' ',est_18K);

                     if (rec < 999) then
                        begin
                           rec:= rec + 1;
                           gotoxy (24,16);
                           write (rec:3);
                        end
                     else
                        begin
                           krec:= krec + 1;
                           gotoxy (20,16);
                           write (krec:3);
                           rec:= 0;
                           gotoxy (24,16);
                           write (rec:3);
                        end;
                  end; {if}
            end; {if}
      end; {while}

   {close input and output files}
   close (in_file);
   close (out_file);

   gotoxy (20,18);
   write ('<< STLOG done. >>');
end. {simplify_tlog_data}
```

SUBPROGRAM SIDTLOG.PRG - PROGRAM FLOW DIAGRAM

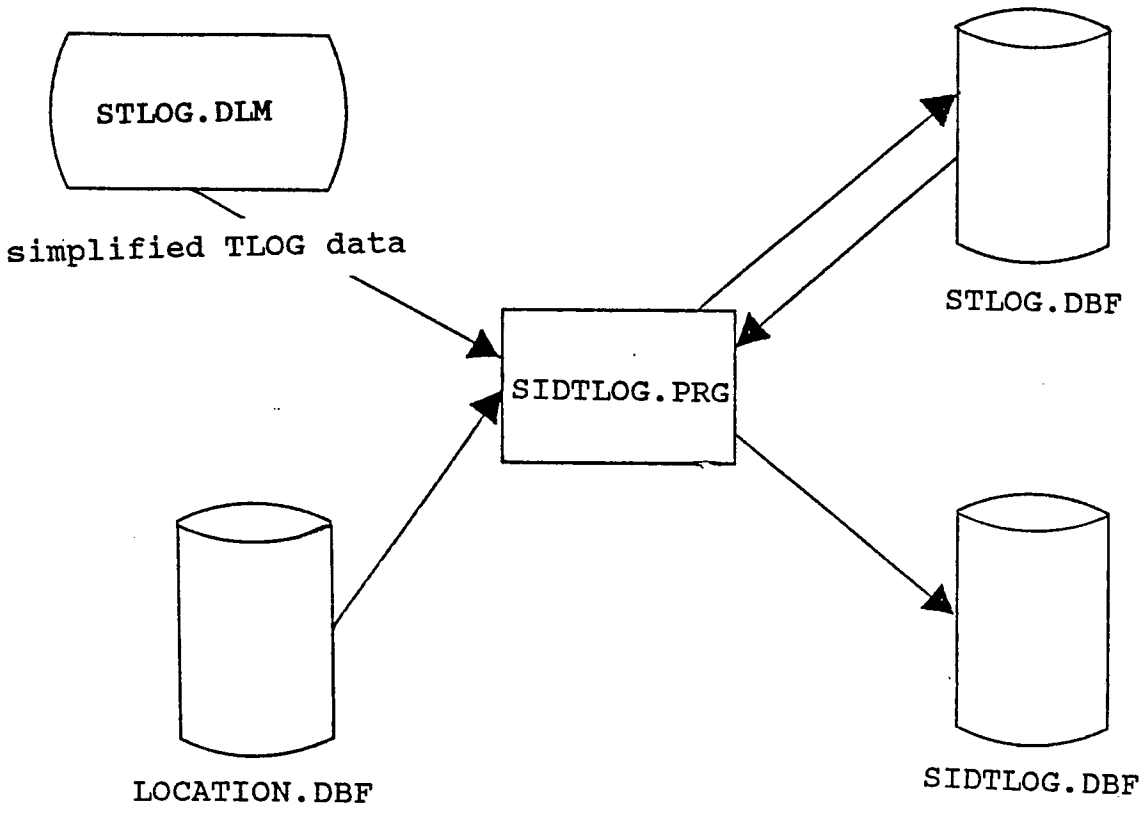


FIGURE 21

Traffic Update
Subprogram 2: SIDTLOG

PROGRAM SPECIFICATIONS

Program Name: SIDTLOG.PRG

Purpose: To use the simplified TLOG data to create a dBASE file containing TLOG data for the section identification numbers found in the Location file.

Procedures/Edits:

1. The TLOG data key is the control section number.
2. The Location database contains information about the control section number and the beginning and ending milepoint of section identification numbers.
3. The program creates a temporary SIDTLOG dBASE file. This file contains the TLOG data for section identification numbers. This is done by joining the simplified TLOG and Location data by the control section number and mile-point range.

Input File(s):
1. Simplified TLOG data file (STLOG.DLM).
2. Location database file (LOCATION.DBF).

I/O File(s):
1. dBASE file for storing the simplified TLOG data (STLOG.DBF).

Output File(s):
1. dBASE file with TLOG data under section identification number (SIDTLOG.DBF)

Traffic Update
Subprogram 2: SIDTLOG

STLOG dBASE File Reocord Layout

File Name: STLOG.DBF

Field	Key	Size/Type	Description
CNTLSEC	*	6N	Control Section Number
TLOGBMP	*	6.3N	Beginning Mile Point
TLOGEMP	*	6.3N	Ending Mile Point
CURYR		4N	Year of Current AADT
ADT		6N	Average Daily Traffic
ADTINCR		4N	ADT Increase Factor
ESTADT		6N	Estimated ADT for Design Year
PCNTRK		3N	Percent Trucks in ADT
EST18K		5N	Estimated 18K Flexible Pavement

Traffic Update
Subprogram 2: SIDTLOG

SIDTLOG dBASE File Record Layout

File Name: SIDTLOG.DBF

Field	Key	Size/Type	Description
SID_NO	*	4N	Section Identification Number
BEGMPNT		6.3N	Beginning Mile Point
ENDMPNT		6.3N	Ending Mile Point
TLOGBMP		6.3N	TLOG Beginning Mile Point
TLOGBMP		6.3N	TLOG Ending Mile Point
CURYR		4N	Year of Current ADT
ADT		6N	Average Daily Traffic
ADTINCR		4N	ADT Increase Factor
ESTADT		6N	Estimated ADT for Design Year
PCNTRK		3N	Percent Trucks in ADT
EST18K		5N	Estimated 18K Flexible Pavement

PROGRAM LISTING

```
* File name:       SIDTLOG.PRG
* Program:         sidtlog
* Project 2456:    Texas Flexible Pavement Database Conversion
* TAMU/TTI
* Written by:      Victor Wong
* Created on:      April 13, 1988
* Last updated:    August 8, 1988
* Purpose:         DBASE III+ program to create the sidtlog.dbf
*                  file. The program will join the location.dbf
*                  and stlog.dbf files by the control secction
*                  number.  The goal is to have tlog information
*                  arranged in sid number instead of control
*                  section number.
* Input file(s):   STLOG.DLM delimited TLOG data file
*                  LOCATION.DBF database with location data
* Output file(s): STLOG.NDX index file, deleted when done
* I/O file(s):    STLOG.DBF database to store the STLOG data
*                  SIDTLOG.DBF database with TLOG data by SID #

* inform the use that the program is running
? "<< SIDTLOG.PRG running ... >>"
? "   Arranging TLOG data by Section Identification Number."

* set flags
SET SAFETY OFF

* prepare the stlog database
USE \PAVEDB\EDITUPDT\TRAFFIC\STLOG
ZAP
APPEND FROM \PAVEDB\EDITUPDT\TRAFFIC\STLOG.DLM DELIMITED WITH BLANK
INDEX ON CNTLSEC TO \PAVEDB\EDITUPDT\TRAFFIC\STLOG
CLOSE ALL

* select files to be used
SELECT A
USE \PAVEDB\FILES\LOCATION
SELECT B
USE \PAVEDB\EDITUPDT\TRAFFIC\STLOG INDEX \PAVEDB\EDITUPDT\TRAFFIC\STLOG
SELECT C
USE \PAVEDB\EDITUPDT\TRAFFIC\SIDTLOG
ZAP

* for every location record find the tlog info
SELECT A
DO WHILE .NOT. EOF()
   IF A->ACTVFLAG

   * find the same control section in stlog file
   SELECT B
```

```
   IF (A->SECTION/10) >= 1
      ACNTLSEC = INT(VAL( STR(A->CONTROL,4)+STR(A->SECTION,2)))
   ELSE
      ACNTLSEC = INT(VAL( STR(A->CONTROL,4) + "0";
                     + LTRIM(STR(A->SECTION,2)) ))
   ENDIF
   SEEK ACNTLSEC

   IF FOUND()
      * if found then for all the stlog record of that control
      * section
      DO WHILE CNTLSEC=ACNTLSEC
         * check to see if the stlog record's mile point range
         * is within the range of location's
         IF ((TLOGBMP>=A->BEGMPNT).AND.(TLOGEMP<=A->ENDMPNT));
         .OR.((TLOGBMP>=A->BEGMPNT).AND.(TLOGBMP<A->ENDMPNT));
         .OR.((TLOGEMP>A->BEGMPNT).AND.(TLOGEMP<=A->ENDMPNT));
         .OR.((TLOGBMP<=A->BEGMPNT).AND.(TLOGEMP>=A->ENDMPNT))
            * if so, append a new sidtlog record with the info
            SELECT C
            APPEND BLANK
            REPLACE SID_NO WITH A-> SID_NO
            REPLACE BEGMPNT WITH A-> BEGMPNT
            REPLACE ENDMPNT WITH A-> ENDMPNT
            REPLACE TLOGBMP WITH B-> TLOGBMP
            REPLACE TLOGEMP WITH B-> TLOGEMP
            REPLACE DESYR   WITH B-> DESYR
            REPLACE CURYR   WITH B-> CURYR
            REPLACE ADT     WITH B-> ADT
            REPLACE ADTINCR WITH B-> ADTINCR
            REPLACE ESTADT  WITH B-> ESTADT
            REPLACE PCNTRK  WITH B-> PCNTRK
            REPLACE EST18K  WITH B-> EST18K
         ENDIF
         * skip to next stlog record
         SELECT B
         SKIP
      ENDDO
   ENDIF
   ENDIF
   * skip to next location record
   SELECT A
   SKIP
ENDDO
* close all the files
CLOSE ALL
* erase intermediate file
ERASE \PAVEDB\EDITUPDT\TRAFFIC\STLOG.NDX
* reset flags
SET SAFETY ON
* inform the user that the program is done
? "<< SIDTLOG.PRG done. >>"
```

SUBPROGRAM LOGTRAF.PRG - PROGRAM FLOW DIAGRAM

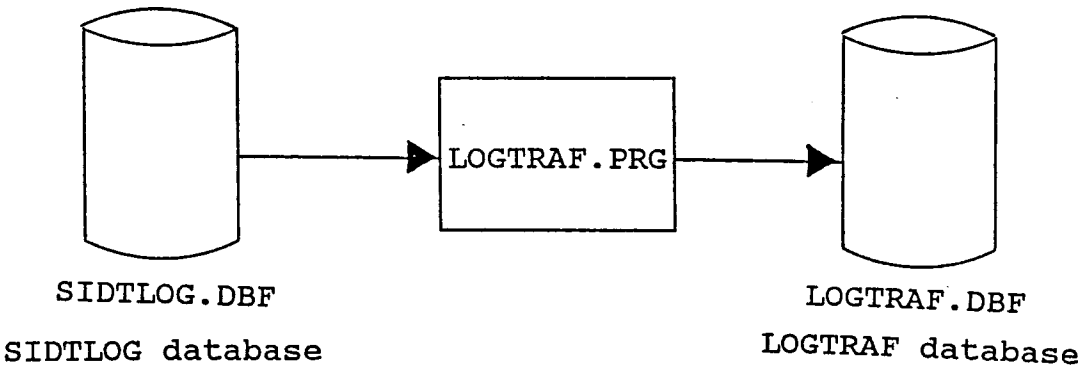


Diagram 4. Program flow for subprogram LOGTRAF.

FIGURE 22

Traffic Update
Subprogram 3: LOGTRAF

PROGRAM SPECIFICATION

Program Name: LOGTRAF.PRG

Purpose: To calculate the annual average daily traffic, the percent of trucks, and the cummulative 18 keal for the Traffic dBASE file by processing the TLOG data of each section identification number.

Procedures/Edits:

1. The TLOG data for section identification numbers is found in SIDTLOG.DBF.

2. The program calculates traffic values needed to update the Traffic database. These values of a new year are stored in dBASE LOGTRAF.DBF.

Input File(s): 1. dBASE file containing TLOG data for each section identification number (SIDTLOG.DBF).

Output File(s):

1. dBASE file with TLOG data under section identification number (SIDTLOG.DBF)

Traffic Update
Subprogram 3: LOGTRAF

LOGTRAF dBASE File Record Layout

File Name: LOGTRAF.DBF

Field	Key	Size/Type	Description
SID_NO	*	4N	Section Identification Number
YEAR	*	4N	Year
AADT1WAY		6N	Average Daily Traffic (one-way)
A18KEAL		9N	Annual Cummulative 18KEAL one way
PCTTRK		4N	Percent Trucks in ADT

NOTE: 18 KEAL is 18000 lbs. equivalent axle load. This figure is calculated using the AASHTO Equivalency factors which convert any weighted truck axle to a number of equivalent of 18000 lbs. single axles. The basics for the equivalency is the observed pavement damage done by different axle loads at the AASHTO road Test (1959-1960).

PROGRAM LISTING

```
* File name:      LOGTRAF.PRG
* Created by:     Victor Wong
* Written on:     Apr 13, 1988
* Last updated:   Apr 13, 1988
* Purpose:        To calculate the adt, 18KIP and percent truck values
*                 for test sections from sidtlog.dbf.  The result will
*                 be stored in logtraf.dbf
* I/O file(s):    LOGTRAF.DBF tlog traffic data base
* Input file(s): SIDTLOG.DBF tlog data base keyed on sid_no

* inform the user that the program is running
? "<< LOGTRAF.PRG running ... >>"
? "   Calculating the traffic data from the TLOG data."

* set flags
SET SAFETY OFF

* open files
SELECT A
USE \PAVEDB\EDITUPDT\TRAFFIC\LOGTRAF
ZAP
SELECT B
USE \PAVEDB\EDITUPDT\TRAFFIC\SIDTLOG

* while not eof for sidtlog
SELECT B
DO WHILE .NOT. EOF()
   * variables initializations
   PREV_SID  = SID_NO
   ACUM_PROP = 0.0
   ACUM_PCNTRK = 0.0
   ACUM_TRAF  = 0.0
   ACUM_18K  = 0.0

   * while it is still the same sid number
   DO WHILE SID_NO = PREV_SID
      * prior adjustment to some field values
      ADT_INCR = ADTINCR/100.0
      PCNT_TRK = PCNTRK/10.0 * 0.01
      EST_18K = EST18K * 1000.0
      CUR_YR = CURYR

      IF EST_18K <> 0.0
      * calculate the 18KIP and adt values for that control section
      TRK_18K = ((ESTADT + ADT)/4) * 365.25 * (DESYR - CURYR) * ;
                PCNT_TRK
      IF TRK_18K <> 0.0
         TRK_18K = EST_18K/TRK_18K
      ENDIF
```

```
      TRAF_ADT = (ADT/2) * 12 * 30.4375

      * calculate the proportion of the control section to the test section
      SEC_PROP =
(MIN(TLOGEMP,ENDMPNT)-MAX(TLOGBMP,BEGMPNT))/(ENDMPNT-BEGMPNT)
      ACUM_PROP = ACUM_PROP + SEC_PROP

      * calculate the accumulated traffic, 18kip and percent truck values
      * for the control sections
      ACUM_TRAF = ACUM_TRAF + (SEC_PROP * TRAF_ADT)
      ACUM_18K = ACUM_18K +(SEC_PROP *(TRAF_ADT *PCNT_TRK)*TRK_18K)

      ACUM_PCNTRK = ACUM_PCNTRK + (PCNT_TRK * 100 * SEC_PROP)

      ENDIF

      * skip to the next control section
      SKIP
   ENDDO

   * if the acum_prop is not zero
   IF ACUM_PROP <> 0.0
      TLOG_ADT = 0.5 + ACUM_TRAF/(ACUM_PROP * 365.25)
      TLOG_18K = 0.5 + ACUM_18K/ACUM_PROP
      TLOG_PCNTRK = ACUM_PCNTRK/ACUM_PROP
   ELSE
      TLOG_ADT = 0.0
      TLOG_18K = 0.0
      TLOG_PCNTRK = 0.0
   ENDIF

   * find the location of the sid no in logtraf.dbf and store the result
   SELECT A

   APPEND BLANK
   REPLACE SID_NO WITH PREV_SID
   REPLACE YEAR WITH CUR_YR
   REPLACE AADT1WAY WITH TLOG_ADT
   REPLACE A18KEAL WITH TLOG_18K
   REPLACE PCTTRK WITH TLOG_PCNTRK

   * get back with sidtlog.dbf
   SELECT B
ENDDO

* close all files
CLOSE ALL

* reset flags
SET SAFETY ON

* inform the user that the program is done
? "<< LOGTRAF.PRG done. >>"
```

SUBPROGRAM NEWTRAF.PRG - PROGRAM FLOW DIAGRAM

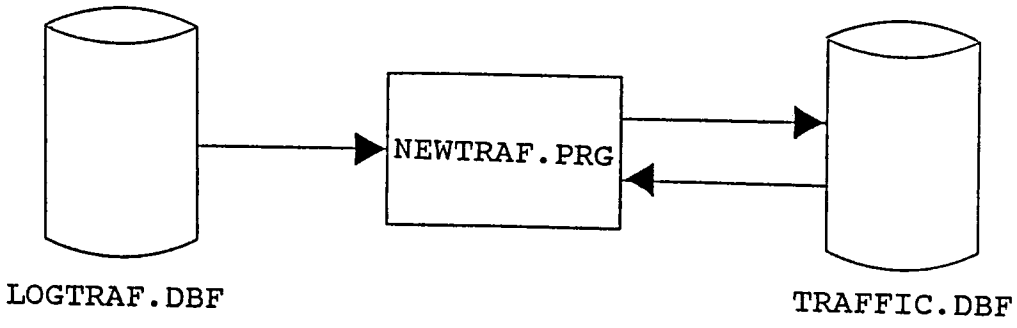


FIGURE 23

Traffic Update
Subprogram 4: NEWTRAF

PROGRAM SPECIFICATIONS

Program Name: NEWTRAF.PRG

Purpose: To use the new traffic data in the LOGTRAF dBASE file to update the Traffic database file.

Procedures/Edits:

1. The updating is done by appending records from the LOGTRAF dBASE file to TRAFFIC.DBF.

2. The Traffic database file is sorted by section identification number and the year after the appending of new data.

Input File(s):

1. Database file containing traffic values for a new year (LOGTRAF.DBF).

Output File(s):

1. Traffic database file (TRAFFIC.DBF).

PROGRAM LISTING

```
* File Name:          NEWTRAF.PRG
* Written by:         Victor Wong
* Created on:         April 13, 1988
* Last updated:       April 13, 1988
* Purpose:            A DBASE III + program that updates the traffic.dbf
*                     (after 1985) file with the logtraf.dbf file.
* Input file(s):      LOGTRAF.DBF
* Output file(s):     TRAFFIC.BK  (backup for TRAFFIC.DBF)
*                     TEMPTRAF.DBF (deleted when done)
* I/O file(s):        TRAFFIC.DBF

* inform the user that newtraf is running
? "<< NEWTRAF.PRG running ... >>"
? "   Updating the Traffic Database with data calculated from TLOG."

* set flags
SET SAFETY OFF

* make a backup of the old traffic database
COPY FILE \PAVEDB\FILES\TRAFFIC.DBF TO \PAVEDB\FILES\TRAFFIC.BK

* append the new traffic data
USE \PAVEDB\FILES\TRAFFIC
APPEND FROM \PAVEDB\EDITUPDT\TRAFFIC\LOGTRAF

* sort the traffic database according to sid_no and year
SORT TO \PAVEDB\FILES\TEMPTRAF ON SID_NO, YEAR
CLOSE ALL

* rename the new traffic database to traffic.dbf
ERASE \PAVEDB\FILES\TRAFFIC.DBF
RENAME \PAVEDB\FILES\TEMPTRAF.DBF TO \PAVEDB\FILES\TRAFFIC.DBF

* erase intermediate files
ERASE \PAVEDB\FILES\TEMPTRAF.DBF

* set flags
SET SAFETY ON

* inform the user that the program is done
? "<< NEWTRAF.PRG done ... >>"
```

CHAPTER 5

APPLICATIONS SUBSYSTEM

GENERAL NARRATIVE

The Applications Subsystem draws on the monitor an exponential decay curve for performance versus accumulated 18 KEAL where performance is indicated by PSI or area of distress for alligator cracking or rutting; creates a 'distress' file which contains the data needed to construct the curve; and builds a file which can be used to develop performance models.

The Distress File is a subset of data from the master files. It was created to substantially decrease the processing time for constructing the curves and for building the Model Files. The Distress File is also used to calculate values for the Model File. In addiition to the values created fromt he Distress File, the Model File also includes a subset of data from the master files. It too was created to substantially reduce the processing time involved in developing performance models. The Model File can be directly processed by SAS. For additional information refer to Volume 1 of this report.

This chapter is divided into the following four sections:

- Application Driver
- Graph Accumulated 18 KIP vs Distress or PSI
- Building Model File
- Create Distress File

Each section contains the following information:

- Program Narrative
- Program Flow Diagram
- Program Specification
- Menu Screens
- Program Listings

Section 1: Application Driver

Applications Subsystem Driver Program Flow

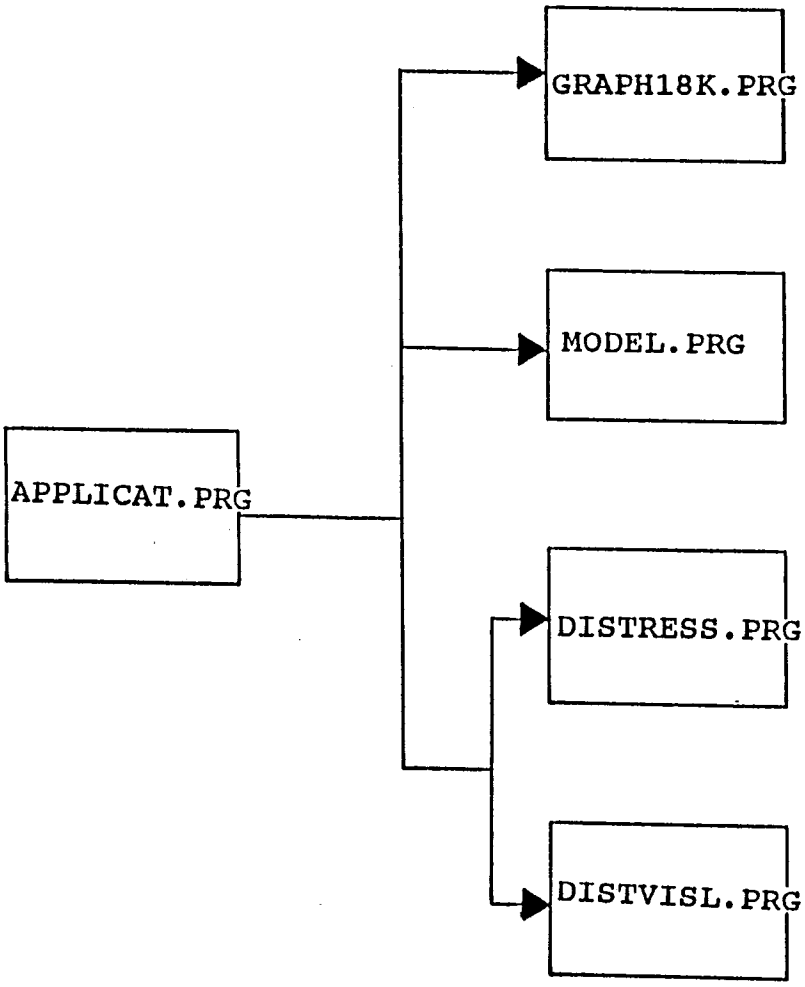


FIGURE 24

Application Driver
Program Narrative

The Application Subsystem main driver program (APPLICAT.PRG) initiates the programs which create the Distress & Model Files and the program which draws the 18 KIP vs Distress or PSI graphs on the screen.

In addition to driving the applications subsystem, APPLICAT.PRG also draws the menu for the application subsystem. When the user chooses an option on the applications menu, the appropriate program is called. If Option 1 (Graph Accumulated 18 KIP vs Distress or PSI) is chosen, program GRAPH18K is called. Option 2 (Build Model File) calls program MODEL.PRG while Option 3 (Build Distress File) calls program DISTRESS.PRG and DISTVISL.PRG. The program APPLICAT.PRG is in the subdirectory \PAVEDB\APPLICAT.

PROGRAM SPECIFICATION

Program Name: APPLICAT.PRG

Purpose: To display the Application Menu Screen and to call the appropriate programs.

Procedures\Edits:

The following are the procedures:

1) Graph the 18 KIP vs Distress or PSI (GRAPH18K.PRG).
2) Build the Model File (MODEL.PRG).
3) Create the Distress file (DISTRESS.PRG & DISTVISL.PRG).

Input\Output Files:

None

Programs Called (See Program Flow Diagram):

- GRAPH18K.PRG
- MODEL.PRG
- DISTRESS.PRG
- DISTVISL.PRG

MENU SCREEN FOR APPLICATION

```
          TEXAS FLEXIBLE PAVEMENT DATABASE
                    MAIN MENU


          1 - Inquiry
          2 - Reports
          3 - Edit & Update
          4 - Applications
          5 - Backup
          6 - Installation
          7 - Reindex Master Files

          Q - QUIT

                                    OPTION ===>
```

Above is Main Menu of the Texas Flexible Pavement System.

When Option 4 - Application is chosen, the application driver program APPLICAT.PRG is called. The applications driver then draws the next screen 4.0.

```
          TEXAS FLEXIBLE PAVEMENT DATABASE          4.0
                    APPLICATIONS MENU


          1)  Graph Accumulated 18KIP vs Distress or PSI
          2)  Build Model File
          3)  Create Distress File


                                    Option ===>
```

```
* File name:    APPLICAT.PRG
* Program name: application
* Project 2456: Texas Flexible Pavement Database
* TAMU/TTI
* Written by:   Victor Wong
* Created on:   July 14, 1988
* Last updated: July 26, 1988
* Purpose:      To call on application programs.

* set flag
SET SAFETY OFF

* initialize variables
REP = .T.

* repeat option menu until escape key is pressed
DO WHILE REP
   OPTION = " "
   * redisplay menu until valid option or escape is entered
   DO WHILE .NOT. (OPTION $ "123") .AND. REP
      * option menu
      CLEAR
      @ 5,24 SAY "TEXAS FLEXIBLE PAVEMENT DATABASE"
      @ 6,31 SAY "APPLICATIONS MENU"
      @ 9,15  SAY "1)  Graph Accumulated 18KIP vs Distress or PSI"
      @ 10,15 SAY "2)  Create Model File"
      @ 11,15 SAY "3)  Create Distress File"
      @ 14,45 SAY "Option ===> " GET OPTION PICTURE "X"
      @ 4,9  TO 16,70  DOUBLE

      * read option
      READ

      * set repeat flag to false if escape key is pressed
      IF READKEY() = 12
         REP = .F.
      ENDIF

      * warning fo invalid option entry
      IF .NOT. (OPTION $ "123") .AND. REP
         @20,10 SAY "Enter only 1, 2 or 3. "
         WAIT
         @20,10 SAY  "                    "
      ENDIF
   ENDDO

   * carry out option
   IF REP
      DO CASE
```

```
         CASE OPTION = "1"
            DO \pavedb\applicat\graph18k\GRAPH18K
         CASE OPTION = "2"
            DO \pavedb\applicat\model\MODEL
         CASE OPTION = "3"
            CLEAR
            MRUN = " "
            @ 10, 5 SAY "This program is going to create the Distress
Database"
            @ 11, 5 SAY "  File.  It will take the approximately 35 HOURS to
run."
            @ 13, 5 SAY "The DISTRESS file makes use of the monitoring data.
If"
            @ 14, 5 SAY "  the latest data is required, please run the
Pavement"
            @ 15, 5 SAY "  Condition Data Program (Option 2 on the Edit &
Update"
            @ 16, 5 SAY "  Menu) before this program."
            @ 19, 5 SAY "Do you want to continue (Y/N) ? " GET MRUN
            READ
            IF MRUN = "Y"
               DO \PAVEDB\EDITUPDT\DISTRESS\DISTRESS
               DO \PAVEDB\EDITUPDT\DISTRESS\DISTVISL
               CLEA TYPE
            ENDIF
            RELEASE MRUN
      ENDCASE
   ENDIF
ENDDO

* reset flags
SET SAFETY ON

RETURN
```

Section 2: Accumulated 18KIP vs Alligator Cracking/Rutting/PSI

GENERAL NARRATIVE

These programs graph the area of alligator cracking versus accumulated 18KIP, area of rutting versus accumulated 18KIP, and PSI versus accumulated 18 KIP for a selected section (SID). The graph is displayed on the monitor. The GRAPH18K.PRG program prompts for the distress type and the section identification number. It then retreives the 18KIP and distress information from DISTRESS.DBF calls GRAPH18K.PAS to calculate the rho and beta value and constructs the graph. The program specifications give more detailed information about the programs. The programs and the DAT files are stored in the subdirectory PAVEDB\EDITUPDT\APPLICAT\GRAPH18K.

The DBF files are stored in the subdirectory \PAVEDB\FILES. The NDX files are stored in the subdirectory \PAVEDB\FILES. The following are provided after this narrative:

- Program Flow Diagram
- Program Specification
  - GRAPH18K.PRG
- Graph Menu Screen
- File Layout - DISTTYP.DBF
- File Layout - DISTTYP.DAT
- File Layout - DISTRESS.DAT
- Program Listing - GRAPH18K.PRG
- Program Specification
  - GRAPH18K.PAS
- Program Listings
  (in the order mentioned in the Program Specifications)

Program Flow for Graphing 18KIP vs Area Distress or PSI

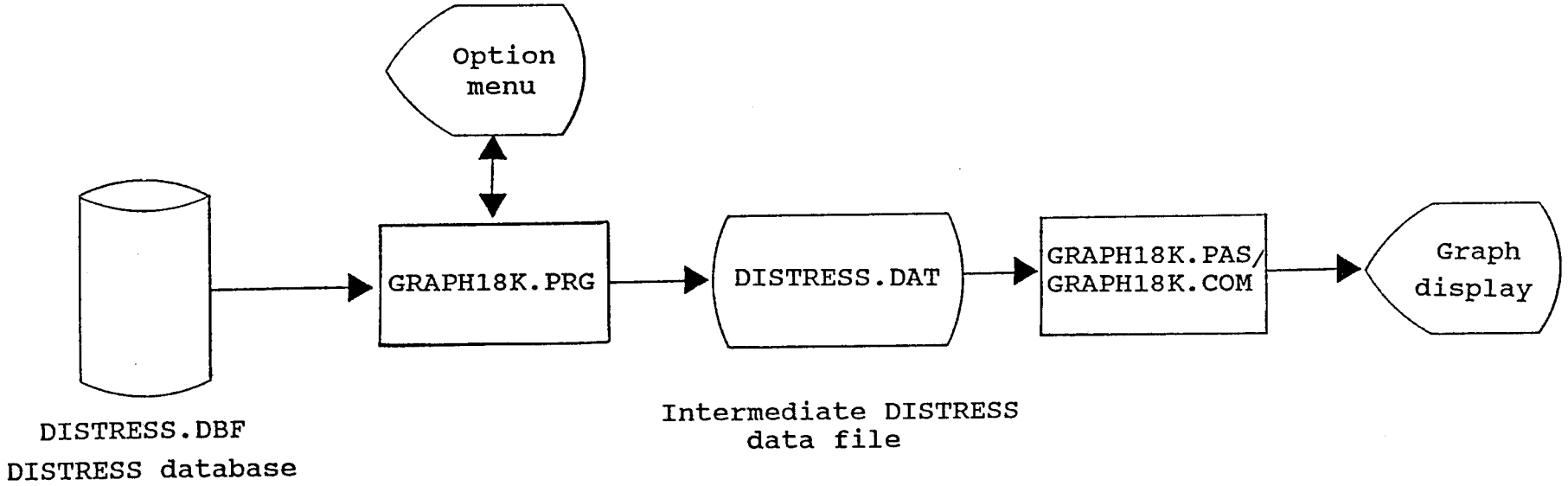


FIGURE 25

PROGRAM SPECIFICATION 1

Program Name: GRAPH18K.PRG

Purpose: To display a menu containing options to graph accumulated 18KIP versus the area of distress for alligator cracking or rutting, or to graph accumulated 18KIP versus PSI. This program will find and pass to a file the distress and 18KIP information needed for the chosen section identification number. With the retrieved data, a program is called to perform the processing and graphing.

Procedure/Edits:

1. The graphing program will be done using Turbo Pascal. Thus, the information needed to calculate the rho and beta values and to fit the curve is passed to the Pascal program (GRAPH18K.PAS) through an intermediate data file (DISTRESS.DAT). The distress type is passed in the file DISTTYP.DAT.

2. The data needed for program GRAPH18K.PAS is retrieved by locating the data of the chosen section from the DISTRESS dBASE file. The Location File is checked to determine if the requested SID is a valid number.

Input Files:

1. Distress dBASE file and index (DISTRESS.DBF) (DISTRESS.NDX).

2. Location dBASE file and index (LOCATION.DBF) (LOCATION.NDX).

3. Distress Type DBASE file (DISTTYP.DBF).

Intermediate Files:

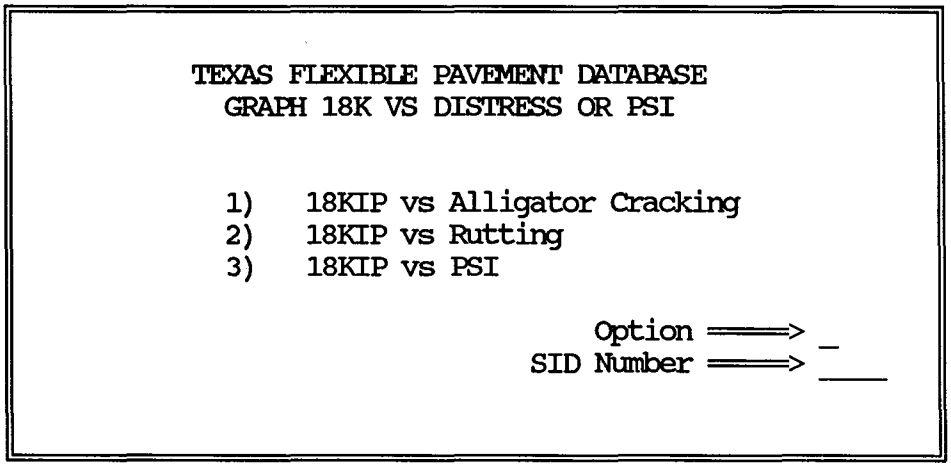
1. Intermediate distress data file (DISTRESS.DAT).

2. Intermediate distress type file (DISTTYP.DAT).

Programs Called: GRAPH18K.PAS

Accumulated 18KIP vs Alligator Cracking/Rutting/PSI

Program Specification 1
Graph Menu Screen



Accumulated 18 KIP vs Alligator Cracking/Rutting/PSI

Program Specification 1
File Layout

File Name: DISTTYP.DBF

Field	Size/Type	Description
DISTTYP	1C	Graph Option

Accumulated 18 KIP vs Alligator Cracking/Rutting/PSI

Program Specification 1
File Layout

File Name: DISTTYP.DAT

Column	Variable
1	Graph Option

Accumulated 18KIP vs Alligator Cracking/Rutting/PSI

Program Specification 1
File Layout

File Name: DISTRESS.DAT

Columns	Variable
1 - 4	Section Identification Number
5 - 6	Year
7 - 17	Annual Cummulative 18KIP Equivalent Axle Load
18	Patching Area Distress Value
19	Rutting Area Distress Value
20	Alligator Cracking Area Distress Value
21 - 27	Serviceability Index Mean Value

Accumulated 18KIP vs Alligator Cracking/Rutting/PSI

Program Specification 1
Program Listing

```
* File name:       GRAPH18K.PRG
* Program name:    graph18k
* Project 2456:    Texas Flexible Pavement Database
* TAMU/TTI
* Written by:      Victor Wong
* Created on:      July 25, 1988
* Last updated:    August 1, 1988
* Purpose:         To set up menu for graphing 18KIP vs certain
*                  type of area distress or PSI.

* set up databases
SELECT A
USE \pavedb\files\LOCATION INDEX \pavedb\indexes\LOCATION
  SELECT B
  USE \pavedb\files\DISTRESS INDEX \pavedb\indexes\DISTRESS SELECT C
USE \pavedb\files\DISTTYP

* initialize variables
PRIVATE REP
PRIVATE OPTION
REP = .T.

* repeat option menu until escape key is pressed
DO WHILE REP
   OPTION = " "
   SID = A->SID_NO
   * redisplay menu until valid option/SID or escape key
   DO WHILE .NOT. (OPTION $ "123") .AND. REP
      * option menu
      CLEAR
      @ 5,23 SAY "TEXAS FLEXIBLE PAVEMENT DATABASE"
      @ 7,23 SAY " GRAPH 18KIP VS DISTRESS OR PSI"

      @ 10,25 SAY "1)  18KIP vs Alligator Cracking"
      @ 11,25 SAY "2)  18KIP vs Rutting"
      @ 12,25 SAY "3)  18KIP vs PSI"

      @ 14,25 SAY "                        Option ==> ";
              GET OPTION PICTURE "X"
      @ 15,25 SAY "                   SID Number ==> ";
              GET SID PICTURE "9999"
      @ 3,9 TO 18,70 DOUBLE

      * read option
      READ

      * set repeat flag to false if escape key is pressed
      IF READKEY() = 12
```

```
         REP = .F.
      ENDIF

      * warning for invalid option
      IF .NOT. (OPTION $ "123") .AND. REP
         @20,10 SAY "Enter only 1, 2 or 3."
         WAIT
         @20,10 SAY "                        "
      ENDIF

   ENDDO

   * carry out option
   IF REP
      SELECT A
      SEEK SID
      IF .NOT. FOUND()
         GOTO 1
         @20,10 SAY "Invalid SID Number."
         WAIT
         @20,10 SAY "                        "
      ELSE
         * copy the distress information of the sid number to
         * a temporary file
         SELECT B
         SEEK STR(SID,4)
         IF FOUND()
           COPY TO \pavedb\applicat\graph18k\DISTRESS.DAT ;
           FIELDS SID_NO,YEAR,A18KEAL,PATCH,RUTT,ALLGCR,SIMEAN;
           WHILE SID_NO=SID TYPE SDF
           SELECT C
           GOTO 1
           REPLACE DISTTYP WITH OPTION
           COPY TO \pavedb\applicat\graph18k\DISTTYP.DAT;
            RECORD 1 TYPE SDF
         ELSE
           @20,10 SAY "Cannot find SID number in DISTRESS file."
           WAIT
           @20,10 SAY "                                          "
         ENDIF

         * perform the graphing
         RUN \pavedb\applicat\graph18k\GRAPH18K
      ENDIF
   ENDIF
ENDDO

* close databases
CLOSE ALL

RETURN
```

Accumulated 18KIP vs Alligator Cracking/Rutting/PSI

Program Specification 2

Program Name: GRAPH18K.PAS

Purpose: To graph the area of distress curve for alligator cracking/rutting or the PSI curve for the section identification number the user chooses. The curve is plotted after the program finds the rho and beta values of the curve.

Edit/Procedure Information:

1. This graphing program is written in Turbo Pascal. The information needed to calculate the rho and beta values, and to fit the curve is passed from the program GRAPH18K.PRG through an intermediate data file (DISTRESS.DAT). The distress type to be graphed is passed in DISTTYP.DAT

2. The equation for the area distress curve of alligator cracking and rutting is

$$\% \text{ area distress} = \exp - \left(\frac{\text{rho}}{\text{accumulated 18KIP}} \right)^{\text{beta}} .$$

   Refer to the program comments for addition information about solving for rho and beta.

3. The equation for the PSI curve is

$$\text{PSI} = \text{initial PSI} - (\text{initial PSI} - \text{final PSI}) * \exp - \left(\frac{\text{rho}}{\text{accumulated 18KIP}} \right)^{\text{beta}} .$$

   Refer to the program comments for additional information about solving for rho and beta.

4. The area of distress data for alligator cracking and rutting is collected using five discrete values representing the percentage of area of distress. The value "9" stands for missing data, and "0" stands for no area of distress. Otherwise, the following shows how the other three values are assigned:

	Code	% Area Distress	Plotting Value
Alligator Cracking	1	1 - 10	5.5
	2	11 - 50	30.5
	3	> 50	50.0
Rutting	1	1 - 25	13.0
	2	26 - 50	38.0
	3	> 50	50.0

5. The program attempts to compensate for an irregular shape area of distress curve for alligator cracking/rutting where the irregularity is due to patching. The percentage area value for patching is added to the alligator cracking/rutting value. This eliminates the dropping portion of the curve due to decreased area distress resulting from patching. The coding assignments for patching are as shown below.

	Code	% Area Distress	Plotting Value
Patching	0	0	0
	1	1 - 10	5.5
	2	11 - 50	30.5
	3	> 50	50.0

If the areas of distress still decreases after the patching has been added to the alligator cracking or rutting value, the point is dropped from the curve.

6. As in item 5, when the PSI inproves, the point is dropped from the curve.

Input File(s):

1. Intermediate distress data file (DISTRESS.DAT)

2. Intermediate distress type file (DISTTYP.DAT)

PASCAL Procedures Called:

DECLARE.PAS
GRAPH.P
MYGRAPH.PAS
RHOBETA.PAS
FITCURVE.PAS
PLOTGRPH.PAS

File Layouts:

1. DISTTYP.DAT (Refer to File Layout 2 in Program Specification 1).

2. DISTRESS.DAT (Refer to File Layout 3 in Program Specification 1).

Accumulated 18KIP vs Alligator Cracking/Rutting/PSI

Program Specification 2
Program Listing

```
{*  File name:    GRAPH18K.PAS
 *  Program name: graph18k
 *  Project 2456: Texas Flexible Pavement Database Conversion
 *  TAMU/TTI
 *  Written by:   Victor Wong
 *  Created on:   Feb 19, 1988
 *  Last updated: June 27, 1988
 *  Purpose:      This program creates an onscreen menu
 *                inquiring user for the type of distress and
 *                sid number.  It then call a procedure to find
 *                the distress information of the sid number
 *                from the distress database file.  With the
 *                information, the porocedure will process and
 *                plot the value on an onscreen graph.
}

{>>> PROGRAM GRAPH18K <<<}
program graph18k;

{>>> constant declaration <<<}
const
   color = 15;
   blank = ' ';
   psi_final = 1.5;
   debug_flag = true;

{>>> type declaration <<<}
{$I \pavedb\applicat\graph18k\declare}

{>>> variable declaration <<<}
var
   {--- variables for menu ---}
   distress_type : integer;
   response_ok,
   continue: boolean;
   response : char;

   {--- variables for data processing ---}
   data_file:        text;

   {input data format}
   sid_no:           string[4];    {sid number      4N}
   year:             string[2];    {year            2N}
   a18keal:          string[11];   {18kip           11N}
   patching:         string[1];    {patching        1N}
   rutting:          string[1];    {rutting         1N}
```

```
allgcrk:          string[1];    {alligator crack 1N}
simean:           string[7];    {simean        7.5N}

{variables needed for conversion of data from strings into }
{numnbers and variables needed for any numeric manipulation}
{with the data}
index:            integer;
char_at_index:    string[1];
err_code,
err_code1,                      {error code for string to}
err_code2:        integer;      {number conversion}
psi_value:        real;         {psi value}
patch_value,                    {patching value}
rutting_value,                  {rutting value}
allgcrk_value:    integer;      {alligator crack value}
patch_area,                     {patching area}
rutting_area,                   {rutting area}
allgcrk_area:     real;         {alligator crack area}
a18k1, a18k2,                   {if a18keal = aa bbb ccc ddd}
a18k3, a18k4,                   {then a18k1=aa,18k2=bbb,     }
a18k1_carry,                    {     81k3=ccc,18k4=ddd      }
a18k2_carry,
a18k3_carry,                    {carry from the digit group }
a18k4_carry,                    {for addition               }
accum_18k1,
accum_18k2,
accum_18k3,                     {accumulated 18k value by   }
accum_18k4:       integer;      {digit group                }
a18k1_zero,
a18k2_zero,
a18k3_zero,                     {flag saying if the digit   }
a18k4_zero:       boolean;      {group = 0                  }
a18keal1:         string[2];
a18keal2,
a18keal3,
a18keal4:         string[3];    {portions of the a18keal str}


{variables needed for graph}
psi_max,                        {maximum psi value}
psi_initial:      real;         {initial psi value}
psi_initial_str:  string[10];   {input string for initial psi}
psi,                            {psi value}
y:                y_array_type; {y value (allgcrk/rutt value)}
x,                              {x value (acccum. 18k value)}
x1,x2,x3, x4:     x_array_type; {x value by digit group}
y_factor,                       {the factor label for axis}
x_factor:         factor_type;
y_min, y_max,
y_interval:       y_type;
x_min, x_max,
x_interval:       x_type;
title:            title_type;
x_label,
```

```
   y_label:          label_type;
   id_string:        id_type;

   {--- other variables in the program ---}
   c:                char;
   i:                integer;
   found:            boolean;
   count:            integer;

{>>> include files <<<}
{$I \pavedb\applicat\graph18k\graph.p}
                          {file with turbo graphics procedure}
{$I \pavedb\applicat\graph18k\mygraph}
                          {file with my graphics procedures}
{$I \pavedb\applicat\graph18k\rhobeta}
                          {file with procedure to calculate}
                          {rho and beta}
{$I \pavedb\applicat\graph18k\fitcurve}
                          {file with procedure to fit curve}
{$I \pavedb\applicat\graph18k\plotgrph}
                          {file with procedure to plot graph}

{>>> beginning of program <<<}
begin
   {--- clear the screen ---}
   ClrScr;

   {--- initialize some variables---}
   accum_18k1:= 0;
   accum_18k2:= 0;
   accum_18k3:= 0;
   accum_18k4:= 0;

   {find the distress type}
   assign (data_file, '\pavedb\applicat\graph18k\disttyp.dat');
   reset (data_file);
   readln (data_file, distress_type);
   close (data_file);

   {set up input file for data}
   assign (data_file, '\pavedb\applicat\graph18k\distress.dat');
   reset (data_file);

   count:= 0;    {initialize the count of valid data to 0}
   psi_max:= 0.0;{initialize the maximum psi value to 0.0}
   patch_area:=0.005;{initialize the patch area to 0.0}

      {while not eof}
      {read data, accumulate 18kip value }
      {and store valid distress or psi value}
      while not eof(data_file)
      do begin
         readln (data_file, sid_no, year, a18keal, patching,
rutting, allgcrk, simean);
```

```
{convert the a81keal string value to 3 integer values}
{so that the concatenation of a18k1, a18k2 and a18k3 }
{will give the whole 18keal value                    }
{if the err_code is non-zero, it means that digits   }
{are missing at those positions and it is the same as}
{having zeros there                                  }
{the positions of leading zeros are indicated by the }
{a18kX_zero flag                                     }

{split a18keal value string into 4 sections }
{(xx,xxx,xxx,xxx)}
a18keal1:= copy(a18keal,1,2);
a18keal2:= copy(a18keal,3,3);
a18keal3:= copy(a18keal,6,3);
a18keal4:= copy(a18keal,9,3);

{delete leading blank spaces of a18kealx string}
{and store convert the string to integer a18kx values}
index:= 1;
char_at_index:= copy (a18keal1,index,1);
while (char_at_index = blank) and (index < 2) do
begin
   index:= index + 1;
   char_at_index:= copy (a18keal1,index,1);
end;
if index < 2 then
   val (copy(a18keal1,index,3-index),a18k1,err_code)
else if char_at_index <> blank then
   val (copy (a18keal1,index,3-index),a18k1,err_code)
else
   begin
   a18k1:= 0;
   a18k1_zero:= true;
   end;

index:= 1;
char_at_index:= copy (a18keal2,index,1);
while (char_at_index = blank) and (index < 3) do
begin
   index:= index + 1;
   char_at_index:= copy (a18keal2,index,1);
end;
if index < 3 then
   val (copy(a18keal2,index,4-index),a18k2,err_code)
else if char_at_index <> blank then
   val (copy (a18keal2,index,4-index),a18k2,err_code)
else
   begin
   a18k2:= 0;
   a18k2_zero:= true;
   end;

index:= 1;
```

```
char_at_index:= copy (a18keal3,index,1);
while (char_at_index = blank) and (index < 3) do
begin
   index:= index + 1;
   char_at_index:= copy (a18keal3,index,1);
end;
if index < 3 then
   val (copy(a18keal3,index,4-index),a18k3,err_code)
else if char_at_index <> blank then
   val (copy (a18keal3,index,4-index),a18k3,err_code)
else
   begin
   a18k2:= 0;
   a18k3_zero:= true;
   end;

index:= 1;
char_at_index:= copy (a18keal4,index,1);
while (char_at_index = blank) and (index < 3) do
begin
   index:= index + 1;
   char_at_index:= copy (a18keal4,index,1);
end;
if index < 3 then
   val (copy(a18keal4,index,4-index),a18k4,err_code)
else if char_at_index <> blank then
   val (copy (a18keal4,index,4-index),a18k4,err_code)
else
   begin
   a18k4:= 0;
   a18k4_zero:= true;
   end;

{if the a18keal value is not zero continue processing}
if not (a18k1_zero and a18k2_zero and a18k3_zero and
        a18k4_zero) then
begin
   {increment the accumulated 18kip value}
   accum_18k4:= accum_18k4 + a18k4;
   a18k4_carry:= accum_18k4 div 1000;
   accum_18k4:= accum_18k4 mod 1000;
   accum_18k3:= accum_18k3 + a18k3 + a18k4_carry;
   a18k3_carry:= accum_18k3 div 1000;
   accum_18k3:= accum_18k3 mod 1000;
   accum_18k2:= accum_18k2 + a18k2 + a18k3_carry;
   a18k2_carry:= accum_18k2 div 1000;
   accum_18k2:= accum_18k2 mod 1000;
   accum_18k1:= accum_18k1 + a18k1 + a18k2_carry;


   {--- check which distress type we are using ---}
   {--- and do the necessary processing        ---}
   case distress_type of
      1: {alligator cracking}
```

```
begin
{convert alligator crack data from string to}
{integer}
val (allgcrk,allgcrk_value,err_code1);

{convert patching data from string to integer}
val (patching,patch_value,err_code2);

{if the conversion is successful}
if (err_code1 = 0) and (err_code2 = 0) then

{check if the value is valid i.e. <> 9}
if (allgcrk_value <> 9) then
   begin
      {increment valid data count}
      count:= count + 1;

      {determine the percentage of alligator}
      {cracking area}
      case allgcrk_value of
         0: allgcrk_area:= 0.005;
         1: allgcrk_area:= 0.055;
         2: allgcrk_area:= 0.305;
         3: allgcrk_area:= 0.50;
      end;

      {determine the percentage of patching}
      {area}
      if patch_value <> 9 then
         case patch_value of
            0: patch_area:= 0.005;
            1: patch_area:= 0.055;
            2: patch_area:= 0.305;
            3: patch_area:= 0.50;
         end;

      {take the sum of the two area %}
      {as the value to be plotted    }
      y[count] := (allgcrk_area + patch_area);
      if y[count] >= 1.0 then
         y[count]:= 0.985;

      {store the new accumulated 18kip value}
      {in x arrays}
      x1[count]:= accum_18k1;
      x2[count]:= accum_18k2;
      x3[count]:= accum_18k3;
      x4[count]:= accum_18k4;
   end
else
   begin
   if err_code1 <> 0 then
   begin
      gotoxy (22,14);
```

```
            write ('***WARNING***');
            write (' bad data encountered');
            gotoxy (22,15);
            write ('                ');
            write ('INVALID CODE FOR ALLIGATOR ');
            write ('CRACKING ');
            gotoxy (22,17);
            write ('Still processing.');
            write ('  Please wait ...');
         end;
         if err_code2 <> 0 then
         begin
            gotoxy (22,14);
            write ('***WARNING***');
            write (' bad data encountered');
            gotoxy (22,15);
            write ('                ');
            write ('INVALID CODE FOR PATCHING');
            write ('           ');
            gotoxy (22,17);
            write ('Still processing.');
            write ('  Please wait ...');
         end;
         end;
      end;

   2: {rutting}
      begin
      {convert rutting data from string to integer}
      val (rutting,rutting_value,err_code1);

      {convert patching data from string to integer}
      val (patching,patch_value,err_code2);

      {if the conversion is successful}
      if (err_code1 = 0) and (err_code2 = 0) then

      {check if the value is valid i.e. <> 9}
      if (rutting_value <> 9) then
         begin
            {increment valid data count}
            count:= count + 1;

            {determine the percentage of rutting}
            {area}
            case rutting_value of
               0: rutting_area:= 0.005;
               1: rutting_area:= 0.13;
               2: rutting_area:= 0.38;
               3: rutting_area:= 0.50;
            end;

            {determine the percentage of patching}
            {area}
```

```
            if patch_value <> 9 then
               case patch_value of
                  0: patch_area:= 0.005;
                  1: patch_area:= 0.055;
                  2: patch_area:= 0.305;
                  3: patch_area:= 0.5;
               end;

            {take the sum of the two area %}
            {as the value to be plotted    }
            y[count]:= (rutting_area + patch_area);
            if y[count] >= 1.0 then
               y[count]:= 0.985;

            {store the new accumulated 18kip value}
            {in x arrays}
            x1[count]:= accum_18k1;
            x2[count]:= accum_18k2;
            x3[count]:= accum_18k3;
            x4[count]:= accum_18k4;
         end
      else
         begin
         if err_code1 <> 0 then
         begin
            gotoxy (22,14);
            write ('***WARNING*** ');
            write ('bad data encountered');
            gotoxy (22,15);
            write ('               ');
            write ('INVALID CODE FOR RUTTING   ');
            gotoxy (22,17);
            write ('Still processing.  ');
            write ('Please wait ...');
         end;
         if err_code2 <> 0 then
         begin
            gotoxy (22,14);
            write ('***WARNING*** ');
            write ('bad data encountered');
            gotoxy (22,15);
            write ('               ');
            write ('INVALID CODE FOR PATCHING   ');
            gotoxy (22,17);
            write ('Still processing.  ');
            write ('Please wait ...');
         end;
         end;
      end;

   3: {psi}
      begin
      {convert psi data from string to real}
      val (simean,psi_value,err_code);
```

```
                {if the conversion is successful}
                if (err_code = 0) then

                if psi_value <> 9.99999 then
                   begin
                      count:= count + 1;

                      {store the psi value in the psi array}
                      if psi_value = 0.0 then
                         psi[count]:= 0.0005
                      else
                         psi[count]:= psi_value;

                      {check for the maximum psi value}
                      if psi_value > psi_max then
                         psi_max:= psi_value;

                      {store the new accumulated 18kip value}
                      {in x arrays}
                      x1[count]:= accum_18k1;
                      x2[count]:= accum_18k2;
                      x3[count]:= accum_18k3;
                      x4[count]:= accum_18k4;
                   end
                else
                   if err_code <> 0 then
                   begin
                      gotoxy (22,14);
                      write ('***WARNING*** ');
                      write ('bad data encountered');
                      gotoxy (22,15);
                      write ('                ');
                      write ('INVALID PSI VALUE ');
                      gotoxy (22,17);
                      write ('Still processing.  ');
                      write ('Please wait ...');
                   end;
                end;
          end; {case}
      end; {if}
end;

{close input file}
close (data_file);

{ask user for an estimated initial psi value}
{the input value has to be larger than or   }
{equal to the existing maximum psi value    }
response_ok:= false;
psi_initial:= 4.5;
if distress_type = 3 then
  begin
     {ask for the estimated initial psi value}
```

```
      clrscr;              {clear the screen}
      HiRes;               {set high resolution graphics}
      DrawBox (80,56,560,144);
      gotoxy (21,10);
      write ('The program needs to know the estimated');
      gotoxy (21,11);
      write ('initial PSI value. (> ',psi_max:4:2,')');
      gotoxy (21,13);
      write ('Initial PSI value (RETURN = 4.5) -> ');

      {keep prompting for estimated initial psi value}
      {if the input is smaller than the maximum psi  }
      {value}
      repeat
         psi_initial:= 4.5;
         gotoxy (57,13);
         write ('          ');
         gotoxy (57,13);
         readln (psi_initial_str);
         val (psi_initial_str,psi_initial,err_code);
         if (err_code <> 0) then
            begin
               gotoxy (21,14);
               write ('Input must be numeric!');
            end
         else
            begin
               if psi_initial >= psi_max then
                  response_ok:= true
               else
                  begin
                     gotoxy (21,14);
                     write ('Value too small! Please reenter.');
                  end;
            end;
      until response_ok;

      {transform the psi value to a distress value}
      for i:= 1 to count do
         y[count]:= ((psi_initial-psi[count]) /
                     (psi_initial-psi_final)) /10;

      TextMode;  {clear screen and back to text mode}
    end; {if}

  {find a suitable accumulated 18kip factor for the}
  {y axis on the graph}
  case distress_type of
     1, 2: y_factor:= 'x 1';
     3:    y_factor:= 'x 1';
  end; {case}

  {find a suitable accumulated 18kip factor for the x axis}
  {on the graph}
```

```
{NOTE: 18 kip values are taken as the 3 most significant}
{      digits i.e. a number 1,234,567 will be expressed }
{      as 123 x 10^4 }
if accum_18k3=0 then
   begin
      x_factor:= 'x 1';
      for i:= 1 to count do x[i]:= x4[i];
   end
else if accum_18k2 = 0 then
   begin
   if (a18k3 div 100) > 0 then
      begin
      x_factor:= 'x 10^3';
      for i:= 1 to count do x[i]:= x3[i];
      end
   else if (a18k3 div 10) > 0 then
      begin
      x_factor:= 'x 10^2';
      for i:= 1 to count do
          x[i]:= x3[i]*10 + x4[i] div 100;
      end
   else
      begin
      x_factor:= 'x 10';
      for i:= 1 to count do
         x[i]:= x3[i]*100 + x4[i] div 10;
      end;
   end
else if accum_18k1 = 0 then
   begin
   if (a18k2 div 100) > 0 then
      begin
      x_factor:= 'x 10^6';
      for i:= 1 to count do x[i]:= x2[i];
      end
   else if (a18k2 div 10 ) > 0 then
      begin
      x_factor:= 'x 10^5';
      for i:= 1 to count do
         x[i]:= x2[i]*10 + x3[i] div 100;
      end
   else
      begin
      x_factor:= 'x 10^4';
      for i:= 1 to count do
         x[i]:= x2[i]*100 + x3[i] div 10;
      end;
   end
else
   begin
   if (a18k1 div 10) > 0 then
      begin
      x_factor:= 'x 10^8';
      for i:= 1 to count do
```

```
            x[i]:= x1[i]*10 + x2[i] div 100;
         end
      else
         begin
            x_factor:= 'x 10^7';
            for i:= 1 to count do
               x[i]:= x1[i]*100 + x2[i] div 10;
         end;
      end;

{set axis parameter before plotting the graph}
{set the appropriate values for y axis scale}
case distress_type of
1,2 : begin
            y_min:= 0.0;
            y_max:= 1.0;
            y_interval:= 0.25;
         end;
3:    begin
            y_min:= 0.0;
            y_max:= 5.0;
            y_interval:= 1.0;
         end;
end;

{set the appropriate values for x axis scale}
if (x[count] div 10) = 0 then
   begin
   x_min:= 10;
   x_max:= 0;
   x_interval:= 1;
   end
else
   begin
   x_min:= 0;
   {NOTE: the value 100 in the following equation for}
   {      x_max makes the curve more centered        }
   {      changing 100 to 10 will make the curve to  }
   {      shift more to the right and the range of   }
   {      the x scale to be smaller                  }
   x_max:= ((x[count] div 100) + 1) * 100;
   x_interval:= x_max div 10;
   end;

{set title, id and label}
case distress_type of
   1: begin
         title:= 'ALLIGATOR CRACK vs ACCUMULATED 18KIP';
         id_string:= 'SID #: ' + sid_no;
         x_label:= 'accumulated 18kip';
         y_label:= '% area distress';
      end;
   2: begin
         title:= 'RUTTING vs ACCUMULATED 18KIP';
```

```
            id_string:= 'SID #: ' + sid_no;
            x_label:= 'accumulated 18kip';
            y_label:= '% area distress';
          end;
       3: begin
            title:= 'PSI vs ACCUMULATED 18KIP';
            id_string:= 'SID #: ' + sid_no;
            x_label:= 'accumulated 18kip';
            y_label:= 'psi';
          end;
    end; {case}


    {plot the graph and points}
    case distress_type of
       1, 2: PlotGraph (title,id_string,x_label,y_label,
                        x_factor,y_factor,x,y,psi,psi_initial,
                        count, x_min, x_max, x_interval,
                        y_min, y_max, y_interval,d);
       3:    PlotGraph (title,id_string,x_label,y_label,
                        x_factor,y_factor,x,y,psi,psi_initial,
                        count, x_min, x_max, x_interval,
                        y_min, y_max, y_interval,p);
    end; {case}

{>>> ending of program <<<}
end.
```

Accumulated 18KIP vs Alligator Cracking/Rutting/PSI

Program Specification 2
Program Listing

```
{* File Name:   DECLARE.PAS}

type {type declarations needed for PlotGraph parameters}
   title_type    = string [60];
   id_type       = string [12];
   label_type    = string [30];
   factor_type   = string[10];
   x_type        = integer;
   y_type        = real;
   x_array_type  = array [1..72] of x_type;
   y_array_type  = array [1..72] of y_type;
   graph_type    = (d, p);
```

PROGRAM LISTING

```
{ * File name:    MYGRAPH.PAS
  * Written by:   Victor Wong
  * Created on:   Feb 19, 1988
  * Last updated: Mar 3, 1988
}
{>>> PROCEDURE: DRAWBOX (X1, Y1, X2, Y2) <<<}
{>>> This procedure draws a box using pixel coordinates (X1,Y1) as <<<}
{>>> the top left corner and (X2, Y2) as the bottom right corner.  <<<}

procedure DrawBox (x1, y1, x2, y2: integer);

begin
   Draw (x1, y1, x1, y2,1);
   Draw (x1, y2, x2, y2,1);
   Draw (x2, y2, x2, y1,1);
   Draw (x2, y1, x1, y1,1);
end;

{>>>> PROCEDURE: PLOTPOINT (X, Y, OPTION) <<<}
{>>> This procedure plots a point on the screen as one of the point <<<}
{>>> options available using the given x and y coordinates.         <<<}

procedure PlotPoint (x, y, option: integer);

begin
   case option of
      1: begin {cross}
            Draw (x-2, y-2, x+2, y+2, 1);
            Draw (x-2, y+2, x+2, y-2, 1);
         end;
      2: begin {plus}
            Draw (x-2, y, x+2, y, 1);
            Draw (x, y-2, x, y+2, 1);
         end;
      3: begin {asterisk}
            Draw (x-2, y, x+2, y, 1);
            Draw (x-2, y-2, x+2, y+2, 1);
            Draw (x-2, y+2, x+2, y-2, 1);
         end;
      4: begin {square}
            Draw (x-1,y-1,x+1, y-1, 1);
            Draw (x-1, y ,x+1,  y , 1);
            Draw (x-1,y+1,x+1, y+1, 1);
         end;
   end;
end;

^Z
```

Accumulated 18KIP vs Alligator Cracking/Rutting/PSI

Program Specification 2
Program Listing

```
{
* File name:       RHOBETA.PAS
* Written by:      Victor Wong
* Written on:      Feb 23, 1988
* Last updated:    Mar 8, 1988
* Procedure name:  find_rho_beta
* Purpose:         To find the rho and beta values for area or
*                  severity distress types.
*
* Methodology:     Equation being used is as follows.
*
*                                 rho    beta
*       distress = exp ( - (---------)     )
*                               traffic
*
*       When rho = traffic,
*                                      beta
*            distress = exp ( - ( 1 )     )
*                     = exp (-1)
*                     = 0.367
*
*       To find the rho for a distress vs traffic curve,
*       this methodology takes the first point as the reference
*       and does a linear estimate of where the rho should be
*       for each following point.  For each of the rhos, the
*       beta can be found by solving the above equation.  After
*       considering all the points, the maximum beta value is
*       taken and the corresponding rho is accepted.  These two
*       values will be the final rho and beta for the curve.
*
*       The linear estimation for the rho value is determined as
*       follows.
*
*       rho estimation for a point
*
*          distress when rho equals traffic - distress of point 1
*       =  ------------------------------------------------------
*               distress of the point - distress of point 1
*
*          x (traffic for the point - traffic of point 1) + traffic (1):
*
*
*                     0.367 - distress (1)
*       i.e. rho(i)= --------------------
*                     distress(i) - distress(1)
*
```

```
*                    * (traffic(i) - traffic(1)) + Traffic (1)
*
*       By rearranging the distress vs traffic equation, we get
*
*              ln (- ln (distress) )
*       beta = ---------------------
*                        rho
*                ln (-------)
*                    traffic
*
*       (Note that distress value is less than one.  Thus
*        - ln (distress) will give a positive number.  Otherwise,
*        ln (- ln (distress)) may be undefined.)
*
*
}

procedure find_rho_beta
   (traffic : x_array_type; {traffic values}
    distress: y_array_type; {distress values}
    no_of_pt: integer;      {no of traffic-distress}
                            {pairs                  }
    var final_rho, final_beta : real);

{>>> variables declaration <<<}
var
   denominator,
   rho,
   beta,
   max_beta : real;
   index,
   i        : integer;
   rho_of_pt,
   beta_of_pt: array[1..72] of real;

{>>> beginning of the procedure find_rho_beta<<<}
begin
   {If there is number of points is zero or less,
    then rho and beta values are zeros.}

   if (no_of_pt = 0) or (no_of_pt < 1) then
   begin
      rho:= 0.0;
      beta:= 1.0;
   end

   {If number of points = 88,
    then  rho = the first value in the traffic area
          beta = 0.5. }

   else if (no_of_pt = 88) then
   begin
      rho:= traffic[1];
      beta:= 0.5;
```

```
end

{If number of points = 99,
 then rho = 17.5 * 12.0
      beta = 1.0.}

else if (no_of_pt = 99) then
begin
   rho:= 17.5 * 12.0;
   beta:= 1.0;
end

{If number of points is greater than zero and not 88 or 99,
 then apply the methodology described above to find the rho     and beta.}

else
begin
   {initialize the rhos and betas for the points to zero}
   for i:= 1 to no_of_pt do
   begin
      rho_of_pt[i]:= 0.0;
      beta_of_pt[i]:= 0.0;
   end;

   {for all the n points, find the rhos and betas}
   for i:= 2 to no_of_pt do
   begin
      denominator:= 1.0 * (distress[i] - distress[1]);
      if (denominator <> 0.0) then
      begin
         rho_of_pt[i]:= (0.367 - distress[1])/denominator
                        * (traffic[i]-traffic[1])
                        + traffic[1];
         if rho_of_pt[i] > 0.0 then
         begin
            beta_of_pt[i]:= - ln (1.0 * distress[i]);
            beta_of_pt[i]:= ln(beta_of_pt[i]) /
                            ln(rho_of_pt[i]/traffic[i]);
         end;
      end;
   end;

   {find the maximum of the betas}
   max_beta:= beta_of_pt[1];
   index:= 1;
   for i:= 2 to no_of_pt do
      if max_beta < abs (beta_of_pt[i]) then
      begin
         max_beta:= beta_of_pt[i];
         index:= i;
      end;

   {choose the maximum beta and
          the rho value corresponding to the beta}
```

```
      rho:= rho_of_pt[index];
      beta:= max_beta;
   end;

   {return the final values for rho and beta}
   final_rho:= rho;
   final_beta:= beta;

{>>> ending of the procedure find_rho_beta <<<}
end;
```

Accumulated 18KIP vs Alligator Cracking/Rutting/PSI

Program Specification 2
Program Listing

```
{*************************************************************}
{*** Procedure FitCurve : Fit a curve to the given data    ***}
{***                     after finding the rho and beta   ***}
{***                     in the curve equation.           ***}
{*************************************************************}
{***   NOTE: The procedure find_rho_beta is needed.  And   ***}
{***         make sure the types in the procedural declar- ***}
{***         ation are well defined.                       ***}
{*************************************************************}

procedure fitcurve (x: x_array_type;
                    y: y_array_type;
                    no_of_pts: integer;
                    x_min, x_max, x_interval: x_type;
                    y_min, y_max, y_interval: y_type;
                    psi_initial: real;
                    distress_or_psi: graph_type);

{>>> constant declaration <<<}
const
   window_border_space = 10;
   window_width_space = 60;
   pixels_per_line = 8;
   pixels_per_char = 8;
   y_axis_lines = 8;

{>>> variable declaration <<<}
var
   rho, beta:          real;

   {--- variables necessary for defining graph window ---}
   x_right, x_left,
   y_top, y_bottom:    integer;

   {--- variables necessary for drawing the curve ---}
   spaces_per_interval,
   lines_per_interval: integer;
   delta_x:            x_type;
   x_begin, x_end:     x_type;
   y_begin, y_end:     y_type;
   x_begin_coord,
   x_end_coord,
   y_begin_coord,
   y_end_coord   :     integer;
   response:           char;
   y_valid:            boolean;

{>>> beginning of procedure fitcurve <<<}
```

```
begin
   {--- find the rho and beta of the curve ---}
   {--- detail on the equation of the curve---}
   {--- is in file RHOBETA                  ---}
   find_rho_beta (x, y, no_of_pts, rho, beta);

   {--- process only when rho is positive ---}
   if rho > 0 then
   begin
   {tell user to wait for curve fitting}
   GotoXY (30,8);
   write ('Wait !! Fitting the curve.');

   {--- set window boundary ---}
   y_top:= 8 * pixels_per_line;
   y_bottom:= 17 * pixels_per_line;
   x_left:= window_border_space * pixels_per_char;
   x_right:= (window_width_space+window_border_space)
             * pixels_per_char;

   {--- set the axis interval ---}
   spaces_per_interval:= window_width_space
                         div ((x_max-x_min) div x_interval);
   lines_per_interval:= y_axis_lines div
                        round((y_max-y_min)/y_interval);

   {--- initialize variables needed for drawing the curve ---}
   {--- incrementally                                     ---}
   delta_x := x_interval div 2;
   x_begin:= 1;
   x_end:= x_begin + delta_x;
   y_end:= 0;

   {--- keep drawing the curve incrementally as long as ---}
   {--- the x coordinate does not exceed the maximum x  ---}
   {--- value and the y coordinate does not exceed the  ---}
   {--- maximum y value                                 ---}
   while (x_end < x_max) and (y_end < y_max) do
   begin
      if distress_or_psi = d then
      begin
      {this demostrates how the curve equation is being
      transformed to the one which is used in this program
          y = exp (- (rho/x) ^ beta)
       => ln y = - (rho/x) ^ beta
       => ln (- ln y) = beta * ln (rho/x)
       => ln (ln y) = beta * ln (rho/x)          since ln y <= 0
       => ln y = - exp (beta * ln (rho/x))       since ln y <= 0
       => y = exp (- exp (beta * ln (rho/x)))
      }

      {use the equation to calculate the y value of the}
      {starting point of a curve segment}
      y_begin:= beta * ln (rho/x_begin);
```

```
if y_begin < 88 then
begin
   y_begin:= exp (y_begin);
   if y_begin < 88 then
      y_begin:= exp (- y_begin)
   else
      y_begin:= 0;
end
else
   y_begin:= 0;
if y_begin < 0.005 then y_begin:= 0;

{use the equation to calculate the y value of the ending}
{point}
{of a curve segment}
y_end:= beta * ln (rho/x_end);
if y_end < 88 then
begin
   y_end:= exp (y_end);
   if y_end < 88 then
      y_end:= exp (- y_end)
   else
      y_end:= 0;
end
else
   y_end:= 0;
if y_end < 0.005 then y_end:= 0;
end;

if distress_or_psi = p then
begin
  {this demostrates how the curve equation is being
   transformed to the one which is used in this program

                                rho  beta
   P = P  - (P  - P ) exp - (-----)
        0     0    f          18kip

   or
   y = psi_initial - (psi_initial-1.5) exp(-(rho/x)^beta)

   =>  (psi_initial-y) = exp (- (rho/x)^beta)
      ----------------
      (psi_initial-1.5)

   =>  (psi_initial-y) = exp (- exp (beta * ln (rho/x)))
      ----------------
      (psi_initial-1.5)

   => y = (psi_initial-y) -
          (psi_initial-1.5) *
          exp (-exp (beta * ln (rho/x)))

}
```

```
{use the equation to calculate the y value of the}
{starting point of a curve segment}
y_begin:= beta * ln (rho/x_begin);
if y_begin < 88 then
begin
   y_begin:= exp (y_begin);
   if y_begin < 88 then
      y_begin:= psi_initial -
                (psi_initial-1.5) * exp (- y_begin)
   else
      y_begin:= psi_initial;
end
else
   y_begin:= psi_initial;
if y_begin < 0.005 then y_begin:= 0;

{use the equation to calculate the y value of the}
{ending point of a curve segment}
y_end:= beta * ln (rho/x_end);
if y_end < 88 then
begin
   y_end:= exp (y_end);
   if y_end < 88 then
      y_end:= psi_initial -
              (psi_initial-1.5) * exp (- y_end)
   else
      y_end:= psi_initial;
end
else
   y_end:= psi_initial;
if y_end < 0.005 then y_end:= 0;
end;

{calculate the window coordinates using the x }
{and y values}
x_begin_coord:= trunc (x_begin * ((spaces_per_interval
                                    * pixels_per_char)
                                  / x_interval));
y_begin_coord:= trunc (y_begin * ((lines_per_interval
                                    * pixels_per_line)
                                  / y_interval));
x_end_coord:= trunc (x_end * ((spaces_per_interval
                                 * pixels_per_char)
                               / x_interval));
y_end_coord:= trunc (y_end * ((lines_per_interval
                                 * pixels_per_line)
                               / y_interval));

{if the y coordinate is 0, increment it by 1 pixel so }
{that the curve will line up with the x axis since the}
{x axis is 1 to 2 pixels from the window boundary}
if y_begin_coord = 0 then y_begin_coord:= 1;
if y_end_coord = 0 then y_end_coord:= 1;
```

```
      {draw the curve segemnt}
      Draw (x_begin_coord,
            y_bottom- y_top- y_begin_coord,
            x_end_coord,
            y_bottom- y_top- y_end_coord,
            1);

      {increment the x value}
      x_begin:= x_end;
      x_end:=   x_begin + delta_x;
   end;

   {--- take message off the screen ---}
   GotoXY (30,8);
   write ('                          ');

   end
   else
   begin
      GotoXY (30,8);
      write ('Sorry !! Cannot fit curve due to');
      GotoXY (30,9);
      write ('decreasing distress values.'Rho, Beta);
   end;

{>>> end of procedure fitcurve <<<}
end;
```

Accumulated 18KIP vs AlligatorCracking/Rutting/PSI

Program Specification 2
Program Listing

```
{*****************************************************************}
{*** Procedure PlotGraph : Plots a graph using the given     ***}
{***                      information on a black and white ***}
{***                      screen. (in High resolution mode)***}
{*****************************************************************}
{***   NOTE: Make sure when you use this procedure, the      ***}
{***         types in the procedure declaration are well     ***}
{***         defined.                                        ***}
{*****************************************************************}

   Procedure PlotGraph (title: title_type;
                        id: id_type;
                        x_label, y_label : label_type;
                        x_factor,
                        y_factor: factor_type;
                        x_values: x_array_type;
                        y_values: y_array_type;
                        psi_values: y_array_type;
                        psi_initial: real;
                        no_values:integer;
                        x_min,
                        x_max,
                        x_interval: x_type;
                        y_min,
                        y_max,
                        y_interval: y_type;
                        distress_or_psi: graph_type
                       );

   {>>> constants declaration <<<}
   const
      line_per_scr  = 25;
      char_per_line = 80;
      window_border_space= 10;
      window_width_space= 60;
      blank         = ' ';
      pixels_per_line = 8;
      pixels_per_char = 8;
      x_ndx_width = 4;
      y_ndx_width = 4;
      y_axis_lines = 8;

   {>>> variables declaration <<<}
   var
      x_value,
      y_value,
      indention : integer;
      response        : char;
```

```
    i, j            : integer;
    y_top, y_bottom,
    x_right, x_left,
    no_x_intervals,
    no_y_intervals,
    spaces_per_interval,
    lines_per_interval: integer;

{>>> beginning of procedure PlotGraph <<<}
begin
    {--- clear the screen first ---}
    ClrScr;

    {--- set the screen mode ---}
    {high resolution 640 x 200 with black background and one
     color}
    HiRes;

    {--- set x and y boundaries for the graph window ---}
    y_top:= 8 * pixels_per_line;
    y_bottom:= 17 * pixels_per_line;
    x_left:= window_border_space * pixels_per_char;
    x_right:= (window_width_space+window_border_space) *
              pixels_per_char;

    {--- output id on the left top corner ---}
    Writeln;
    Write ('   ',id);

    {--- output centered title to screen ---}
    indention:= (char_per_line - length (title)) div 2
                -length(id);
    for i:= 1 to indention do
       Write (blank);
    Writeln (title);

    {--- output y axis lable to screen ---}
    for i:= 1 to 3 do Writeln;
    Writeln ('   ', y_label);
    Writeln ('   (', y_factor, ')');
    Writeln;

    {--- output y axis values ---}
    no_y_intervals:= trunc ((y_max - y_min) / y_interval);
    if no_y_intervals <= y_axis_lines then
    begin
       lines_per_interval:= y_axis_lines div no_y_intervals;
       for i:= 1 to (y_axis_lines mod no_y_intervals) do
          writeln;
       for i:= no_y_intervals downto 1 do
       begin
          for j:= 1 to (window_border_space-y_ndx_width-2) do
             write (blank);
          write ((y_interval*i+y_min):y_ndx_width:2);
```

```
      for j:= 1 to lines_per_interval do writeln;
   end;
   for j:= 1 to (window_border_space-y_ndx_width-2) do
      write (blank);
   writeln (y_min:y_ndx_width:2);
end;

{--- output x axis values ---}
writeln;
for i:= 1 to (window_border_space-x_ndx_width) do
   write (blank);
no_x_intervals:= (x_max - x_min) div x_interval;
write (x_min:x_ndx_width);
spaces_per_interval:= window_width_space
                      div no_x_intervals;
if spaces_per_interval > x_ndx_width then
begin
   for i:= 1 to (spaces_per_interval-x_ndx_width) do
      write (blank);
   for i:= 1 to no_x_intervals do
   begin
      write ((x_interval*i+x_min):x_ndx_width);
      for j:= 1 to (spaces_per_interval-x_ndx_width) do
         write (blank);
   end;
   writeln;
end;

{--- output x axis label to screen ---}
Writeln;
indention:= (char_per_line - length (x_label)) div 2;
for i:= 1 to indention do
   Write (blank);
Writeln (x_label);
for i:= 1 to indention do
   Write (blank);
Writeln ('(', x_factor, ')');

{--- output prompt to ask user to return to main menu  ---}
Writeln;
indention:= (char_per_line - 40) div 2;
for i:= 1 to indention do
   Write (blank);
Write ('< Press RETURN to return to menu ... >');

{--- draw the box for the screen output ---}
DrawBox (1,1,639,199);

Draw (1,3*pixels_per_line,639,3*pixels_per_line,1);
Draw (1,22*pixels_per_line,639,22*pixels_per_line,1);

{--- define graph output window ---}
GraphWindow (x_left, y_top, x_right, y_bottom);
```

```
    {--- draw y axis ---}
    Draw (1,1, 1,(y_bottom-y_top-1),1);

    {--- draw x axis ---}
    Draw (1,(y_bottom-y_top-1), (x_right-x_left-1),
          (y_bottom-y_top-1),1);

    {--- plot points ---}
    for i:= 1 to no_values do
    begin
       x_value:= x_values[i] * (spaces_per_interval *
                 pixels_per_char) div x_interval;
       case distress_or_psi of
          d: y_value:= trunc ((y_values[i] *
                     (lines_per_interval
                      * pixels_per_line) / y_interval));
          p: y_value:= trunc ((psi_values[i] *
                     (lines_per_interval
                      * pixels_per_line) / y_interval));
       end;
       if y_value = 0
       then y_value:= (y_bottom - y_top)-2
       else y_value:= (y_bottom -y_top)- y_value;
       PlotPoint (x_value, y_value,1);
    end;

    {--- fit curve to points ---}
    FitCurve (x_values, y_values, no_values, x_min, x_max,
              x_interval, y_min, y_max, y_interval,
              psi_initial, distress_or_psi);

    {--- wait for user to respond to prompt ---}
    Read (response);

    {--- return to text mode and clear screen ---}
    TextMode;
    ClrScr;

end;
{>>> ending of procedure PlotGraph <<<}
```

Section 3: Building Model File

GENERAL NARRATIVE

This option allows the user to build a model file which may be used for roadway analysis. The model file contains various information collected from many system database files. This includes location, distress, layer, environmental, traffic and deflection data. Because of the amount of data involved, processing all this information is time-consuming. To allow the flexibility of building the model file section by section, this option provides an alternative to process the above-mentioned data groups individually. However, it is advisory to use the automatic process in most cases to avoid unnecessary confusion.

MODEL.PRG is the driver program for building the model file. The program specifications for the programs called by MODEL.PRG follow the MODEL.PRG program listing. Unless otherwise indicated in a program specification, the DBF files are stored in the subdirectory \PAVEDB\FILES, the NDX files are stored in the subdirectory \PAVEDB\INDEXES, and the DAT files are stored in the subdirectory \PAVEDB\APPLICAT\MODEL. The programs are stored in the subdirectory \PAVEDB\APPLICAT\MODEL unless indicated otherwise in a program specification.

Program Flow for Building Model File Automatically by Choosing Option 1 of Mode.Prg Menu

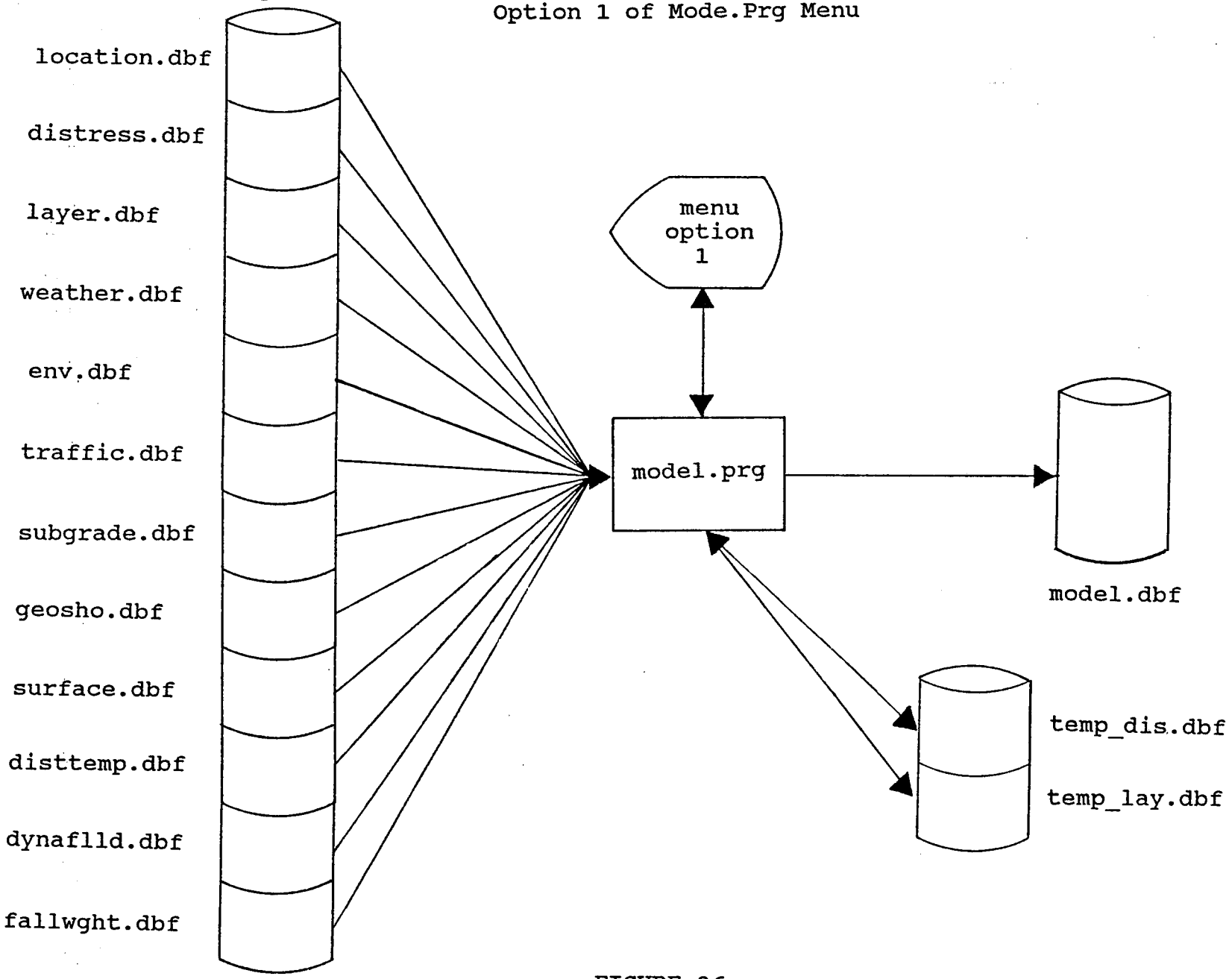


FIGURE 26

Program Flow for Building Model File by Individual Data Files
by Choosing Option 2 of Model.Prg Menu

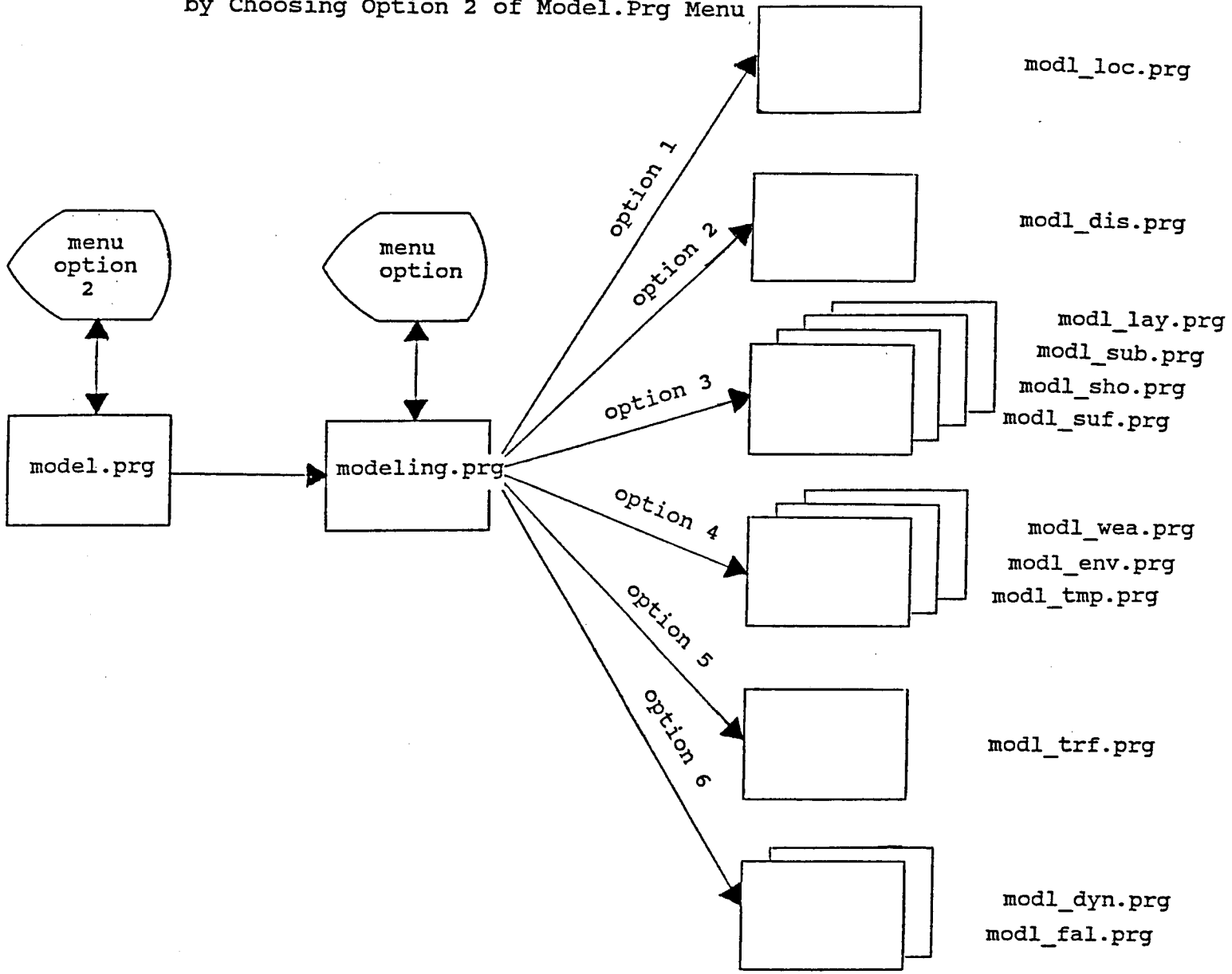


FIGURE 27

Building Model File

Program Specification 1

Program Name: MODEL.PRG

Purpose: To set up a menu for the option to build a model file. This program will ask the user to choose between two menu options: 1) To build the model file automatically, or 2) To build the model file by Individual data files.

Edit/Procedure Information:

1. The automatic option should be the one taken under most conditions. It will extract all the necessary data for the model file from various database files in the system.

2. The individual data files option should be taken with caution. It allows the user to extract information for the model file individually.

Input File(s):

1. Location database file (LOCATION.DBF) and index file (LOCATION.NDX).

2. Distress database file (DISTRESS.DBF) and index file (DISTRESS.NDX).

3. Layer database file (LAYER.DBF) and index file (LAYNDX.NDX).

4. Weather database file (WEATHER.DBF) and index file (WEATHER.NDX).

5. Environment database file (ENV.DBF) and index file (ENV.NDX).

6. Traffic database file (TRAFFIC.DBF) and index file (TRAFFIC.NDX).

7. Subgrade database file (SUBGRADE.DBF) and index file (SUBGNDX.NDX).

8. Shoulder database file (GEOSHO.DBF) and index file (GEONDX.NDX).

9. Surface database file (SURFACE.DBF) and index file (SURFNDX.NDX).

10. District temperature database file (DISTTEMP.DBF) and index file (DISTTEMP.NDX).

11. Dynaflec database file (DYNAFLLD.DBF) and index file (DYNAFLLD.NDX).

12. Falling weight database file (FALLWGHT.DBF) and index file (FALLWGHT.NDX).

Intermediate and Input/Output File(s):

1. Temporary distress database file (TEMP_DIS.DBF).

2. Temporary layer database file (TEMP_LAY.DBF).

3. Intermediate distress data file (DISTRESS.DAT).

4. Intermediate rho-beta-value data file (RHOBETA.DAT).

5. Intermediate base thickness data file (BASETHCK.DAT).

Output File(s):

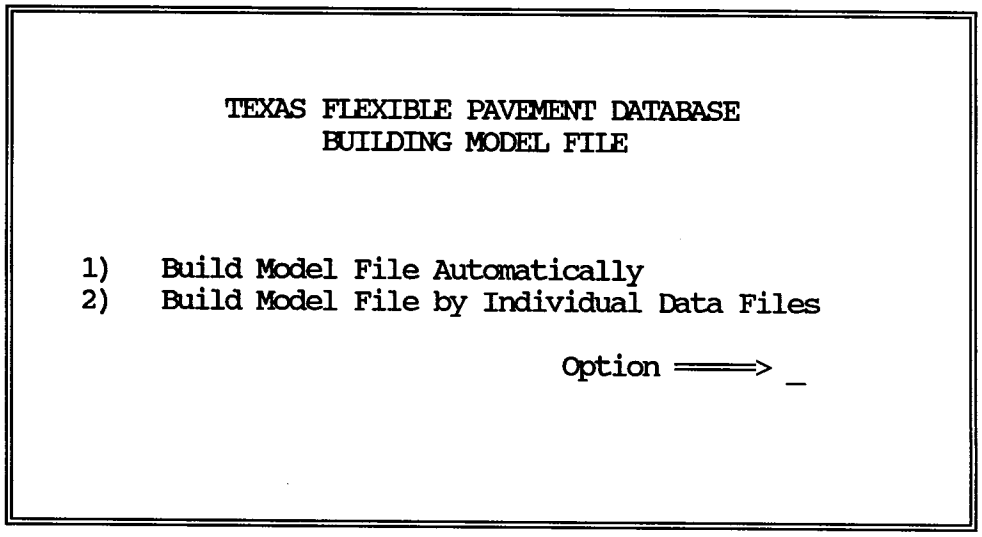
1. Model database file (MODEL.DBF).

dBASE Program Called:

MODELING.PRG
MOD_LOC.PRG
MODL_DIS.PRG
MODL_LAY.PRG
MODL_WEA.PRG
MODL_ENV.PRG
MODL_TRF.PRG
MODL_SUB.PRG
MODL_SHO.PRG
MODL_SUF.PRG
MODL_TMP.PRG
MODL_DYN.PRG
MODL_FAL.PRG

Building Model File

Program Specification 1
Model Menu Screen



Building Model File

Program Specification 1
Intermediate and Input/Output File Layout 1

File Name: TEMP_DIS.DBF

Field	Key	Field Name	Type	Width	Dec
1		SID_NO	Numeric	4	
2		PAVETYP	Numeric	2	
3		ALLG_RHO	Numeric	7	4
4		ALLG_BETA	Numeric	7	4
5		RUTT_RHO	Numeric	7	4
6		RUTT_BETA	Numeric	7	4
7		PSI_RHO	Numeric	7	4
8		PSI_BETA	Numeric	7	4

Building Model File

Program Specification 1
Intermediate and Input/Output File Layout 2

File Name: TEMP_LAY.DBF

Field	Key	Field Name	Type	Width	Dec
1		SID_NO	Numeric	4	
2		BASETYPE	Numeric	2	
3		BASETHCK	Numeric	7	2

Building Model File

Program Specification 1
Intermediate and Input/Output File Layout 3

File Name: DISTRESS.DAT

Columns	Size/Type	Variables
1 - 4	4N	Section Identification Number
5 - 6	2N	Year
7 - 17	11N	Annual Cummulative 18KIP Equivalent Axle Load
18	1N	Patching Area Distress Value
19	1N	Rutting Area Distress Value
20	1N	Alligator Cracking Area Distress Value
21 - 27	7.5N	Serviceability Index Mean Value

Building Model File

Program Specification 1
Intermediate and Input/Output File Layout 4

File Name: RHOBETA.DAT

Columns	Size/Type	Variables
1 - 2	2N	Pave Type
3 - 9	7.4N	Rho Value for Alligator Cracking Area Distress Curve
10 - 16	7.4N	Beta Value for Alligator Cracking Area Distress Curve
17 - 23	7.4N	Rho Value for Rutting Area Distress Curve
24 - 30	7.4N	Beta Value for Rutting Area Distress Curve
31 - 37	7.4N	Rho Value for PSI Curve
38 - 44	7.4N	Beta Value for PSI Curve

Building Model File

Program Specification 1
Intermediate and Input/Output File Layout 5

File Name: BASETHCK.DAT

Columns	Size/Type	Variables
1 - 4	4N	Section Identification Number
5 - 6	2N	Structure Number
7 - 8	2N	Layer Number
9 - 10	2N	Layer Description
11 - 12	2N	Layer Material
13 - 17	5.2N	Center Thickness of Layer

Building Model File

Program Specification 1
Program Listing

```
* File name:    MODEL.PRG
* Program name: model
* Project 2456: Texas Flexible Pavement Database
* TAMU/TTI
* Written by:   Victor Wong
* Created on:   July 14, 1988
* Last updated: July 26, 1988
* Purpose:      To call on modeling programs.

* initialize variables
PRIVATE OPTION
PRIVATE REP
REP = .T.

* repeat option menu if escape key is not pressed
DO WHILE REP
   OPTION = " "
   * redisplay menu unitl invalid option or escape is entered
   DO WHILE .NOT. (OPTION $ "12") .AND. REP
      * option menu
      CLEAR
      @ 6,24 SAY "TEXAS FLEXIBLE PAVEMENT DATABASE"
      @ 7,24 SAY "      BUILDING MODEL FILE"

      @ 11,15 SAY "1)  Build Model File Automatically"
      @ 12,15 SAY "2)  Build Model File by Individual Data Files"

      @ 14,42 SAY "Option ======> " GET OPTION PICTURE "X"
      @ 3,9  TO 18,70  DOUBLE

      * read option
      READ

      * set repeat flag to false if escape key is pressed
      IF READKEY() = 12
         REP = .F.
      ENDIF

      * warning for invalid option entry
      IF .NOT. (OPTION $ "12") .AND. REP
         @20,10 SAY "Enter only 1 or 2."
         WAIT
         @20,10 SAY  "                    "
      ENDIF
   ENDDO

   * carry out the option
   IF REP
```

```
      DO CASE
         CASE OPTION = "1"
            ANS = " "
            DO WHILE .NOT. (ANS $ "yYnN")
               CLEAR
               ANS = "N"
               @10,20 SAY "This process will take more than 2
                      hours."
               @11,20 SAY "Do you still want this option?(Y/N) ";
           GET ANS PICTURE "X"
               READ
            ENDDO
            IF (ANS $ "Yy")
               DO \pavedb\applicat\model\MODL_LOC
               DO \pavedb\applicat\model\MODL_DIS
               DO \pavedb\applicat\model\MODL_LAY
               DO \pavedb\applicat\model\MODL_WEA
               DO \pavedb\applicat\model\MODL_ENV
               DO \pavedb\applicat\model\MODL_TRF
               DO \pavedb\applicat\model\MODL_SUB
               DO \pavedb\applicat\model\MODL_SHO
               DO \pavedb\applicat\model\MODL_SUF
               DO \pavedb\applicat\model\MODL_TMP
               DO \pavedb\applicat\model\MODL_DYN
               DO \pavedb\applicat\model\MODL_FAL
            ENDIF
         CASE OPTION = "2"
            DO \pavedb\applicat\model\MODELING
      ENDCASE
   ENDIF
ENDDO

RETURN
```

Building Model File

Program Specification 2

Program Name: MODELING.PRG

Purpose: To set up a menu for building the model file by extracting information from individual group of data.

Edit/Procedure Information:

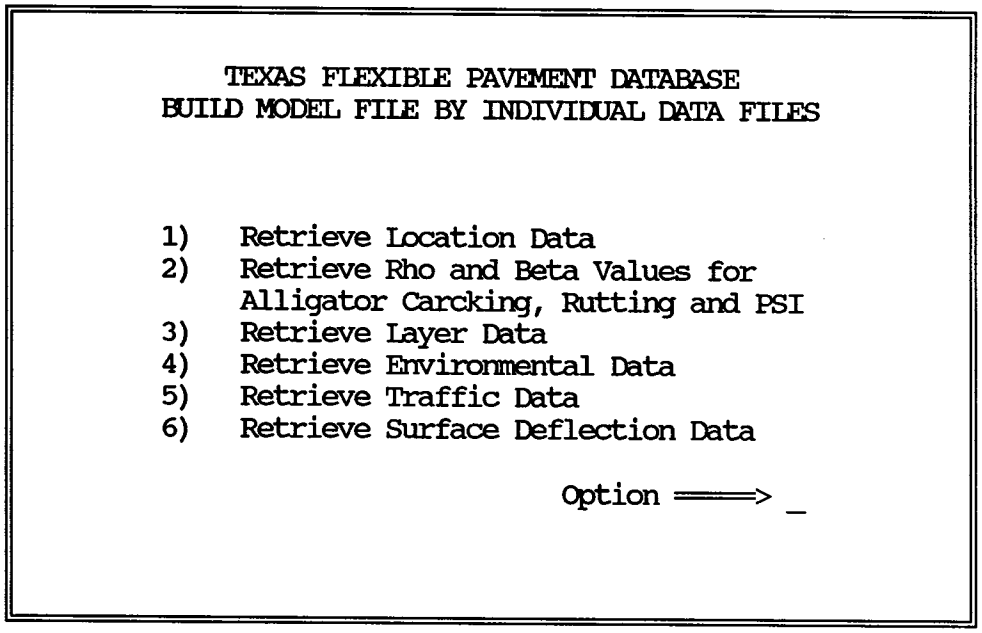
1. This program calls the appropriate subprograms to extract information from various system database files.

dBASE Programs Called:

MODL_LOC.PRG
MODL_DIS.PRG
MODL_LAY.PRG
MODL_WEA.PRG
MODL_ENV.PRG
MODL_TRF.PRG
MODL_SUB.PRG
MODL_SHO.PRG
MODL_SUF.PRG
MODL_TMP.PRG
MODL_DYN.PRG
MODL_FAL.PRG

Building Model File

Program Specification 2
Model Menu Screen



Building Model File

Program Specification 2
Program Listing

```
* File name:    MODELING.PRG
* Program name: modeling
* Project 2456: Texas Flexible Pavement Database
* TAMU/TTI
* Written by:   Victor Wong
* Created on:   July 14, 1988
* Last updated: July 26, 1988
* Purpose:      To call on modeling programs individually.

* initialize variables
PRIVATE OPTION
PRIVATE REP
REP = .T.
* repeat option menu if escape key is not pressed
DO WHILE REP
   OPTION = " "
   * redisplay menu until valid option or escape is pressed
   DO WHILE .NOT. (OPTION $ "123456") .AND. REP
      CLEAR
      @ 5,24 SAY "TEXAS FLEXIBLE PAVEMENT DATABASE"
      @ 6,20 SAY "BUILD MODEL FILE BY INDIVIDUAL DATA FILES"

      @  9,20 SAY "1)  Retrieve Location Data"
      @ 10,20 SAY "2)  Retrieve Rho and Beta Values for"
      @ 11,20 SAY "    Alligator Cracking, Rutting and PSI"
      @ 12,20 SAY "3)  Retrieve Layer Data"
      @ 13,20 SAY "4)  Retrieve Environmental Data"
      @ 14,20 SAY "5)  Retrieve Traffic Data"
      @ 15,20 SAY "6)  Retrieve Surface Deflection Data"

      @ 17,42 SAY "Option ======> " GET OPTION PICTURE "X"
      @ 3,9  TO 20,70  DOUBLE

      * read option
      READ

      * set repeat flag to false if escape key is pressed
      IF READKEY() = 12
         REP = .F.
      ENDIF

      * warning for invalid option entry
      IF .NOT. (OPTION $ "123456") .AND. REP
         @20,10 SAY  "Enter only 1 through 6."
         WAIT
         @20,10 SAY  "                        "
      ENDIF
   ENDDO
```

```
   * carry out the option
   IF REP
      DO CASE
         CASE OPTION = "1"
            CLEAR
            @10,15 SAY "This option will delete the most recent
                   model file,"
            @11,15 SAY "and rebuild the model file. "
            ANS = "N"
            @12,15 SAY "Do you still want to choose this option?
                   (Y/N) ";
                   GET ANS PICTURE "X"
            READ
            IF (ANS="Y") .OR. (ANS="y")
               CLEAR
               @10,18 SAY "Retrieving Location Data for Model
                      File..."
               DO \pavedb\applicat\model\MODL_LOC
            ENDIF
         CASE OPTION = "2"
            CLEAR
            @10,15 SAY "Retrieving Rho and Beta Values for Model
                   File..."
            DO \pavedb\applicat\model\MODL_DIS
         CASE OPTION = "3"
            CLEAR
            @ 10,20 SAY "Retrieving Layer Data for Model File..."
DO \pavedb\applicat\model\MODL_LAY
            DO \pavedb\applicat\model\MODL_SUB
            DO \pavedb\applicat\model\MODL_SHO
            DO \pavedb\applicat\model\MODL_SUF
         CASE OPTION = "4"
            CLEAR
            @ 10,15 SAY "Retrieving Environmental Data for Model
                    File..."
            DO \pavedb\applicat\model\MODL_WEA
            DO \pavedb\applicat\model\MODL_ENV
            DO \pavedb\applicat\model\MODL_TMP
         CASE OPTION = "5"
            CLEAR
            @10,20 SAY "Retrieving Traffic Data for Model
                   File..."
            DO \pavedb\applicat\model\MODL_TRF
         CASE OPTION = "6"
            CLEAR
            @10,14 SAY "Retrieving Surface Deflection Data for
                   Model File..."
            DO \pavedb\applicat\model\MODL_DYN
            DO \pavedb\applicat\model\MODL_FAL
      ENDCASE
   ENDIF
ENDDO

RETURN
```

Building Model File

Program Specification 3

Program Name: MODL_LOC.PRG

Purpose: To find the location data for the model file.

Edit/Procedure Information:

1. The program takes each active section identification number in the location database and inspects its most recent widening flag from the geometric shoulder database. It ignores the section if the program cannot find the section in the geometric and shoulder database file or the widening flag value is two.

2. If the section should not be ignored, copy the section identification number, highway prefix, highway district and county number to the model file.

3. When this program is run, it destroys the old contents in the model file (MODEL.DBF).

Input File(s):

1. Location database file (LOCATION.DBF) with index file (LOCATION.NDX).

2. Geometric shoulder database file (GEOSHO.DBF) with index file (GEONDX.NDX).

Output File(s):

1. Model database file (MODEL.DBF).

File Layout(s):

1. LOCATION.DBF (See Appendix A.)

2. GEOSHO.DBF (See Appendix A.)

3. MODEL.DBF (See Appendix A.)

Program Flow for Modl_Loc.Prg

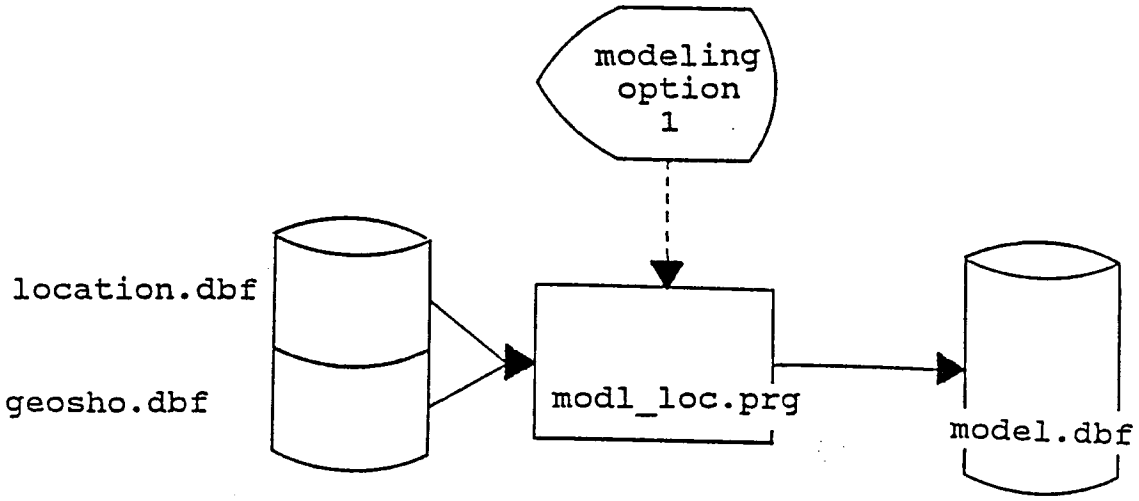


FIGURE 28

Building Model File

Program Specification 3
Program Listing

```
* File name:    MODL_LOC.PRG
* Program name: modl_loc
* Project 2456: Texas Flexible Pavement Database
* TAMU/TTI
* Written by:   Victor Wong
* Created on:   June 29, 1988
* Last updated: July 14, 1988
* Purpose:      To find location data for the model file.

* open files
SELECT A
USE \pavedb\files\LOCATION INDEX \pavedb\indexes\LOCATION
SELECT B
USE \pavedb\files\MODEL
ZAP
SELECT C
USE \pavedb\files\GEOSHO INDEX \pavedb\indexes\GEONDX

* for each active sid number in the location file
SELECT A
DO WHILE .NOT. EOF()
   * see if the most recent widening flag of this section, if it exists
   * in the database; ignore if it is 2
   SELECT C
   SEEK STR(A->SID_NO, 4)
   IF FOUND()
      DO WHILE SID_NO = A->SID_NO
         SKIP
      ENDDO
      SKIP -1
      IF WIDENFLG = 2
         IGNORE = .T.
      ELSE
         IGNORE = .F.
      ENDIF
   ELSE
      IGNORE = .T.
   ENDIF

   * continue if this section should not be ignored
   IF .NOT. IGNORE
      SELECT B
      APPEND BLANK

      * copy the sid number
      REPLACE SID_NO WITH A->SID_NO

      * copy the highway type
```

```
      REPLACE HWYTYPE WITH A->HWYPREFX

      * copy the district number
      REPLACE HWYDIST WITH A->HWYDIST

      * copy the county number
      REPLACE CNTYNUM WITH A->CNTYNUM
   ENDIF

   * skip to the next sid number in location file
   SELECT A
   SKIP
ENDDO

* close files
CLOSE ALL
```

Building Model File

Program Specification 4

Program Name: MODL_DIS.PRG

Purpose: To find the distress data for the model file.

Edit/Procedure Information:

1. For each section identification number in the model file, this program will try to find the data in the distress database file. If there is such section in the distress file, the SID number, year, annual cummulative 18KIP equivalent axle load value, area of distress value for patching, rutting, and alligator cracking, serviceability index mean value, and the pavement type are written to an intermediate file (DISTRESS.DAT).

2. With the intermediate file, the program will call an external Turbo Pascal program (MODL_R_B.PAS) to find the rho and beta values of the distress and psi curves. The Pascal program returns the values through another intermediate file (RHOBETA.DAT).

3. MODL_DIS.PRG then reads the rho and beta values from RHOBETA.DAT into an intermediate datbase file (TEMP_DIS.DBF). The values are in turn copied into the model file.

Input File(s):

1. Distress database file (DISTRESS.DBF) with index file (DISTRESS.NDX).

Intermediate and Input/Output File(s):

1. Temporary distress database file (TEMP_DIS.DBF).

2. Intermediate distress data file (DISTRESS.DAT).

3. Intermediate rho/beta value data file (RHOBETA.DAT).

Output File(s):

1. Model database file (MODEL.DBF).

Program Flow for Modl_Dis.Prg

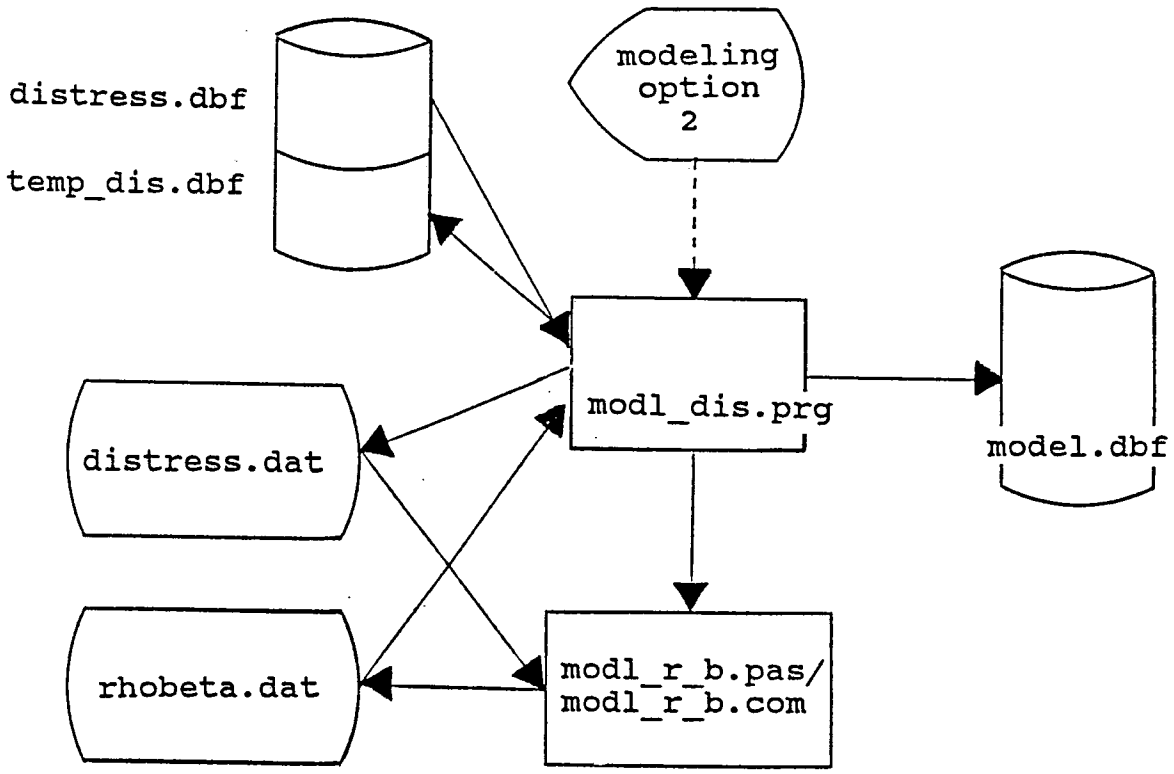


FIGURE 29

Building Model File

Program Specification 4
Program Listing

```
* File name:     MODL_DIS.PRG
* Program name: modl_dis
* Project 2456: Texas Flexible Pavement Database
* TAMU/TTI
* Written by:   Victor Wong
* Created on:   June 29, 1988
* Last updated: July 26, 1988
* Purpose:      To find the rho and beta values for rutting,
*               alligator cracking and psi.

* open files
SELECT A
USE \pavedb\files\MODEL
SELECT B
USE \pavedb\files\DISTRESS INDEX \pavedb\indexes\DISTRESS
SELECT C
USE \pavedb\files\TEMP_DIS
ZAP

* for each sid number in the model file
SELECT A
DO WHILE .NOT. EOF()
   * find the data in the distress database file
   SELECT B
   SEEK STR(A->SID_NO,4)

   * if it is found
   IF FOUND()
      * copy the data to a column arranged file
      COPY TO \pavedb\applicat\model\DISTRESS.DAT ;
         FIELDS SID_NO,YEAR,A18KEAL,PATCH, RUTT, ALLGCR,;
SIMEAN,PAVETYP WHILE SID_NO=A->SID_NO TYPE SDF

      * run the Turbo Pascal program to find the rho and beta
      RUN \pavedb\applicat\model\MODL_R_B

      * copy data to model file
      SELECT C
      APPEND FROM \pavedb\applicat\model\RHOBETA.DAT TYPE SDF
      SELECT A
      REPLACE PAVETYP WITH C->PAVETYP
      REPLACE ALLG_RHO WITH C->ALLG_RHO
      REPLACE ALLG_BETA WITH C->ALLG_BETA
      REPLACE RUTT_RHO WITH C->RUTT_RHO
      REPLACE RUTT_BETA WITH C->RUTT_BETA
      REPLACE PSI_RHO WITH C->PSI_RHO
      REPLACE PSI_BETA WITH C->PSI_BETA
```

```
   ENDIF

   * skip to the next sid number in model file
   SELECT A
   SKIP
ENDDO

* close files
CLOSE ALL
```

Building Model File

Program Specification 4
Program Listing

```
{*  File name:     MODL_R_B.PAS
 *  Program name: model_rho_beta
 *  Project 2456: Texas Flexible Pavement Database Conversion
 *  TAMU/TTI
 *  Written by:   Victor Wong
 *  Created on:   June 29, 1988
 *  Last updated: July 14, 1988
 *  Purpose:      To find the rho's and beta's of alligator
 *                cracking, rutting and PSI for the model file.
 *  Input file:   DISTRESS.DAT
 *  Output file:  RHOBETA.DAT
}

{>>> PROGRAM MODL_R_B <<<}
program model_rho_beta;

{>>> constant declaration <<<}
const
   blank = ' ';
   psi_initial = 4.5;
   psi_final = 1.5;

{>>> type declaration <<<}
{$I \pavedb\applicat\graph18k\declare}

{>>> variable declaration <<<}
var
   {--- variables for data processing ---}
   data_file:        text;

   {input data format}
   sid_no:           string[4];    {sid number      4N}
   year:             string[2];    {year            2N}
   a18keal:          string[11];   {18kip          11N}
   patching:         string[1];    {patching        1N}
   rutting:          string[1];    {rutting         1N}
   allgcrk:          string[1];    {alligator crack 1N}
   simean:           string[7];    {simean        7.5N}
   pavetyp:          string[2];    {pavement type   2N}

   {variables needed for conversion of data from strings into }
   {numnbers and variables needed for any numeric manipulation}
   {with the data}
   err_code,
   err_code1,                      {error code for string to}
   err_code2:        integer;      {number conversion}
   sid_no_value,                   {sid no value}
```

```
psi_value:          real;          {psi value}
patch_value,                       {patching value}
rutting_value,                     {rutting value}
allgcrk_value:      integer;       {alligator crack value}
patch_area,                        {patching area}
rutting_area,                      {rutting area}
allgcrk_area:       real;          {alligator crack area}
a18k1, a18k2,                      {if a18keal = aa bbb ccc ddd}
a18k3, a18k4,                      {then a18k1=aa,18k2=bbb,     }
a18k1_carry,                       {     81k3=ccc,18k4=ddd      }
a18k2_carry,
a18k3_carry,                       {carry from the digit group }
a18k4_carry,                       {for addition               }
accum_18k1,
accum_18k2,
accum_18k3,                        {accumulated 18k value by   }
accum_18k4:         integer;       {digit group                }
a18k1_zero,
a18k2_zero,
a18k3_zero,                        {flag saying if the digit   }
a18k4_zero:         boolean;       {group = 0                  }
a18keal1:           string[2];
a18keal2,
a18keal3,
a18keal4:           string[3];     {portions of the a18keal str}


{variables needed for finding rho's and beta's}
allg_rho, allg_beta,
rutt_rho, rutt_beta,
psi_rho, psi_beta:real;
psi,                               {psi value}
allg_y,
rutt_y,
psi_y:              y_array_type; {y value (allgcrk/rutt value)}    allg_x,
rutt_x,
psi_x,                             {x value (acccum. 18k value)}
allg_x1,
allg_x2,
allg_x3,
allg_x4,
rutt_x1,
rutt_x2,
rutt_x3,
rutt_x4,
psi_x1,
psi_x2,
psi_x3,
psi_x4:             x_array_type; {x value by digit group}

{--- other variables in the program ---}
c:                  char;
i:                  integer;
char_at_index:      char;
```

```
   index:              integer;
   found:              boolean;
   allg_count,
   rutt_count,
   psi_count:          integer;
   outfile:            text;

{>>> include files <<<}
{$I \pavedb\applicat\graph18k\rhobeta}
                        {file with procedure to calculate}
                        {rho and beta}

{>>> beginning of program <<<}
begin

   {--- open the distress data file ---}
   assign (data_file, '\pavedb\applicat\model\distress.dat');
   reset (data_file);

   {--- initialize some variables---}
   accum_18k1:= 0;
   accum_18k2:= 0;
   accum_18k3:= 0;
   accum_18k4:= 0;


   readln (data_file, sid_no, year, a18keal, patching, rutting,
        allgcrk, simean, pavetyp);
   allg_count:= 0;     {initialize the count of valid data to 0}
   rutt_count:= 0;     {initialize the count of valid data to 0}
   psi_count:= 0;      {initialize the count of valid data to 0}
   patch_area:=0.005;{initialize the patch area to 0.0}

   {while not eof}
   {read data, accumulate 18kip value }
   {and store valid distress or psi value}
   while not eof(data_file) do
      begin
         {convert the a81keal string value to 3 integer values}
         {so that the concatenation of a18k1, a18k2 and a18k3 }
         {will give the whole 18keal value                    }
         {if the err_code is non-zero, it means that digits   }
         {are missing at those positions and it is the same as}
         {having zeros there                                  }
         {the positions of leading zeros are indicated by the }
         {a18kX_zero flag                                     }

         {split a18keal value string into 4 sections }
         {(xx,xxx,xxx,xxx)}
         a18keal1:= copy(a18keal,1,2);
         a18keal2:= copy(a18keal,3,3);
         a18keal3:= copy(a18keal,6,3);
         a18keal4:= copy(a18keal,9,3);
```

```
{delete leading blank spaces of a18kealx string}
{and store convert the string to integer a18kx values}
index:= 1;
char_at_index:= copy (a18keal1,index,1);
while (char_at_index = blank) and (index < 2) do
begin
   index:= index + 1;
   char_at_index:= copy (a18keal1,index,1);
end;
if index < 2 then
   val (copy(a18keal1,index,3-index),a18k1,err_code)
else if char_at_index <> blank then
   val (copy (a18keal1,index,3-index),a18k1,err_code)
else
   begin
   a18k1:= 0;
   a18k1_zero:= true;
   end;

index:= 1;
char_at_index:= copy (a18keal2,index,1);
while (char_at_index = blank) and (index < 3) do
begin
   index:= index + 1;
   char_at_index:= copy (a18keal2,index,1);
end;
if index < 3 then
   val (copy(a18keal2,index,4-index),a18k2,err_code)
else if char_at_index <> blank then
   val (copy (a18keal2,index,4-index),a18k2,err_code)
else
   begin
   a18k2:= 0;
   a18k2_zero:= true;
   end;

index:= 1;
char_at_index:= copy (a18keal3,index,1);
while (char_at_index = blank) and (index < 3) do
begin
   index:= index + 1;
   char_at_index:= copy (a18keal3,index,1);
end;
if index < 3 then
   val (copy(a18keal3,index,4-index),a18k3,err_code)
else if char_at_index <> blank then
   val (copy (a18keal3,index,4-index),a18k3,err_code)
else
   begin
   a18k2:= 0;
   a18k3_zero:= true;
   end;

index:= 1;
```

```
char_at_index:= copy (a18keal4,index,1);
while (char_at_index = blank) and (index < 3) do
begin
   index:= index + 1;
   char_at_index:= copy (a18keal4,index,1);
end;
if index < 3 then
   val (copy(a18keal4,index,4-index),a18k4,err_code)
else if char_at_index <> blank then
   val (copy (a18keal4,index,4-index),a18k4,err_code)
else
   begin
   a18k4:= 0;
   a18k4_zero:= true;
   end;

{if the a18keal value is not zero continue processing}
if not (a18k1_zero and a18k2_zero and a18k3_zero and
        a18k4_zero) then
begin
   {increment the accumulated 18kip value}
   accum_18k4:= accum_18k4 + a18k4;
   a18k4_carry:= accum_18k4 div 1000;
   accum_18k4:= accum_18k4 mod 1000;
   accum_18k3:= accum_18k3 + a18k3 + a18k4_carry;
   a18k3_carry:= accum_18k3 div 1000;
   accum_18k3:= accum_18k3 mod 1000;
   accum_18k2:= accum_18k2 + a18k2 + a18k3_carry;
   a18k2_carry:= accum_18k2 div 1000;
   accum_18k2:= accum_18k2 mod 1000;
   accum_18k1:= accum_18k1 + a18k1 + a18k2_carry;


   {--- for each distress type      ---}
   {--- do the necessary processing ---}
   {alligator cracking}
   {convert alligator crack data from string to}
   {integer}
   val (allgcrk,allgcrk_value,err_code1);

   {convert patching data from string to integer}
   val (patching,patch_value,err_code2);

   {if the conversion is successful}
   if (err_code1 = 0) and (err_code2 = 0) then
      begin
         {check if the value is valid i.e. <> 9}
         if (allgcrk_value <> 9) then
         begin
            {increment valid data count}
            allg_count:= allg_count + 1;

            {determine the percentage of alligator}
            {cracking area}
```

```
   case allgcrk_value of
   0: allgcrk_area:= 0.005;
   1: allgcrk_area:= 0.055;
   2: allgcrk_area:= 0.305;
   3: allgcrk_area:= 0.50;
   end;

   {determine the percentage of patching}
   {area}
   if patch_value <> 9 then
      case patch_value of
      0: patch_area:= 0.005;
      1: patch_area:= 0.055;
      2: patch_area:= 0.305;
      3: patch_area:= 0.50;
      end;

   {take the sum of the two area %}
   {as the value to be plotted    }
   allg_y[allg_count] := (allgcrk_area +
                          patch_area);
   if allg_y[allg_count] >= 1.0 then
      allg_y[allg_count]:= 0.985;

   {store the new accumulated 18kip value}
   {in x arrays}
   allg_x1[allg_count]:= accum_18k1;
   allg_x2[allg_count]:= accum_18k2;
   allg_x3[allg_count]:= accum_18k3;
   allg_x4[allg_count]:= accum_18k4;
end
else
begin
   if err_code1 <> 0 then
   begin
      gotoxy (22,14);
      write ('***WARNING***');
      write (' bad data encountered');
      gotoxy (22,15);
      write ('              ');
      write ('INVALID CODE FOR ALLIGATOR ');
      write ('CRACKING ');
      gotoxy (22,17);
      write ('Still processing.');
      write ('  Please wait ...');
   end;
   if err_code2 <> 0 then
   begin
      gotoxy (22,14);
      write ('***WARNING***');
      write (' bad data encountered');
      gotoxy (22,15);
      write ('              ');
      write ('INVALID CODE FOR PATCHING');
```

```
            write ('           ');
            gotoxy (22,17);
            write ('Still processing.');
            write ('  Please wait ...');
         end;
      end;
   end;

{rutting}
{convert rutting data from string to integer}
val (rutting,rutting_value,err_code1);

{convert patching data from string to integer}
val (patching,patch_value,err_code2);

{if the conversion is successful}
if (err_code1 = 0) and (err_code2 = 0) then
begin
   {check if the value is valid i.e. <> 9}
   if (rutting_value <> 9) then
   begin
      {increment valid data count}
      rutt_count:= rutt_count + 1;

      {determine the percentage of rutting}
      {area}
      case rutting_value of
      0: rutting_area:= 0.005;
      1: rutting_area:= 0.13;
      2: rutting_area:= 0.38;
      3: rutting_area:= 0.50;
      end;

      {determine the percentage of patching}
      {area}
      if patch_value <> 9 then
         case patch_value of
         0: patch_area:= 0.005;
         1: patch_area:= 0.055;
         2: patch_area:= 0.305;
         3: patch_area:= 0.5;
         end;

      {take the sum of the two area %}
      {as the value to be plotted    }
      rutt_y[rutt_count]:= (rutting_area +
                            patch_area);
      if rutt_y[rutt_count] >= 1.0 then
         rutt_y[rutt_count]:= 0.985;

      {store the new accumulated 18kip value}
      {in x arrays}
      rutt_x1[rutt_count]:= accum_18k1;
      rutt_x2[rutt_count]:= accum_18k2;
```

```
      rutt_x3[rutt_count]:= accum_18k3;
      rutt_x4[rutt_count]:= accum_18k4;
   end
   else
   begin
      if err_code1 <> 0 then
      begin
         gotoxy (22,14);
         write ('***WARNING*** ');
         write ('bad data encountered');
         gotoxy (22,15);
         write ('               ');
         write ('INVALID CODE FOR RUTTING   ');
         gotoxy (22,17);
         write ('Still processing.  ');
         write ('Please wait ...');
      end;
      if err_code2 <> 0 then
      begin
         gotoxy (22,14);
         write ('***WARNING*** ');
         write ('bad data encountered');
         gotoxy (22,15);
         write ('               ');
         write ('INVALID CODE FOR PATCHING   ');
         gotoxy (22,17);
         write ('Still processing.  ');
         write ('Please wait ...');
      end;
   end;
end;

{psi}
{convert psi data from string to real}
val (simean,psi_value,err_code);

{if the conversion is successful}
if (err_code = 0) then
begin
   if psi_value <> 9.99999 then
   begin
      psi_count:= psi_count + 1;

      {store the psi value in the psi array}
      if psi_value = 0.0 then
         psi[psi_count]:= 0.0005
      else
        psi[psi_count]:= psi_value;

      {transform the psi value to a distress value}
      psi_y[psi_count]:= ((psi_initial -
        psi[psi_count]) / (psi_initial-
        psi_final))/10;
```

```
            {store the new accumulated 18kip value}
            {in x arrays}
            psi_x1[psi_count]:= accum_18k1;
            psi_x2[psi_count]:= accum_18k2;
            psi_x3[psi_count]:= accum_18k3;
            psi_x4[psi_count]:= accum_18k4;
         end
         else
         begin
            if err_code <> 0 then
            begin
               gotoxy (22,14);
               write ('***WARNING*** ');
               write ('bad data encountered');
               gotoxy (22,15);
               write ('                ');
               write ('INVALID PSI VALUE ');
               gotoxy (22,17);
               write ('Still processing.  ');
               write ('Please wait ...');
            end;
         end;
      end;

   end; {if}

   if not eof(data_file) then
      readln(data_file, sid_no, year, a18keal, patching,
         rutting, allgcrk, simean, pavetyp);       end;

{find a suitable accumulated 18kip factor for the x axis}
{on the graph}
{NOTE: 18 kip values are taken as the 3 most significant}
{      digits i.e. a number 1,234,567 will be expressed }
{      as 123 x 10^4 }
if accum_18k3=0 then
   begin
   for i:= 1 to allg_count do allg_x[i]:= allg_x4[i];
   for i:= 1 to rutt_count do rutt_x[i]:= rutt_x4[i];
   for i:= 1 to psi_count do psi_x[i]:= psi_x4[i];
   end
else if accum_18k2 = 0 then
   begin
   if (a18k3 div 100) > 0 then
      begin
      for i:= 1 to allg_count do allg_x[i]:= allg_x3[i];
      for i:= 1 to rutt_count do rutt_x[i]:= rutt_x3[i];
      for i:= 1 to psi_count do psi_x[i]:= psi_x3[i];
      end
   else if (a18k3 div 10) > 0 then
      begin
      for i:= 1 to allg_count do
         allg_x[i]:= allg_x3[i]*10 + allg_x4[i] div 100;
      for i:= 1 to rutt_count do
```

```
            allg_x[i]:= rutt_x3[i]*10 + rutt_x4[i] div 100;
         for i:= 1 to psi_count do
            allg_x[i]:= psi_x3[i]*10 + psi_x4[i] div 100;
         end
      else
         begin
         for i:= 1 to allg_count do
            allg_x[i]:= allg_x3[i]*100 + allg_x4[i] div 10;
         for i:= 1 to rutt_count do
            allg_x[i]:= rutt_x3[i]*100 + rutt_x4[i] div 10;
         for i:= 1 to psi_count do
            allg_x[i]:= psi_x3[i]*100 + psi_x4[i] div 10;
         end
      end
   else if accum_18k1 = 0 then
      begin
      if (a18k2 div 100) > 0 then
         begin
         for i:= 1 to allg_count do allg_x[i]:= allg_x2[i];
         for i:= 1 to rutt_count do rutt_x[i]:= rutt_x2[i];
         for i:= 1 to psi_count do psi_x[i]:= psi_x2[i];
         end
      else if (a18k2 div 10 ) > 0 then
         begin
         for i:= 1 to allg_count do
            allg_x[i]:= allg_x2[i]*10 + allg_x3[i] div 100;
         for i:= 1 to rutt_count do
            allg_x[i]:= rutt_x2[i]*10 + rutt_x3[i] div 100;
         for i:= 1 to psi_count do
            allg_x[i]:= psi_x2[i]*10 + psi_x3[i] div 100;
         end
      else
         begin
         for i:= 1 to allg_count do
            allg_x[i]:= allg_x2[i]*100 + allg_x3[i] div 10;
         for i:= 1 to rutt_count do
            rutt_x[i]:= rutt_x2[i]*100 + rutt_x3[i] div 10;
         for i:= 1 to psi_count do
            psi_x[i]:= psi_x2[i]*100 + psi_x3[i] div 10;
         end
      end
   else
      begin
      if (a18k1 div 10) > 0 then
         begin
         for i:= 1 to allg_count do
            allg_x[i]:= allg_x1[i]*10 + allg_x2[i] div 100;
         for i:= 1 to rutt_count do
            rutt_x[i]:= rutt_x1[i]*10 + rutt_x2[i] div 100;
         for i:= 1 to psi_count do
            psi_x[i]:= psi_x1[i]*10 + psi_x2[i] div 100;
         end
      else
         begin
```

```
            for i:= 1 to allg_count do
                allg_x[i]:= allg_x1[i]*100 + allg_x2[i] div 10;
            for i:= 1 to rutt_count do
                rutt_x[i]:= rutt_x1[i]*100 + rutt_x2[i] div 10;
            for i:= 1 to psi_count do
                psi_x[i]:= psi_x1[i]*100 + psi_x2[i] div 10;
            end
        end;

    Find_rho_beta (allg_x,allg_y,allg_count,allg_rho,
                   allg_beta);
    Find_rho_beta (rutt_x,rutt_y,rutt_count,rutt_rho,
rutt_beta);
    Find_rho_beta (psi_x,psi_y,psi_count,psi_rho,psi_beta);

    assign (outfile, '\pavedb\applicat\model\rhobeta.dat');
    rewrite (outfile);
    writeln (outfile, sid_no,pavetyp,allg_rho:7:4,
allg_beta:7:4,rutt_rho:7:4,rutt_beta:7:4,
psi_rho:7:4,psi_beta:7:4);
    close(outfile);
    close (data_file);
{>>> ending of program <<<}
end.
```

Building Model File

Program Specification 5

Program Name: MODL_LAY.PRG

Purpose: To find the layer data for the model file.

Edit/Procedure Information:

1. For each section identification number in the model file, the program finds the section in the layer file. If it is found, the data needed to find the the total thickness of the most recent same material base layers is copied to an intermediate file (BASETHCK.DAT).

2. A Turbo Pascal program (BASETHCK.PAS) will take the data in BASETHCK.DAT and find the most recent base layer. Then it will keep adding the thickness of other base layers that have the same material code number. It does so by backtracking through the layers starting with the highest layer number. It stops when it encounter a base layer with a different material than the latest base layer, or the earliest layer is encountered.

3. The total base layer thickness is then passed back to the MODL_LAY.PRG program by writing the result in the intermediate file (BASETHCK.DAT). The type of base material is also passed in the file. The dBASE program (MODL_LAY.PRG) will copy the result to a temporary database file (TEMP_LAY.DBF) and then to the model file.

Input File(s):

1. Layer database file (LAYER.DBF) with index file (LAYER.NDX).

Intermediate and Input/Output File(s):

1. Temporary layer database file (TEMP_LAY.DBF).

2. Intermediate base thickness data file (BASTHCK.DAT).

Output File(s):

1. Model database file (MODEL.DBF).

Building Model File

Program Specification 5

File Layout(s):

1. LAYER.DBF (See Appendix A)
2. TEMP_LAY.DBF (Refer to Intermediate and Input/Ouput File Layout 2 in Program Specification 1.)
3. BASEIHCK.DAT (Refer to Intermediate and Input/Output File Layout 5 in Program Specification 1.)
4. MODEL.DBF (See Appendix A)

Programs Called:

BASEIHCK.PAS

Program Flow for Modl_Lay.Prg

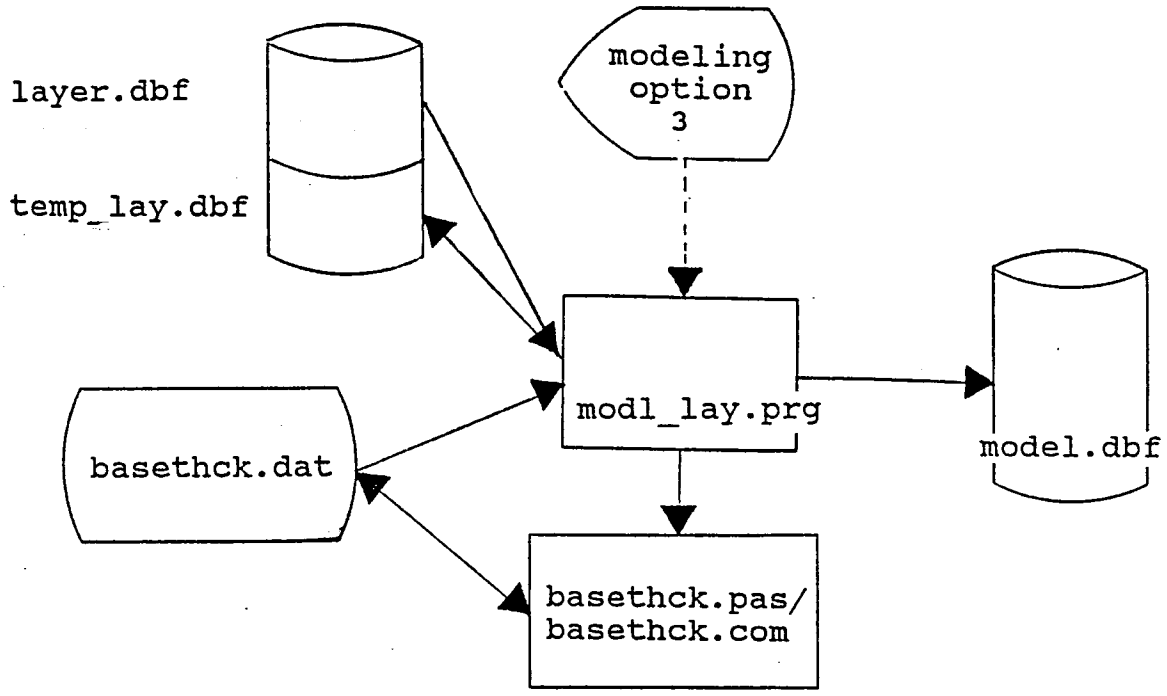


FIGURE 30

Building Model File

Program Specification 5
Program Listing

```
* File name:    MODL_LAY.PRG
* Program name: modl_lay
* Project 2456: Texas Flexible Pavement Database
* TAMU/TTI
* Written by:   Victor Wong
* Created on:   June 29, 1988
* Last updated: July 26, 1988
* Purpose:      To find the layer data for the model file.

* open files
SELECT A
USE \pavedb\files\MODEL
SELECT B
USE \pavedb\files\LAYER INDEX \pavedb\indexes\LAYNDX
SELECT C
USE \pavedb\files\TEMP_LAY

* fo each sid number in the model file
SELECT A
DO WHILE .NOT. EOF()
   * find the sid number in layer file
   SELECT B
   SEEK STR(A->SID_NO,4)

   * if found
   IF FOUND()
      * copy the data to a column arranged file
      COPY TO \pavedb\applicat\model\BASETHCK.DAT ;
         FIELDS SID_NO,STRUCNUM,LAYNUM,LAYDESC,LAYMATCL,CENTTHK ;
WHILE SID_NO=A->SID_NO TYPE SDF

      * run a Turbo Pascal program to find the base thickness       RUN
\pavedb\applicat\model\BASETHCK

      CLEAR
      @10,19 SAY "Retrieving Layer Data for Model File..."

      * copy the result to a temporary file
      SELECT C
      ZAP
      APPEND FROM \pavedb\applicat\model\BASETHCK.DAT TYPE SDF

      * copy the result to the model file
      SELECT A
      REPLACE BASETHCK WITH C->BASETHCK
      REPLACE BASETYP WITH C->BASETYPE
   ENDIF
```

```
   * next sid number in the model file
   SELECT A
   SKIP
ENDDO

* close files
CLOSE ALL
```

Building Model File

Program Specification 5
Program Listing

```
{* File name:      BASETHCK.PAS
 * Program name:   find_base_thickness
 * Project 2456:   Texas Flexible Pavement Database
 * TAMU/TTI
 * Written by:     Victor Wong
 * Created on:     July 13, 1988
 * Last updated:   July 26, 1988
 * Purpose:        To find the total thickness of the most recent
 *                 base material.
 * Input file(s): basethck.dat
 * Output file(s):basethck.dat
}

program find_base_thickness;

type
   dataitem = record
      sid_no:          string[4];
      strunum:         string[2];
      laynum:          string[2];
      laydesc:         string[2];
      laymatcl:        string[2];
      centthk:         string[5];
   end;

var
   datafile:          text;
   base_thick,
   total_base_thick: real;
   err_code,
   no_of_records,
   index:             integer;
   data:              array[1..30] of dataitem;
   base_type,
   previous_laynum:  string[2];
   exit_loop,
   most_recent_base: boolean;

begin
   assign (datafile, '\pavedb\applicat\model\basethck.dat');

   {---read in data needed for the solution}
   reset (datafile);
   index:= 0;
   while not eof(datafile) do
   begin
      index:= index + 1;
```

```
      with data[index] do
      begin
         readln (datafile, sid_no,strunum,laynum,laydesc,
                           laymatcl, centthk);
         if centthk[1] = ' ' then centthk[1]:= '0';
      end;
   end;
   no_of_records:= index;
   close (datafile);

   {---find the total thickness of the most base material}
   {initializations}
   previous_laynum:= ' 0';
   most_recent_base:= true;
   exit_loop:= false;
   index:= no_of_records;
   total_base_thick:= 0.0;

   {for all the layers starting backwards}
   while (index >= 1) and (not exit_loop) do
   with data[index] do
   begin
      {if the layer number is different from the previous one}       if
laynum <> previous_laynum then
      begin
         {if the layer is a base}
         if laydesc = ' 5' then
         begin
            {if the base layer is the most recent one}
            if most_recent_base then
            begin
               {make note of the base type}
               base_type:= laymatcl;
               {the next base layer will not be the most recent}
               most_recent_base:= false;
               {increment the total base thickness}
               val (centthk, base_thick, err_code);
               total_base_thick:= total_base_thick + base_thick;
end

            {if the base layer is not the most recent one}             else
            begin
               {if the base material is the same as the most
                recent one}
               if laymatcl = base_type then
               begin
                  {increment the total base thickness}
                  val (centthk, base_thick, err_code);
                  total_base_thick:= total_base_thick +
               base_thick;
             end

             {if the base material is different from the most
              recent one}
```

```
            else
               {exit the loop}
               exit_loop:= true;
            end;
         end;
      end;

      {increment the index}
      previous_laynum:= laynum;
      index:= index - 1;
   end;

   {output the solution}
   rewrite (datafile);
   writeln (datafile, data[1].sid_no, base_type,
total_base_thick:7:2);
   close (datafile);
end.
```

Building Model File

Program Specification 6

Program Name: MODL_WEA.PRG

Purpose: To find the weather data for the model file.

Edit/Procedure Information:

1. For each section identification number in the model file, the program determines if the record for the county in which the section resides exists in the weather database file (WEATHER.DBF). If so, the program finds the annual total freeze thaw cycle values and the mean precipitation values. This is done by summing the monthly values for that county.

2. The result is copied to the model file.

Input File(s):

1. Weather database file (WEATHER.DBF) with index file (WEATHER.NDX).

Output File(s):

1. Model database file (MODEL.DBF).

File Layout(s):

1. WEATHER.DBF (See Appendix A)

2. MODEL.DBF (See Appendix A)

Program Flow for Modl_Wea.Prg

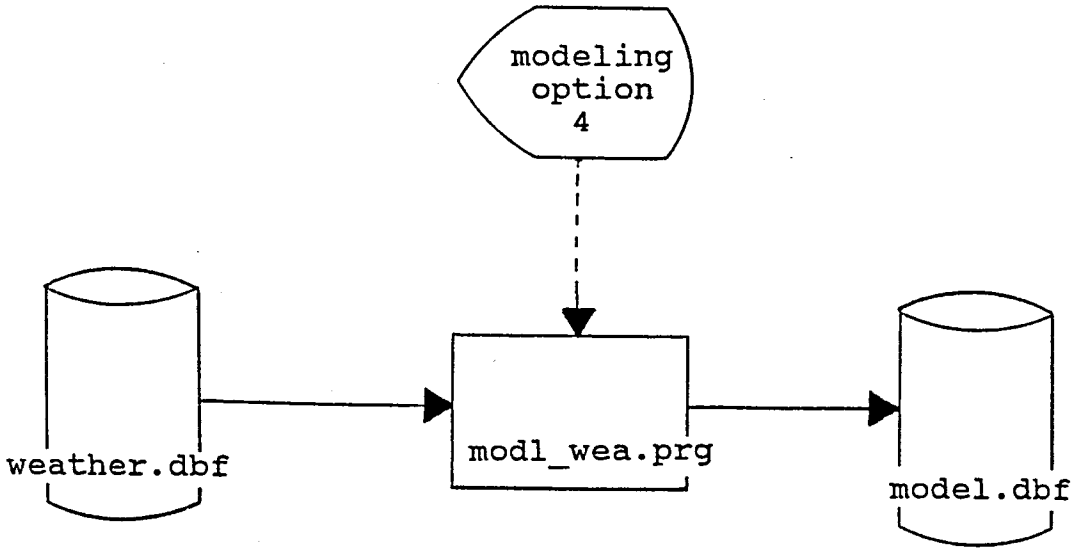


FIGURE 31

Building Model File

Program Specification 6
Program Listing

```
* File name:     MODL_WEA.PRG
* Program name: modl_wea
* Project 2456: Texas Flexible Pavement Database
* TAMU/TTI
* Written by:   Victor Wong
* Created on:   June 29, 1988
* Last updated: July 26, 1988
* Purpose:      To find the weather data for the model file.

* open files
SELECT A
USE \pavedb\files\MODEL
SELECT B
USE \pavedb\files\WEATHER INDEX \pavedb\indexes\WEATHER

* for each active sid number in the model file
SELECT A
DO WHILE .NOT. EOF()
   * find the data in the weather database file
   SELECT B
   SEEK STR(A->CNTYNUM,3)

   * copy the data to the model database file
   IF FOUND()
      * calculate the annual total freeze thaw cycles and
      * mean precipitation values
      TOT_F_T = 0
      TOT_M_P = 0
      DO WHILE CNTYNUM = A->CNTYNUM
         TOT_F_T = TOT_F_T + TFTCMN
         TOT_M_P = TOT_M_P + PRECMN
         SKIP
      ENDDO
      REPLACE A->TFTCMN WITH TOT_F_T
      REPLACE A->PRECMN WITH TOT_M_P
   ENDIF

   * skip to the next sid number in model file
   SELECT A
   SKIP
ENDDO

* close files
CLOSE ALL
```

Building Model File

Program Specification 7

Program Name: MODL_ENV.PRG

Purpose: To find the environment data for the model file.

Edit/Procedure Information:

1. For each section identification number in the model file, the program determines if the record for the county in which the section resides exists in the environment database file (ENV.DBF). If so, the program copies the Thornthwaite index value to the model file.

Input File(s):

1. Environment database file (ENV.DBF) with index file (ENV.NDX).

Output File(s):

1. Model database file (MODEL.DBF).

File Layout(s):

1. ENV.DBF (See Appendix A)
2. MODEL.DBF (See Appendix A)

Program Flow for Modl_Env.Prg.

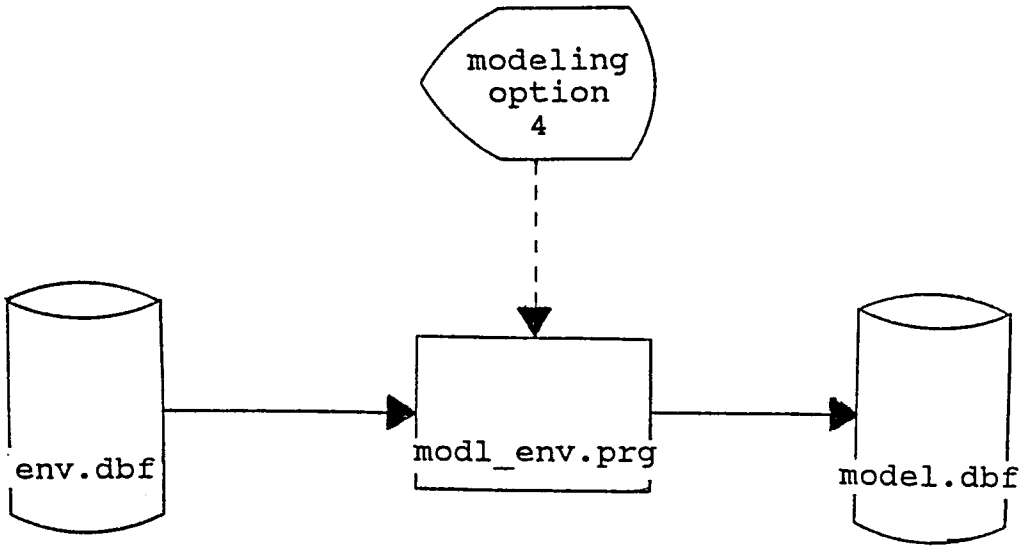


FIGURE 32

Building Model File

Program Specification 7
Program Listing

```
* File name:     MODL_ENV.PRG
* Program name: modl_env
* Project 2456: Texas Flexible Pavement Database
* TAMU/TTI
* Written by:   Victor Wong
* Created on:   June 29, 1988
* Last updated: July 26, 1988
* Purpose:      To find the environmental data for the model
*               file.

* open files
SELECT A
USE \pavedb\files\MODEL
SELECT B
USE \pavedb\files\ENV INDEX \pavedb\indexes\ENV

* for each sid number in the model file
SELECT A
DO WHILE .NOT. EOF()
   * find the data in the env database file
   SELECT B
   SEEK A->CNTYNUM

   * copy the data to the model database file
   IF FOUND()
      * copy the mean thornthwaite index
      REPLACE A->THORNMN WITH THORNMN
   ENDIF

   * skip to the next sid number in model file
   SELECT A
   SKIP
ENDDO

* close files
CLOSE ALL
```

Building Model File

Program Specification 8

Program Name: MODL_TRF.PRG

Purpose:

To find the traffic data for the model file.

Edit/Procedure Information:

1. For each section identification number in the model file, the program determines if the section exists in the traffic database file (TRAFFIC.DBF). If so, the program finds the latest average daily traffic (ADT) value and copies it to the model file.

Input File(s):

1. Traffic database file (TRAFFIC.DBF) with index file (TRAFFIC.NDX).

Output File(s):

1. Model database file (MODEL.DBF).

File Layout(s):

1. TRAFFIC.DBF (See Appendix A)
2. MODEL.DBF (See Appendix A)

Program Flow for Modl_Trf.Prg.

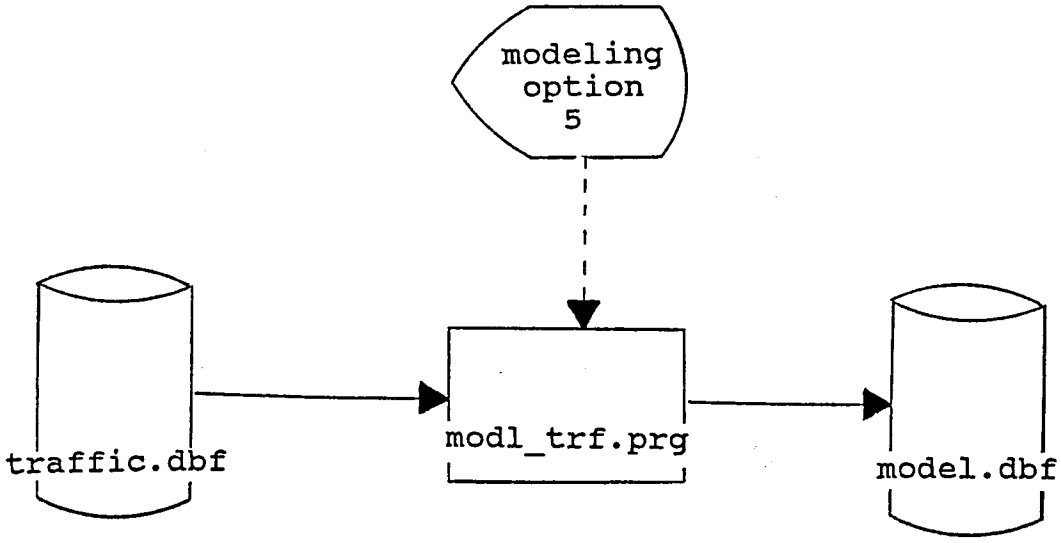


FIGURE 33

Building Model File

Program Specification 8
Program Listing

```
* File name:    MODL_TRF.PRG
* Program name: modl_trf
* Project 2456: Texas Flexible Pavement Database
* TAMU/TTI
* Written by:   Victor Wong
* Created on:   June 29, 1988
* Last updated: July 26, 1988
* Purpose:      To find the traffic data for the model file.

* open files
SELECT A
USE \pavedb\files\MODEL
SELECT B
USE \pavedb\files\TRAFFIC INDEX \pavedb\indexes\TRAFFIC

* for each active sid number in the location file
SELECT A
DO WHILE .NOT. EOF()
   * find the data in the traffic database file
   SELECT B
   SEEK STR(A->SID_NO,4)

   * copy the data to the model database file
   IF FOUND()
      * copy the most recent one way ADT value
      DO WHILE SID_NO = A->SID_NO
         SKIP
      ENDDO
      SKIP -1
      REPLACE A->CURADT WITH AADT1WAY
   ENDIF

   * skip to the next sid number in model file
   SELECT A
   SKIP
ENDDO

* close files
CLOSE ALL
```

Building Model File

Program Specification 9

Program Name: MODL_SUB.PRG

Purpose: To find the subgrade data for the model file.

Edit/Procedure Information:

1. For each section identification number in the model file, the program determines if the section exists in the subgrade database file (SUBGRADE.DBF). If so, the program finds the subgrade layer in the layer database file (LAYER.DBF) and copies the layer material code number for the subgrade to the model file. The percentage passing 200 seive value in the subgrade database file is also copied to the model file.

Input File(s):

1. Subgrade database file (SUBGRADE.DBF) with index file (SUBGNDX.NDX).

Output File(s):

1. Model database file (MODEL.DBF).

File Layout(s):

1. SUBGRADE.DBF (See Appendix A)
2. MODEL.DBF (See Appendix A)

Program Flow for Modl_Sub.Prg.

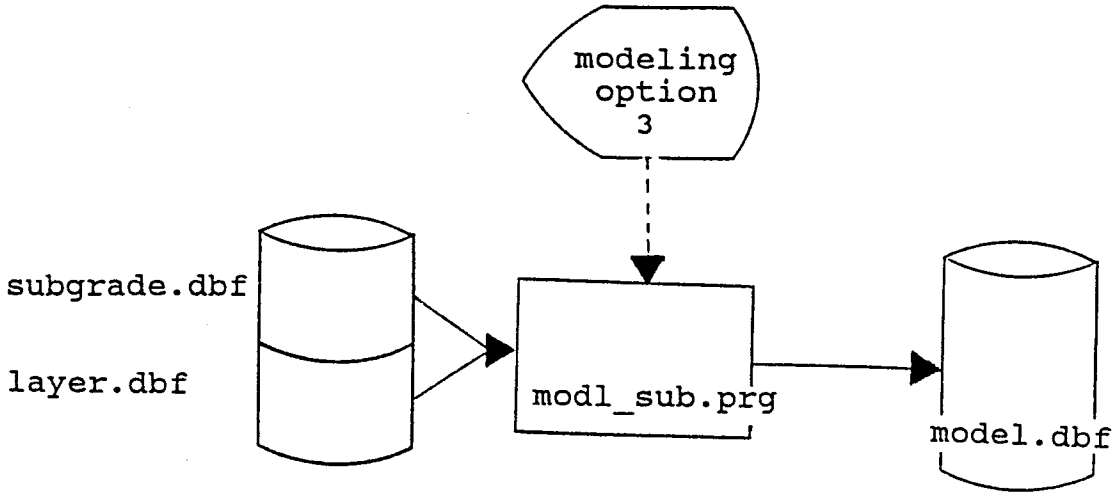


FIGURE 34

Building Model File

Program Specification 9
Program Listing

```
* File name:     MODL_SUB.PRG
* Program name: modl_sub
* Project 2456: Texas Flexible Pavement Database
* TAMU/TTI
* Written by:   Victor Wong
* Created on:   June 29, 1988
* Last updated: July 26, 1988
* Purpose:      To find the subgrade data for the model file.

* open files
SELECT A
USE \pavedb\files\MODEL
SELECT B
USE \pavedb\files\SUBGRADE INDEX \pavedb\indexes\SUBGNDX
SELECT C
USE \pavedb\files\LAYER INDEX \pavedb\indexes\LAYNDX

* for each sid number in the model file
SELECT A
DO WHILE .NOT. EOF()
   * find the data in the subgrade database file
   SELECT B
   SEEK STR(A->SID_NO,4)

   * copy the data to the model database file
   IF FOUND()
      * copy the subgrade material type from layer file
      SELECT C
      SEEK STR(B->SID_NO,4)+STR(B->STRUCNUM,2)+STR(B->LAYNUM,2) IF FOUND()
         REPLACE A->SUBGRDTY WITH LAYMATCL
      ENDIF

      * copy the percent passing 200 seive
      SELECT B
      REPLACE A->PPSV200 WITH PPSV200

      * copy the liquid limit
      REPLACE A->LIQLIM WITH LIQLIM
   ENDIF

   * skip to the next sid number in model file
   SELECT A
   SKIP
ENDDO

* close files
CLOSE ALL
```

Building Model File

Program Specification 10

Program Name: MODL_SHO.PRG

Purpose: To find the geometric shoulder data for the model file.

Edit/Procedure Information:

1. For each section identification number in the model file, the program determines if the section exists in the geometric shoulder database file (GEOSHO.DBF). If so, the program finds the latest shoulder layer and copies the shoulder type value to the model file.

Input File(s):

1. Geometric shoulder database file (GEOSHO.DBF) with index file (GEONDX.NDX).

Output File(s):

1. Model database file (MODEL.DBF).

File Layout(s):

1. GEOSHO.DBF (Appendix A.)
2. MODEL.DBF (Appendix A.)

Program Flow for Modl_Sho.Prg.

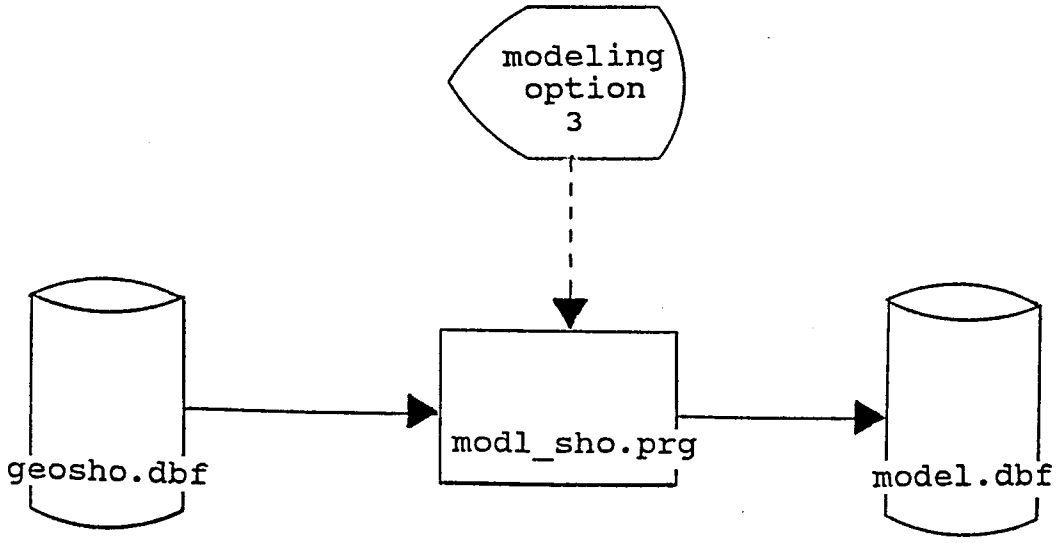


FIGURE 35

Building Model File

Program Specification 10
Program Listing

```
* File name:    MODL_SHO.PRG
* Program name: modl_sho
* Project 2456: Texas Flexible Pavement Database
* TAMU/TTI
* Written by:   Victor Wong
* Created on:   June 29, 1988
* Last updated: July 26, 1988
* Purpose:      To find the shoulder data for the model file.

* open files
SELECT A
USE \pavedb\files\MODEL
SELECT B
USE \pavedb\files\GEOSHO INDEX \pavedb\indexes\GEONDX

* for each sid number in the model file
SELECT A
DO WHILE .NOT. EOF()
   * find the data in the shoulder database file
   SELECT B
   SEEK STR(A->SID_NO,4)

   * copy the data to the model database file
   IF FOUND()
      * go to the most recent surface
      DO WHILE SID_NO = A->SID_NO
         SKIP
      ENDDO
      SKIP -1

      * copy the shoulder surface type
      REPLACE A->SHOSFTYP WITH SHOSFTYP
   ENDIF

   * skip to the next sid number in model file
   SELECT A
   SKIP
ENDDO

* close files
CLOSE ALL
```

Building Model File

Program Specification 11

Program Name: MODL_SUF.PRG

Purpose: To find the surface data for the model file.

Edit/Procedure Information:

1. For each section identification number in the model file, the program determines if the section exists in the surface database file (SURFACE.DBF). If so, the program finds the surface layers in the layer database file (LAYER.DBF). The program then adds up the center thickness of the layers. After all the surface layers for a SID have been processed, the program copies the total surface thickness value to the model file and begins processing the next SID.

Input File(s):

1. Surface database file (SURFACE.DBF) with index file (SURFNDX.NDX).
2. Layer database file (LAYER.DBF) with index file (LAYNDX.NDX).

Output File(s):

1. Model database file (MODEL.DBF).

File Layout(s):

1. SURFACE.DBF (See Appendix A.)
2. LAYER.DBF (See Appendix A.)
3. MODEL.DBF (See Appendix A.)

Program Flow for Modl_Suf.Prg.

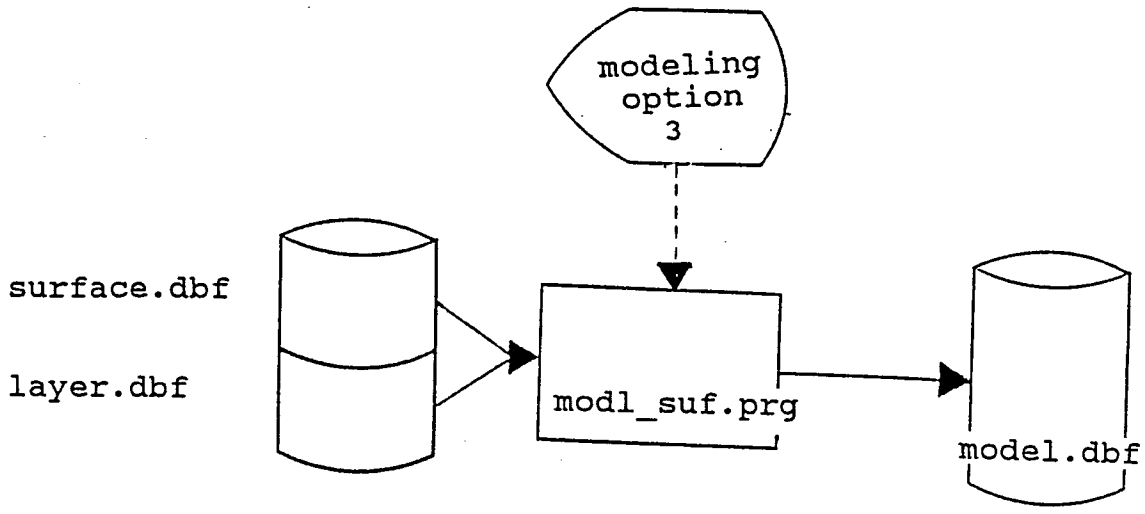


FIGURE 36

Building Model File

Program Specification 11
Program Listing

```
* File name:    MODL_SUF.PRG
* Program name: modl_suf
* Project 2456: Texas Flexible Pavement Database
* TAMU/TTI
* Written by:   Victor Wong
* Created on:   June 29, 1988
* Last updated: July 26, 1988
* Purpose:      To find the surface data for the model file.

* open files
SELECT A
USE \pavedb\files\MODEL
SELECT B
USE \pavedb\files\SURFACE INDEX \pavedb\indexes\SURFNDX
SELECT C
USE \pavedb\files\LAYER INDEX \pavedb\indexes\LAYNDX

* for each sid number in the model file
SELECT A
DO WHILE .NOT. EOF()
   * find the data in the surface database file
   SELECT B
   SEEK STR(A->SID_NO,4)

   * copy the data to the model database file
   IF FOUND()
      * copy the total thickness of surface layers from layer
      * file
      TOT_S_T = 0
      PREV_LAY = 0
      DO WHILE SID_NO = A->SID_NO
         SELECT C
         SEEK STR(B->SID_NO,4)+STR(B->STRUCNUM,2)+ ;
              STR(B->LAYNUM,2)
         IF FOUND() .AND. (CENTIHK <> 99.99) .AND.;
            (LAYNUM <> PREV_LAY)
            TOT_S_T = TOT_S_T + CENTIHK
            PREV_LAY = LAYNUM
         ENDIF
         SELECT B
         SKIP
      ENDDO
      REPLACE A->SURFTHCK WITH TOT_S_T

      * copy the percent asphalt for most recent surface layer       SELECT B
      SKIP -1
      REPLACE A->ASAPPLRT WITH ASAPPLRT
```

```
   ENDIF

   * skip to the next sid number in model file
   SELECT A
   SKIP
ENDDO

* close files
CLOSE ALL
```

Building Model File

Program Specification 12

Program Name: MODL_TMP.PRG

Purpose: To find the district temperature data for the model file.

Edit/Procedure Information:

1. For each section identification number in the model file, the program determines if the district in which the section resides exists in the district temperature database file (DISTTEMP.DBF). If so, the program copies the temperature constant value to the model file.

Input File(s):

1. District temperature database file (DISTTEMP.DBF) with index file (DISTTEMP.NDX).

Output File(s):

1. Model database file (MODEL.DBF).

File Layout(s):

1. DISTTEMP.DBF (See Appendix A.)
2. MODEL.DBF (See Appendix A.)

Program Flow for Modl_Tmp.Prg.

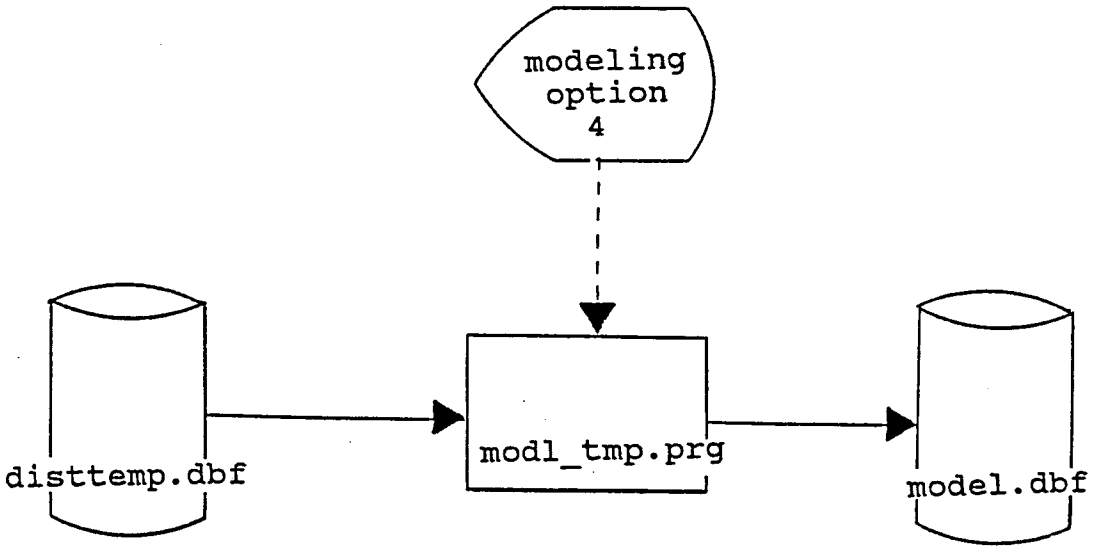


FIGURE 37

Building Model File

Program Specification 12
Program Listing

```
* File name:    MODL_TMP.PRG
* Program name: modl_tmp
* Project 2456: Texas Flexible Pavement Database
* TAMU/TTI
* Written by:   Victor Wong
* Created on:   June 29, 1988
* Last updated: July 26, 1988
* Purpose:      To find the district temperature constant for *
model file.

* open files
SELECT A
USE \pavedb\files\MODEL
SELECT B
USE \pavedb\files\DISTTEMP INDEX \pavedb\indexes\DISTTEMP

* for each sid number in the model file
SELECT A
DO WHILE .NOT. EOF()
   * find the data in the district temperature database file
   SELECT B
   SEEK A->HWYDIST

   * copy the data to the model database file
   IF FOUND()
      SELECT A
      REPLACE TEMPCONS WITH B->TEMPCONS
   ENDIF

   * skip to the next sid number in model file
   SELECT A
   SKIP
ENDDO

* close files
CLOSE ALL
```

Building Model File

Program Specification 13

Program Name: MODL_DYN.PRG

Purpose: To find the dynaflect data for the model file.

Edit/Procedure Information:

1. For each section identification number in the model file, the program determines if the section exists in the dynaflect database file (DYNAFLLD.DBF). If so, the program finds the average of each sensor reading and copies the results to the model file.

Input File(s):

1. Dynaflec database file (DYNAFLLD.DBF) with index file (DYNAFLLD.NDX).

Output File(s):

1. Model database file (MODEL.DBF).

File Layout(s):

1. DYNAFLLD.DBF (See Appendix A.)
2. MODEL.DBF (See Appendix A.)

Program Flow for Modl_Dyn.Prg.

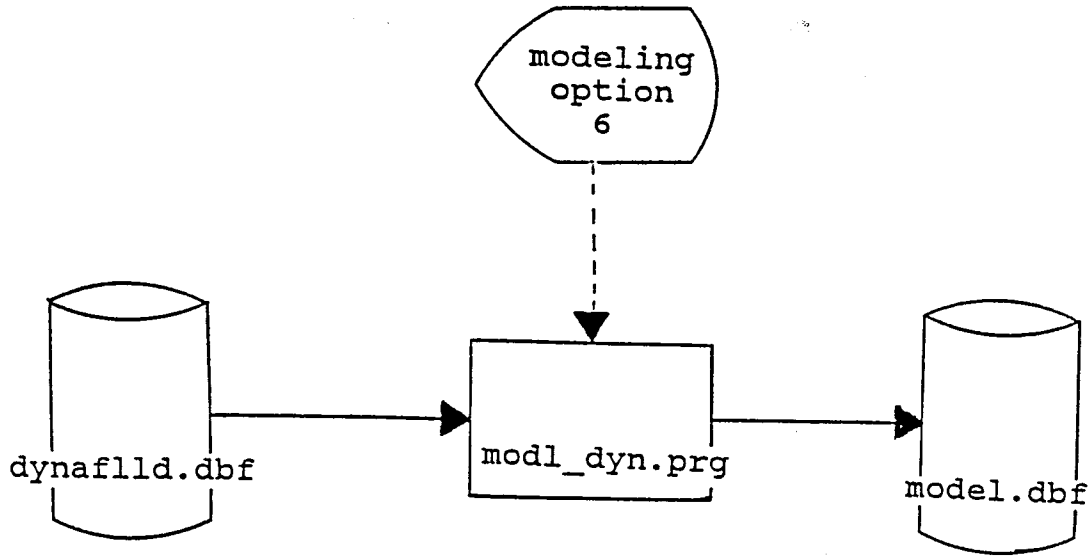


FIGURE 38

Building Model File

Program Specification 13
Program Listing

```
* File name:     MODL_DYN.PRG
* Program name: modl_dyn
* Project 2456: Texas Flexible Pavement Database
* TAMU/TTI
* Written by:   Victor Wong
* Created on:   June 29, 1988
* Last updated: July 26, 1988
* Purpose:      To find the dynaflec data for the model file.

* open files
SELECT A
USE \pavedb\files\MODEL
SELECT B
USE \pavedb\files\DYNAFLLD INDEX \pavedb\indexes\DYNAFLLD

* for each SID number in model file
SELECT A
DO WHILE .NOT. EOF()
   * find the data in the dynaflec database file
   SELECT B
   SEEK STR(A->SID_NO,4)

   * copy the data to the model database file
   IF FOUND()
      AV_SENS1RD = 0
      AV_SENS2RD = 0
      AV_SENS3RD = 0
      AV_SENS4RD = 0
      AV_SENS5RD = 0
      * for all the stations
      DO WHILE SID_NO = A->SID_NO
         * calculate the average sensor reading
         AV_SENS1RD = AV_SENS1RD + SENS1RD
         AV_SENS2RD = AV_SENS2RD + SENS2RD
         AV_SENS3RD = AV_SENS3RD + SENS3RD
         AV_SENS4RD = AV_SENS4RD + SENS4RD
         AV_SENS5RD = AV_SENS5RD + SENS5RD
         SKIP
      ENDDO

      SELECT A
      REPLACE DYNAFLC1 WITH AV_SENS1RD/14
      REPLACE DYNAFLC2 WITH AV_SENS2RD/14
      REPLACE DYNAFLC3 WITH AV_SENS3RD/14
      REPLACE DYNAFLC4 WITH AV_SENS4RD/14
      REPLACE DYNAFLC5 WITH AV_SENS5RD/14
   ENDIF
```

```
   * skip to the next sid number in model file
   SELECT A
   SKIP
ENDDO

* close files
CLOSE ALL
```

Building Model File

Program Specification 14

Program Name: MODL_FAL.PRG

Purpose: To find the falling weight data for the model file.

Edit/Procedure Information:

1. For each section identification number in the model file, the program determines if the section exists in the falling weight database file (FALLWGHT.DBF). If so, the program finds the average of each sensor reading and copies the results to the model file.

Input File(s):

1. Falling weight database file (FALLWGHT.DBF) with index file (FALLWGHT.NDX).

Output File(s):

1. Model database file (MODEL.DBF).

File Layout(s):

1. FALLWGHT.DBF (See Appendix A.)
2. MODEL.DBF (See Appendix A.)

Program Flow for Modl_Fal.Prg.

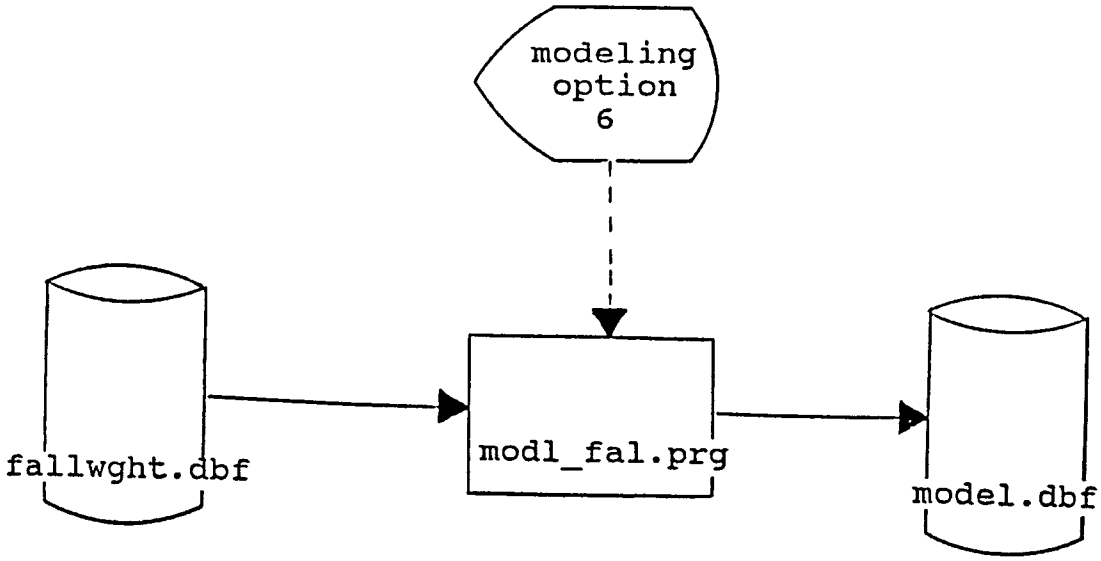


FIGURE 39

Building Model File

Program Specification 14
Program Listing

```
* File name:    MODL_FAL.PRG
* Program name: modl_fal
* Project 2456: Texas Flexible Pavement Database
* TAMU/TTI
* Written by:   Victor Wong
* Created on:   June 29, 1988
* Last updated: July 26, 1988
* Purpose:      To find falling weight data for the model file.

* open files
SELECT A
USE \pavedb\files\MODEL
SELECT B
USE \pavedb\files\FALLWGHT INDEX \pavedb\indexes\FALLWGHT
* for each sid number in the model file
SELECT A
DO WHILE .NOT. EOF()
   * find the data in the falling weight database file
   SELECT B
   SEEK STR(A->SID_NO,4)
   * copy the data to the model database file
   IF FOUND()
      * find the most recent layer
      DO WHILE SID_NO = A-> SID_NO
         SKIP
      ENDDO
      SKIP -1
      * calculate the average ssi reading
      AV_SSIGP1 = (SSIGP11+SSIGP21+SSIGP31+SSIGP41+SSIGP51) / 5
      AV_SSIGP2 = (SSIGP12+SSIGP22+SSIGP32+SSIGP42+SSIGP52) / 5
      AV_SSIGP3 = (SSIGP13+SSIGP23+SSIGP33+SSIGP43+SSIGP53) / 5
      AV_SSIGP4 = (SSIGP14+SSIGP24+SSIGP34+SSIGP44+SSIGP54) / 5
      AV_SSIGP5 = (SSIGP15+SSIGP25+SSIGP35+SSIGP45+SSIGP55) / 5
      SELECT A
      REPLACE SSIGP1 WITH AV_SSIGP1
      REPLACE SSIGP2 WITH AV_SSIGP2
      REPLACE SSIGP3 WITH AV_SSIGP3
      REPLACE SSIGP4 WITH AV_SSIGP4
      REPLACE SSIGP5 WITH AV_SSIGP5
   ENDIF

   * skip to the next sid number in model file
   SELECT A
   SKIP
ENDDO

* close files
CLOSE ALL
```

Section 4: Create the Distress File

Program Narrative for Distress

The Applications program (APPLICAT.PRG) calls the distress program (DISTRESS.PRG and DISTVISL.PRG) to create the Distress vs. 18 KEAL dBASE File. The file is created from the Master dBASE files - Location, Layer Identification, Traffic, Visual, Geometric and Shoulder Information and Serviceability Index. This file is used to create the 18 KIP graphs as well as to create the Model dBASE file. This program can be found in the subdirectory \PAVEDB\APPLICAT\DISTRESS. The files are stored in the subdirectory \PAVEDB\FILES. The indices are stored in the subdiectory \PAVEDB\INDEXES.

Program Flow for Distress.Prg

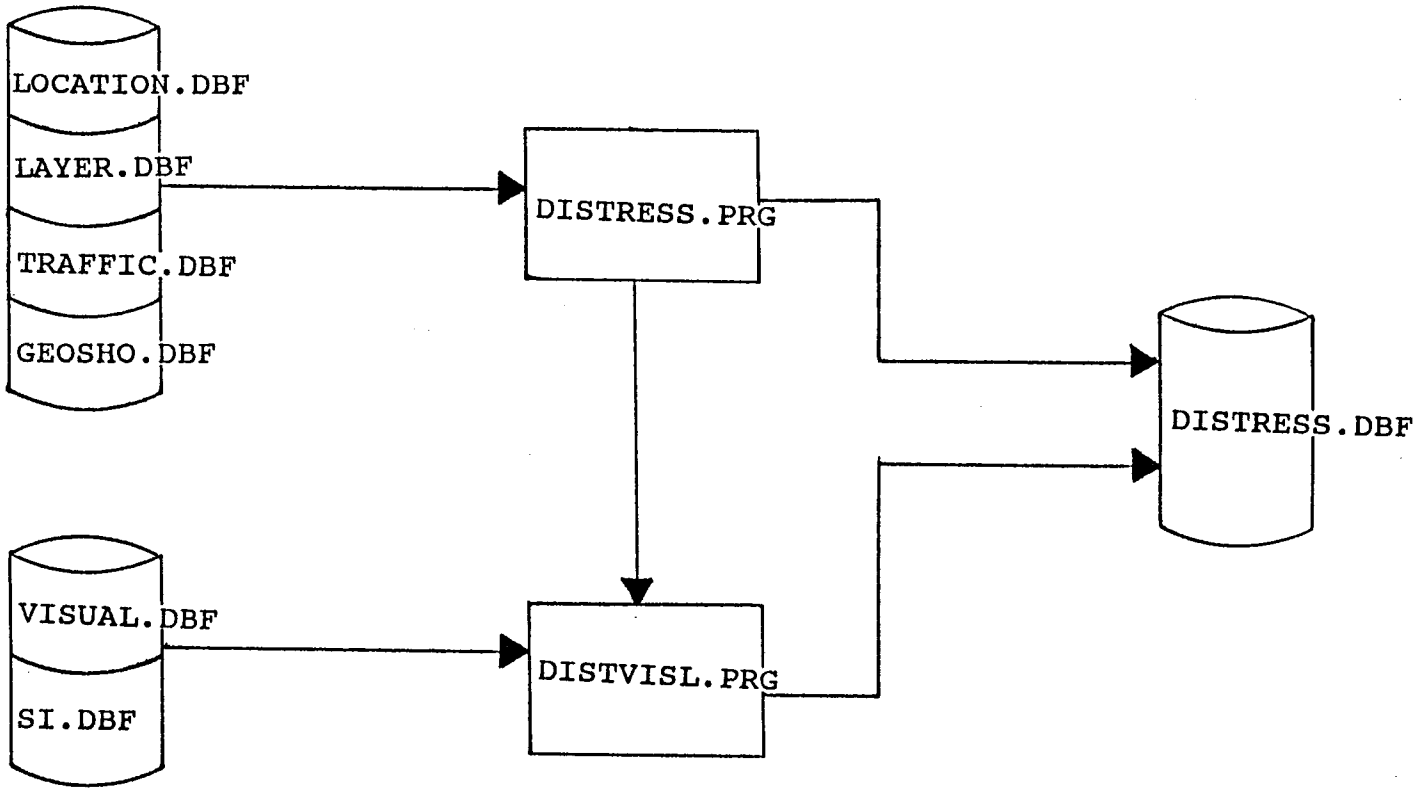


FIGURE 40

Distress vs. 18 Keal File

PROGRAM SPECIFICATION

Program Name: DISTRESS.PRG

Program Called: DISTVISL.PRG

Purpose: To create a dBASE III Distress vs 18 Keal File

Input Files:

	Files	Indices
Geometric & Shoulder	GEOSHO.DBF	GEONDX.NDX
Layer Identification File	LAYER.DBF	LAYNDX.NDX
Location File	LOCATION.DBF	LOCSID.NDX
Serviceability Index File	SI.DBF	SI.NDX
Traffic File	TRAFFIC.DBF	TRAFFIC.NDX
Visual File	VISUAL.DBF	VISUAL.NDX

Output Files:

Distress vs 18 Keal File (DISTRESS.DBF)

Distress vs. 18 KEAL File
File Layout

Field	Key	Size/Type	File * Source	Note**	Description
SID_NO	*	4N	L		Section Identification Number
YEAR	*	2N		A	Year
A18KEAL		11N	T	1	Annual Cumulative 18 Keal oneway
RUTT		1N	V	2	Rutting Area
ALLGCR		1N	V	2	Alligator Cracking Area
PATCH		1N	V	2	Patch
SIMEAN		7.5N	SI	6	Serviceability Index Mean
CONYEAR		2N	LI	3	Year of Last Major Construction
CONMON		2N	LI	3	Month of Last Major Construction
PAVETYP		2N	G	4	Pavement Type
OVRYEAR		2N	LI	5	Year of Last HMAC Overlay
OVRMON		2N	LI	5	Month of Last HMAC Overlay

*L - LOCATION
T - TRAFFIC FILE
V - VISUAL FILE
LI - LAYER IDENTIFICATION FILE (LAYER)
G - GEOMETRIC & SHOULDER INFORMATION FILE (GEOSHO)
SI - SERVICEABILITY INDEX FILE

**NOTE: The referenced note can be found on the following page.

NOTES:

A. A record is created for 1972 and for every year there after. (eg. If the year of the run is 1987, there is a record for 1972, 73, 74, 75 . . . . 87).

1. The 18 Keal is accumulated from the date of last major construction until and including 1972 for the first record. Every record there after only has the 18 Keal for that year.

2. This field is initialized to 9. The RUTTSL, RUTTMD, and RUTTSV fields are checked. If a 1, 2, or 3 exists in one of these fields, a 1, 2, or 3 is put in RUTT. If zeros exist in all three fields, a zero is put in RUTT. If a VISUAL record does not exist for the year, a 9 is put in RUTT (Note: The single digit fields RUTTSL, RUTTMD, and RUTTSV will have the values 000, 100, 200, 300, 010, 020, 030, 001, 002, or 003). The same process is followed for ALLGCRSL, ALLGCRMD, ALLGCRSV and PATCHPR, PATCHFR, PATCHGD.

3. If the last major construction was before January 1, 1973, the year and month of the most recent base layer is used. If the section (SID record) was constructed after December 31, 1972, the year and month of the subgrade is used.

4. The Pavement Type which corresponds to the year of the record is used.

5. The year and month of the most recent HMAC overlay for the year of the Distress vs 18 KIP record is used.

6. This field is initialized to 9.99999. So if a SI record does not exist, this field should has a value of 9.99999.

PROGRAM LISTING

```
*
* SUBSYSTEM:      APPLICATIONS
* PROGRAM NAME:   DISTRESS.PRG        5/23/88
* CALLED FROM:    APPLICAT.PRG
* REVISED ON:     06/02/88
* PROJECT 2456 - TEXAS FLEXIBLE PAVEMENT DATABASE CONVERSION
* TAMU/TTI
* AUTHOR:         TREVOR X. PEREIRA
* PURPOSE:        TO CREATE THE DISTRESS VS 18 KEAL FILE
*
* THE FOLLOWING FILES ARE NEEDED TO RUN THIS PROGRAM:
*          GEOSHO.DBF      GEONDX.NDX
*          LAYER.DBF       LAYNDX.NDX
*          TRAFFIC.DBF     TRAFFIC.NDX
*

SET TALK OFF
SET ECHO OFF
SET SAFETY OFF
CLEAR ALL

* assign databases to different work areas
SELECT 1
IF .NOT. FILE('\PAVEDB\FILES\DISTRESS.DBF')
     ? "DISTRESS FILE not found.  Please Check . . . "
     WAIT
     RETURN TO MASTER
ENDIF
IF .NOT. FILE('\PAVEDB\INDEXES\DISTRESS.NDX')
     ? "DISTRESS INDEX not found.  Please Check . . . "
     WAIT
     RETURN TO MASTER
ENDIF
USE \PAVEDB\FILES\DISTRESS INDEX \PAVEDB\INDEXES\DISTRESS
ZAP

SELECT 2
USE \PAVEDB\FILES\TRAFFIC INDEX \PAVEDB\INDEXES\TRAFFIC

SELECT 4
USE \PAVEDB\FILES\LAYER INDEX \PAVEDB\INDEXES\LAYNDX

SELECT 5
USE \PAVEDB\FILES\GEOSHO INDEX \PAVEDB\INDEXES\GEONDX

SELECT 7
USE \PAVEDB\FILES\LOCATION INDEX \PAVEDB\INDEXES\LOCSID

* INITIALIZE VARIABLES
CLEAR
MYEAR = 1972
```

```
MYEAR2 = 72
@ 10,10 say " "
mtemp = "  "
ACCEPT "Please enter the year of the run (19YY): " to MTEMP
MYEARRUN = VAL(MTEMP)
CLEAR
SET STAT ON
@ 10, 10 SAY "Processing.  Please wait . . . ."

SELECT 7
GOTO TOP
DO WHILE .NOT. EOF()
  DO WHILE MYEARRUN >= MYEAR
    * Initialize variables
    STORE 9 TO MRUTT, MALLGCR, MPATCH
    STORE 9.99999 TO MSIMEAN
    STORE 0 TO MA18KEAL
    STORE 9999 TO MSID_NO
    STORE 99 TO MCONYEAR, MCONMON, MPAVETYP, MOVRYEAR, MOVRMON

    * Getting Sid_no from Location file
    MSID_NO = G->SID_NO
    MSTRUC = 1
    MLAY = 1
    IF .NOT. G->ACTVFLAG .AND. G->INACTYR < MYEAR2
       MYEAR = MYEAR + 1
       LOOP
    ENDIF

    * GET CONSTRUCTION YEAR & MONTH FROM LAYER IDENTIFICATION FILE
    SELE 4
    SEEK STR(MSID_NO,4)+STR(MSTRUC,2)+STR(MLAY,2)
    IF FOUND()
      IF D->JOBCMPYR > 1972
        MCONYEAR = D->JOBCMPYR
        MCONMON  = D->JOBCMPMO
        MSTRUC = D->STRUCNUM
      ELSE
        DO WHILE D->SID_NO = MSID_NO
          SKIP
        ENDDO
        SKIP -1
        DO WHILE D->JOBCMPYR > MYEAR2
          SKIP -1
        ENDDO
        MSTRUC = D->STRUCNUM
        DO WHILE .NOT. BOF() .AND. D->LAYDESC <> 5
          SKIP -1
        ENDDO
        IF D->SID_NO <> MSID_NO
          MCONYEAR = 0
          MCONMON = 0
        ELSE
          MCONYEAR = D->JOBCMPYR
```

```
        MCONMON  = D->JOBCMPMO
      ENDIF
    ENDIF
  ELSE
    MCONYEAR = 0
    MCONMON = 0
    MSTRUC = 0
  ENDIF

  * GET PAVEMENT TYPE FROM GEOMETRIC AND SHOULDER INFORMATION FILE
  SELE 5
  SEEK STR(MSID_NO,4)+STR(MSTRUC,2)
  IF FOUND()
     MPAVETYP = E->PAVETYP
     IF WIDENFLG = 2
        MYEAR = MYEAR + 1
        LOOP
     ENDIF
  ELSE
     MPAVETYP = 0
  ENDIF

  * Getting 18 Kip from Traffic Data
  SELECT 2
  IF MYEAR = 1972
    SEEK STR(MSID_NO,4)
    DO WHILE B->SID_NO = MSID_NO .AND. B->YEAR < 1973
      MA18KEAL = MA18KEAL + B->A18KEAL
      SKIP
    ENDDO
  ELSE
    SEEK STR(MSID_NO,4)+STR(MYEAR,4)
    IF FOUND()
      MA18KEAL = B->A18KEAL
    ENDIF
  ENDIF

  * GET OVERLAY INFORMATION FROM LAYER IDENTIFICATION FILE
  SELE 4
  SEEK STR(MSID_NO,4)
  IF FOUND()
    DO WHILE D->SID_NO = MSID_NO
      SKIP
    ENDDO
    SKIP -1
    DO WHILE D->JOBCMPYR > MYEAR2
      SKIP -1
    ENDDO
    DO WHILE D->LAYMATCL <> 1 .AND. D->LAYMATCL <> 2 .AND. D->LAYMATCL <> 4
      SKIP -1
    ENDDO
    IF D->SID_NO = MSID_NO .AND. (D->LAYMATCL=1 .OR. D->LAYMATCL=2 .OR.
D->LAYMATCL=4)
      MOVRYEAR = D->JOBCMPYR
```

```
        MOVRMON  = D->JOBCMPMO
      ELSE
        MOVRYEAR = 0
        MOVRMON = 0
      ENDIF
    ELSE
      MOVRYEAR = 0
      MOVRMON = 0
    ENDIF

    SELE 1
    APPEND BLANK
    REPLACE SID_NO  WITH MSID_NO
    REPLACE YEAR    WITH MYEAR2
    REPLACE A18KEAL WITH MA18KEAL
    REPLACE RUTT    WITH MRUTT
    REPLACE ALLGCR  WITH MALLGCR
    REPLACE PATCH   WITH MPATCH
    REPLACE SIMEAN  WITH MSIMEAN
    REPLACE CONYEAR WITH MCONYEAR
    REPLACE CONMON  WITH MCONMON
    REPLACE PAVETYP WITH MPAVETYP
    REPLACE OVRYEAR WITH MOVRYEAR
    REPLACE OVRMON  WITH MOVRMON

    MYEAR = MYEAR + 1
    MYEAR2 = MYEAR2 + 1
  ENDDO
  MYEAR = 1972
  MYEAR2 = 72
  SELECT 7
  SKIP
ENDDO
CLOSE DATABASES
CLEAR
DO \PAVEDB\EDITUPDT\DISTVISL
CLEAR ALL
SET STAT OFF
RETURN
```

```
*
* SUBSYSTEM:      APPLICATIONS
* PROGRAM NAME:   DISTVISL.PRG        5/23/88
* REVISED ON:     07/01/88
* CALLED FROM:    DISTRESS.PRG
* PROJECT 2456 - TEXAS FLEXIBLE PAVEMENT DATABASE CONVERSION
* TAMU/TTI
* AUTHOR:         TREVOR X. PEREIRA
* PURPOSE:        TO ADD THE VISUAL & SI INFORMATION TO THE
*                 DISTRESS VS 18 KEAL FILE
*
* THE FOLLOWING FILES ARE NEEDED TO RUN THIS PROGRAM:
*         SI.DBF          SI.NDX
*                         SIDIST.NDX (This File is not mandatory)
*         VISUAL.DBF      VISUAL.NDX
*                         VISLDIST.NDX (This File is not mandatory)
*         DISTRESS.DBF    DISTRESS.DBF
*

* assign databases to different work areas
SET SAFETY OFF
SELECT 1
USE \PAVEDB\FILES\DISTRESS INDEX \PAVEDB\INDEXES\DISTRESS

SELECT 2
USE \PAVEDB\FILES\VISUAL
INDEX ON STR(SID_NO,4)+STR(YEAR,2) TO \PAVEDB\INDEXES\VISLDIST

SELECT 3
USE \PAVEDB\FILES\SI
INDEX ON STR(SID_NO,4)+STR(YEAR,2) TO \PAVEDB\INDEXES\SIDIST

* INITIALIZE VARIABLES
SELE 1
GOTO TOP
STORE 9 TO MRUTT, MALLGCR, MPATCH
STORE 9.99999 TO MSIMEAN
MSID_NO = A->SID_NO
MYEAR2  = A->YEAR
DO WHILE .NOT. EOF()
    STORE 9 TO MRUTT, MALLGCR, MPATCH
    STORE 9.99999 TO MSIMEAN

    * GET VISUAL INFORMATION
    SELE 2
    SEEK STR(MSID_NO,4)+STR(MYEAR2,2)
    IF FOUND()
      DO CASE
       CASE RUTTSL <> 0
         MRUTT = B->RUTTSL
       CASE RUTTMD <> 0
         MRUTT = B->RUTTMD
```

```
        CASE RUTISV <> 0
          MRUTT = B->RUTISV
        OTHERWISE
          MRUTT = 0
       ENDCASE
       DO CASE
        CASE ALLGCRSL <> 0
          MALLGCR = B->ALLGCRSL
        CASE ALLGCRMD <> 0
          MALLGCR = B->ALLGCRMD
        CASE ALLGCRSV <> 0
          MALLGCR = B->ALLGCRSV
        OTHERWISE
          MALLGCR = 0
       ENDCASE
       DO CASE
        CASE PATCHGD <> 0
          MPATCH = B->PATCHGD
        CASE PATCHFR <> 0
          MPATCH = B->PATCHFR
        CASE PATCHPR <> 0
          MPATCH = B->PATCHPR
        OTHERWISE
          MPATCH = 0
       ENDCASE
     ELSE
        MRUTT = 9
        MALLGCR = 9
        MPATCH = 9
     ENDIF

     * GET SI INFORMATION
     SELE 3
     SEEK STR(MSID_NO,4)+STR(MYEAR2,2)
     IF FOUND()
       MSIMEAN = C->SIMEAN
     ELSE
       MSIMEAN = 9.99999
     ENDIF
     SELE 1
     REPLACE RUTT    WITH MRUTT
     REPLACE ALLGCR  WITH MALLGCR
     REPLACE PATCH   WITH MPATCH
     REPLACE SIMEAN  WITH MSIMEAN
     SKIP
     MSID_NO = A->SID_NO
     MYEAR2  = A->YEAR
ENDDO
CLOSE DATABASES
CLEAR
SET SAFETY ON
RETURN
```

CHAPTER 6

DATABASE MAINTENANCE SUBSYSTEM

GENERAL NARRATIVE

The Database Maintenance programs permit the user to reindex files, change the installation defaults or backup all the files. These programs are selected from the Main Menu of the Flexible Pavement System and are called by the main program DBMAIN.PRG. See the Main Menu Program Flow Diagram (Figure 5) in Chapter 1.

The Reindex program (REINDEX.PRG) reindexes all the master files and some of the temporary files. All the files used are in the subdirectory \PAVEDB\FILES and the respective indices are in \PAVEDB\INDEXES. REINDEX.PRG is stored in \PAVEDB.

The Installation program (INSTDEFL.PRG) sets up the drive that is to be used, the floppy drive to backup up files to and the defaults for the printers. The defaults are stored in a memory variable file in INSTALL.MEM. At the start of the program, the file is loaded into memory. The defaults can be changed at any time by going through the Installation Menu (choice 6 on the Main Menu). INSTDEFL.PRG and INSTALL.MEM are stored in \PAVEDB.

The Backup program (BACKUP.PRG) backs up all the master files to the floppy disk drive designated in the installation menu. This program is in \PAVEDB\BACKUP.

PROGRAM SPECIFICATION

Program Name: REINDEX.PRG

Purpose: To reindex all the master files.

Input\Output Files:

The following files are used along with their indices:

Master Files/Table	Data File	Index File
Location	LOCATION.DBF	LOCSID.NDX
Layer Identification	LAYER.DBF	LAYNDX.NDX
Geometric & Shoulder	GEOSHO.DBF	GEONDX.NDX
Surface	SURFACE.DBF	SURFNDX.NDX
Subgrade	SUBGRADE.DBF	SUBGNDX.NDX
Layer Thickness	LAYTHICK.DBF	LAYTNDX.NDX
Visual Rating	VISUAL.DBF	VISUAL.NDX
Serviceability Index	SI.DBF	SI.NDX
Falling Weight SSI	FALLWGHT.DBF	FALLWGHT.NDX
Dynaflect Measure	DYNAFLLD.DBF	DYNAFLLD.NDX
Skid Measurement	SKID.DBF	SKID.NDX
Environment	ENV.DBF	ENV.NDX
Weather	WEATHER.DBF	WEATHER.NDX
Traffic	TRAFFIC.DBF	TRAFFIC.NDX
County Name Table	CNTYTBL.DBF	CTYTBLNO.NDX
Temporary Files	**Data File**	**Index File**
PES SSI	PESSSI.DBF	PESSSI.NDX
PES Skid	PESSKD.DBF	PESSKD.NDX
PES Visual	PESVISL.DBF	PESVISL.NDX
PES MRM	PESMRM.DBF	PESMRM.NDX
PES General	PESGEN.DBF	PESGEN.NDX
PES Scores	PESSCR.DBF	PESSCR.NDX
Location	LOCNCHNG.DBF	LOCNCHNG.NDX
	LOCN_NEW.DBF	LOCN_NEW.NDX
Layer	LAYRCHNG.DBF	LAYRCHNG.NDX
	LAYR_NEW.DBF	LAYR_NEW.NDX
Geometric and Shoulder	GEOSCHNG.DBF	GEOSCHNG.NDX
	GEOS_NEW.DBF	GEOS_NEW.NDX
Layer Thickness	LAYTCHNG.DBF	LAYTCHNG.NDX
	LAYT_NEW.DBF	LAYT_NEW.NDX
Surface	SURFCHNG.DBF	SURFCHNG.NDX
	SURF_NEW.DBF	SURF_NEW.NDX
Subgrade	SUBGCHNG.DBF	SUBGCHNG.NDX
	SUBG_NEW.DBF	SUBG_NEW.NDX

Program REINDEX

```
*
* SUBSYSTEM:            DATABASE MAINTENANCE
* PROGRAM NAME:         REINDEX.PRG     010/04/88
* MODIFIED ON:          10/21/88
* PROJECT 2456 - TEXAS FLEXIBLE PAVEMENT DATABASE CONVERSION
* TAMU/TTI
* AUTHOR:               TREVOR X. PEREIRA
* PURPOSE:              TO REINDEX ALL THE MASTER AND TEMPORARY
*                      FILES USED IN THE FLEXIBLE PAVEMENT SYSTEM
*

CLEAR
SET PATH TO \PAVEDB\FILES;\PAVEDB\INDEXES;\PAVEDB\EDITUPDT
SET STAT ON
SET ECHO Off
SET TALK Off
@ 3, 10 SAY "REINDEXING Files.  Please Wait . . ."
? " "
? "Reindexing Layer File . . ."
USE LAYER INDE LAYNDX
REIN
? "Reindexing Layer Thickness File . . ."
USE LAYTHICK INDE LAYTNDX
REIN
? "Reindexing Geometric & Shoulder File . . ."
USE GEOSHO INDE GEONDX
REIN
? "Reindexing Surface File . . ."
USE SURFACE INDE SURFNDX
REIN
? "Reindexing Subgrade File . . ."
USE SUBGRADE INDE SUBGNDX
REIN
? "Reindexing Serviceability Index File . . ."
USE SI INDE SI
REIN
? "Reindexing Visual Rating File . . ."
USE VISUAL INDE VISUAL
REIN
? "Reindexing Skid File . . ."
USE SKID INDE SKID
REIN
? "Reindexing Dynaflect File . . ."
USE DYNAFLLD INDE DYNAFLLD
REIN
? "Reindexing Falling Weight File . . ."
USE FALLWGHT INDE FALLWGHT
REIN
? "Reindexing Environment File . . ."
USE ENV INDE ENV
```

```
REIN
? "Reindexing Weather File . . ."
USE WEATHER INDE WEATHER
REIN
? "Reindexing Location File . . ."
USE LOCATION INDE LOCSID
REIN
? "Reindexing Traffic File . . ."
USE TRAFFIC INDE TRAFFIC
REIN
? "Reindexing all PES Files . . ."
USE PESSSI INDE PESSSI
REIN
USE PESSKD INDE PESSKD
REIN
USE PESVISL INDE PESVISL
REIN
USE PESMRM INDE PESMRM
REIN
USE PESGEN INDE PESGEN
REIN
USE PESSCR INDE PESSCR
REIN

? "Reindexing all temporary Inventory Files . . ."
IF FILE('\PAVEDB\EDITUPDT\LAYRCHNG.DBF')
     USE LAYRCHNG INDE LAYRCHNG
     REIN
ENDIF
IF FILE('\PAVEDB\EDITUPDT\LOCNCHNG.DBF')
     USE LOCNCHNG INDE LOCNCHNG
     REIN
ENDIF
IF FILE('\PAVEDB\EDITUPDT\LAYTCHNG.DBF')
     USE LAYTCHNG INDE LAYTCHNG
     REIN
ENDIF
IF FILE('\PAVEDB\EDITUPDT\GEOSCHNG.DBF')
     USE GEOSCHNG INDE GEOSCHNG
     REIN
ENDIF
IF FILE('\PAVEDB\EDITUPDT\SURFCHNG.DBF')
     USE SURFCHNG INDE SURFCHNG
     REIN
ENDIF
IF FILE('\PAVEDB\EDITUPDT\SUBGCHNG.DBF')
     USE SUBGCHNG INDE SUBGCHNG
     REIN
ENDIF
USE LAYR_NEW INDE LAYR_NEW
REIN
USE LOCN_NEW INDE LOCN_NEW
REIN
USE LAYT_NEW INDE LAYT_NEW
```

```
REIN
USE GEOS_NEW INDE GEOS_NEW
REIN
USE SURF_NEW INDE SURF_NEW
REIN
USE SUBG_NEW INDE SUBG_NEW
REIN
IF FILE('\PAVEDB\FILES\DISTRESS')
     USE \PAVEDB\FILES\DISTRESS INDE \PAVEDB\INDEXES\DISTRESS
     REIN
ENDIF

CLOSE ALL
SET PATH TO
? "Reindexing done ! ! !"
WAIT
SET STAT OFF
```

PROGRAM SPECIFICATION

Program Name: INSTDEFL.PRG

Purpose: To set up the printer and drive defaults for the system.

Input/Output Files:

Memory Variable File - INSTALL.MEM

Program INSTDEFL

```
*
* SUBSYSTEM:            DATABASE MAINTAINENCE
* PROGRAM NAME:         INSTDEFL.PRG     09/19/88
* PROJECT 2456 - TEXAS FLEXIBLE PAVEMENT DATABASE CONVERSION
* TAMU/TTI
* AUTHOR:               TREVOR X. PEREIRA
* PURPOSE:              TO INSTALL THE MAIN MENU SYSTEM DEFAULTS
*                       AND STORE THEM TO A MEMORY VARIABLE FILE.
*

SET SAFETY OFF
CLEA
@ 3, 25 SAY "INSTALLATION MENU"
@  6, 10 SAY "Please enter the drive being used (NORMALLY C )" GET IDRIVE
@  7, 15 SAY "DO NOT specify drive 'A' or drive 'B'"
@  9, 10 SAY "Please enter the drive you want to BACK UP files:" GET IDRIVE2
@ 10, 15 SAY "MUST Specify 'A' or 'B' only"
@ 12, 10 SAY "Which port do you want to send the SUMMARY REPORT: " GET IPORT1
@ 13, 10 SAY "  and other Landscape reports"
@ 14, 15 SAY "Specify LPT1, LPT2 or LPT3"
@ 16, 10 SAY "Which port do you want to send the INVENTORY UPDATE REPORT: "
GET IPORT2
@ 17, 15 SAY "Specify LPT1, LPT2 or LPT3"
READ
SET DEFAULT TO &IDRIVE
MDRIVE = IDRIVE
MDRIVE2 = IDRIVE2
MPORT1  = IPORT1
MPORT2  = IPORT2
SAVE TO \PAVEDB\INSTALL ALL LIKE I*
RELE ALL LIKE I*
SET SAFETY ON
RETURN
```

PROGRAM SPECIFICATION

Program Name: BACKUP.PRG

Purpose: To backup the master files to floppy diskettes.

Input/Output Files:

The following files are used:

Location	LOCATION.DBF
Layer Identification	LAYER.DBF
Geometric & Shoulder	GEOSHO.DBF
Surface	SURFACE.DBF
Subgrade	SUBGRADE.DBF
Layer Thickness	LAYTHICK.DBF
Visual Rating	VISUAL.DBF
Serviceability Index	SI.DBF
Falling Weight SSI	FALLWGHT.DBF
Dynaflect Measure	DYNAFLLD.DBF
Skid Measurement	SKID.DBF
Environment	ENV.DBF
Weather	WEATHER.DBF
Traffic	TRAFFIC.DBF

Program BACKUP

```
*
* SUBSYSTEM:        BACKUP.PRG
* PROGRAM NAME:     BACKUP.PRG              07/03/88
* MODIFIED ON:      09/20/88
* CALLED FROM:      DBMAIN.PRG
* PROJECT 2456 -    TEXAS FLEXIBLE PAVEMENT DATABASE CONVERSION
* TAMU/TTI
* AUTHOR:           TREVOR X. PEREIRA
* PURPOSE:          TO BACKUP DATA TO FLOPPY DISKETTES
*

CLEAR
SET TYPE TO 0
STORE 0 TO MSIZE, MNUMFIELDS, MHEADER, MCOUNT, MDISKS, MBYTES
STORE 0 TO MLOCATION, MLAYER, MLAYTHICK, MGEOSHO, MSURFACE, MSUBGRADE,
MVISUAL
STORE 0 TO MDYNAFLLD, MSI, MTRAFFIC, MWEATHER, MFALLWGHT, MENV, MSKID
STORE "" TO MNULL
SET ESCAPE OFF
@ 10, 10 SAY "Calculating file sizes.  Please Wait . . . ."

SET DEFAU TO &MDRIVE
MBACKDRV = MDRIVE2 + ":"
* Making sure the Master Files are present
IF .NOT. FILE('\PAVEDB\FILES\LOCATION.DBF')
   ? "LOCATION file is not found.  Please Check . . ."
   WAIT
   RETURN TO MASTER
ENDIF
IF .NOT. FILE('\PAVEDB\FILES\LAYER.DBF')
   ? "LAYER file is not found.  Please Check . . ."
   WAIT
   RETURN TO MASTER
ENDIF
IF .NOT. FILE('\PAVEDB\FILES\LAYTHICK.DBF')
   ? "LAYTHICK file is not found.  Please Check . . ."
   WAIT
   RETURN TO MASTER
ENDIF
IF .NOT. FILE('\PAVEDB\FILES\GEOSHO.DBF')
   ? "GEOSHO file is not found.  Please Check . . ."
   WAIT
   RETURN TO MASTER
ENDIF
IF .NOT. FILE('\PAVEDB\FILES\SURFACE.DBF')
   ? "SURFACE file is not found.  Please Check . . ."
   WAIT
   RETURN TO MASTER
ENDIF
IF .NOT. FILE('\PAVEDB\FILES\SUBGRADE.DBF')
```

```
   ? "SUBGRADE file is not found.  Please Check . . ."
   WAIT
   RETURN TO MASTER
ENDIF
IF .NOT. FILE('\PAVEDB\FILES\VISUAL.DBF')
   ? "VISUAL file is not found.  Please Check . . ."
   WAIT
   RETURN TO MASTER
ENDIF
IF .NOT. FILE('\PAVEDB\FILES\DYNAFLLD.DBF')
   ? "DYNAFLLD file is not found.  Please Check . . ."
   WAIT
   RETURN TO MASTER
ENDIF
IF .NOT. FILE('\PAVEDB\FILES\SI.DBF')
   ? "SI file is not found.  Please Check . . ."
   WAIT
   RETURN TO MASTER
ENDIF
IF .NOT. FILE('\PAVEDB\FILES\TRAFFIC.DBF')
   ? "TRAFFIC file is not found.  Please Check . . ."
   WAIT
   RETURN TO MASTER
ENDIF
IF .NOT. FILE('\PAVEDB\FILES\WEATHER.DBF')
   ? "WEATHER file is not found.  Please Check . . ."
   WAIT
   RETURN TO MASTER
ENDIF
IF .NOT. FILE('\PAVEDB\FILES\FALLWGHT.DBF')
   ? "FALLWGHT file is not found.  Please Check . . ."
   WAIT
   RETURN TO MASTER
ENDIF
IF .NOT. FILE('\PAVEDB\FILES\ENV.DBF')
   ? "ENV file is not found.  Please Check . . ."
   WAIT
   RETURN TO MASTER
ENDIF
IF .NOT. FILE('\PAVEDB\FILES\SKID.DBF')
   ? "SKID file is not found.  Please Check . . ."
   WAIT
   RETURN TO MASTER
ENDIF

* Calculation file sizes of all files
USE \PAVEDB\FILES\LOCATION
STORE RECCOUNT() * RECSIZE() TO MSIZE
MNULL = ""
DO WHILE MNULL < FIELD(MNUMFIELDS + 1)
   MNUMFIELDS = MNUMFIELDS + 1
ENDDO
MHEADER = (32 * MNUMFIELDS) + 34
MLOCATION = MSIZE + MHEADER + 20
```

```
MBYTES = MBYTES + MLOCATION
MDISKS = 1

USE \PAVEDB\FILES\LAYER
MNULL = ""
STORE 0 TO MNUMFIELDS, MHEADER, MSIZE
STORE RECCOUNT() * RECSIZE() TO MSIZE
DO WHILE MNULL < FIELD(MNUMFIELDS + 1)
   MNUMFIELDS = MNUMFIELDS + 1
ENDDO
MHEADER = (32 * MNUMFIELDS) + 34
MLAYER = MSIZE + MHEADER + 20
MBYTES = MBYTES + MLAYER
IF MBYTES > 360000
     MBYTES = MLAYER
     MDISKS = MDISKS + 1
ENDIF

USE \PAVEDB\FILES\LAYTHICK
MNULL = ""
STORE 0 TO MNUMFIELDS, MHEADER, MSIZE
STORE RECCOUNT() * RECSIZE() TO MSIZE
DO WHILE MNULL < FIELD(MNUMFIELDS + 1)
   MNUMFIELDS = MNUMFIELDS + 1
ENDDO
MHEADER = (32 * MNUMFIELDS) + 34
MLAYTHICK = MSIZE + MHEADER + 20
MBYTES = MBYTES + MLAYTHICK
IF MBYTES > 360000
     MBYTES = MLAYTHICK
     MDISKS = MDISKS + 1
ENDIF

USE \PAVEDB\FILES\GEOSHO
MNULL = ""
STORE 0 TO MNUMFIELDS, MHEADER, MSIZE
STORE RECCOUNT() * RECSIZE() TO MSIZE
DO WHILE MNULL < FIELD(MNUMFIELDS + 1)
   MNUMFIELDS = MNUMFIELDS + 1
ENDDO
MHEADER = (32 * MNUMFIELDS) + 34
MGEOSHO = MSIZE + MHEADER + 20
MBYTES = MBYTES + MGEOSHO
IF MBYTES > 360000
     MBYTES = MGEOSHO
     MDISKS = MDISKS + 1
ENDIF

USE \PAVEDB\FILES\SURFACE
MNULL = ""
STORE 0 TO MNUMFIELDS, MHEADER, MSIZE
STORE RECCOUNT() * RECSIZE() TO MSIZE
DO WHILE MNULL < FIELD(MNUMFIELDS + 1)
   MNUMFIELDS = MNUMFIELDS + 1
```

```
ENDDO
MHEADER = (32 * MNUMFIELDS) + 34
MSURFACE = MSIZE + MHEADER + 20
MBYTES = MBYTES + MSURFACE
IF MBYTES > 360000
     MBYTES = MSURFACE
     MDISKS = MDISKS + 1
ENDIF

USE \PAVEDB\FILES\SUBGRADE
MNULL = ""
STORE 0 TO MNUMFIELDS, MHEADER, MSIZE
STORE RECCOUNT() * RECSIZE() TO MSIZE
DO WHILE MNULL < FIELD(MNUMFIELDS + 1)
   MNUMFIELDS = MNUMFIELDS + 1
ENDDO
MHEADER = (32 * MNUMFIELDS) + 34
MSUBGRADE = MSIZE + MHEADER + 20
MBYTES = MBYTES + MSUBGRADE
IF MBYTES > 360000
     MBYTES = MSUBGRADE
     MDISKS = MDISKS + 1
ENDIF

USE \PAVEDB\FILES\VISUAL
MNULL = ""
STORE 0 TO MNUMFIELDS, MHEADER, MSIZE
STORE RECCOUNT() * RECSIZE() TO MSIZE
DO WHILE MNULL < FIELD(MNUMFIELDS + 1)
   MNUMFIELDS = MNUMFIELDS + 1
ENDDO
MHEADER = (32 * MNUMFIELDS) + 34
MVISUAL = MSIZE + MHEADER + 20
MBYTES = MBYTES + MVISUAL
IF MBYTES > 360000
     MBYTES = MVISUAL
     MDISKS = MDISKS + 1
ENDIF

USE \PAVEDB\FILES\DYNAFLLD
MNULL = ""
STORE 0 TO MNUMFIELDS, MHEADER, MSIZE
STORE RECCOUNT() * RECSIZE() TO MSIZE
DO WHILE MNULL < FIELD(MNUMFIELDS + 1)
   MNUMFIELDS = MNUMFIELDS + 1
ENDDO
MHEADER = (32 * MNUMFIELDS) + 34
MDYNAFLLD = MSIZE + MHEADER + 20
MBYTES = MBYTES + MDYNAFLLD
IF MBYTES > 360000
     MBYTES = MDYNAFLLD
     MDISKS = MDISKS + 1
ENDIF
```

```
USE \PAVEDB\FILES\FALLWGHT
MNULL = ""
STORE 0 TO MNUMFIELDS, MHEADER, MSIZE
STORE RECCOUNT() * RECSIZE() TO MSIZE
DO WHILE MNULL < FIELD(MNUMFIELDS + 1)
   MNUMFIELDS = MNUMFIELDS + 1
ENDDO
MHEADER = (32 * MNUMFIELDS) + 34
MFALLWGHT = MSIZE + MHEADER + 20
MBYTES = MBYTES + MFALLWGHT
IF MBYTES > 360000
     MBYTES = MFALLWGHT
     MDISKS = MDISKS + 1
ENDIF

USE \PAVEDB\FILES\SI
MNULL = ""
STORE 0 TO MNUMFIELDS, MHEADER, MSIZE
STORE RECCOUNT() * RECSIZE() TO MSIZE
DO WHILE MNULL < FIELD(MNUMFIELDS + 1)
   MNUMFIELDS = MNUMFIELDS + 1
ENDDO
MHEADER = (32 * MNUMFIELDS) + 34
MSI =  MSIZE + MHEADER + 20
MBYTES = MBYTES + MSI
IF MBYTES > 360000
     MBYTES = MSI
     MDISKS = MDISKS + 1
ENDIF

USE \PAVEDB\FILES\TRAFFIC
MNULL = ""
STORE 0 TO MNUMFIELDS, MHEADER, MSIZE
STORE RECCOUNT() * RECSIZE() TO MSIZE
DO WHILE MNULL < FIELD(MNUMFIELDS + 1)
   MNUMFIELDS = MNUMFIELDS + 1
ENDDO
MHEADER = (32 * MNUMFIELDS) + 34
MTRAFFIC = MSIZE + MHEADER + 20
MBYTES = MBYTES + MTRAFFIC
IF MBYTES > 360000
     MBYTES = MTRAFFIC
     MDISKS = MDISKS + 1
ENDIF

USE \PAVEDB\FILES\WEATHER
MNULL = ""
STORE 0 TO MNUMFIELDS, MHEADER, MSIZE
STORE RECCOUNT() * RECSIZE() TO MSIZE
DO WHILE MNULL < FIELD(MNUMFIELDS + 1)
   MNUMFIELDS = MNUMFIELDS + 1
ENDDO
MHEADER = (32 * MNUMFIELDS) + 34
MWEATHER = MSIZE + MHEADER + 20
```

```
MBYTES = MBYTES + MWEATHER
IF MBYTES > 360000
     MBYTES = MWEATHER
     MDISKS = MDISKS + 1
ENDIF

USE \PAVEDB\FILES\ENV
MNULL = ""
STORE 0 TO MNUMFIELDS, MHEADER, MSIZE
STORE RECCOUNT() * RECSIZE() TO MSIZE
DO WHILE MNULL < FIELD(MNUMFIELDS + 1)
   MNUMFIELDS = MNUMFIELDS + 1
ENDDO
MHEADER = (32 * MNUMFIELDS) + 34
MENV = MSIZE + MHEADER + 20
MBYTES = MBYTES + MENV
IF MBYTES > 360000
     MBYTES = MENV
     MDISKS = MDISKS + 1
ENDIF

USE \PAVEDB\FILES\SKID
MNULL = ""
STORE 0 TO MNUMFIELDS, MHEADER, MSIZE
STORE RECCOUNT() * RECSIZE() TO MSIZE
DO WHILE MNULL < FIELD(MNUMFIELDS + 1)
   MNUMFIELDS = MNUMFIELDS + 1
ENDDO
MHEADER = (32 * MNUMFIELDS) + 34
MSKID = MSIZE + MHEADER + 20
MBYTES = MBYTES + MSKID
IF MBYTES > 360000
     MDISKS = MDISKS + 1
ENDIF
USE
MGET = ""
@ 10, 05 SAY "You will need " + STR(MDISKS,3) + "  blank formatted disk(s) of
360 K"
@ 12, 05 SAY "If you do not have the diskettes ready, press the Esc key."
@ 13, 05 SAY "Otherwise press any key to continue."
CLEA TYPE
READ
IF READKEY() = 12
   CLEAR
   RETURN TO MASTER
ELSE
   SET SAFETY ON
   CLEAR
   @ 5,0 SAY "Please wait.  Backing up LOCATION files . . ."
   IF FILE('\PAVEDB\FILES\BACKUP\LOCATION.DBF')
     DELE FILE \PAVEDB\FILES\BACKUP\LOCATION.DBF
   ENDIF
   COPY FILE \PAVEDB\FILES\LOCATION.DBF TO \PAVEDB\FILES\BACKUP\LOCATION.DBF
   SET DEFAULT TO &MBACKDRV
```

```
MDISKSPACE = DISKSPACE()
DO WHILE MDISKSPACE < MLOCATION
  ? "Not enough space on diskette in drive " + MBACKDRV
  ? "Please replace with another diskette and press any key to continue"
  WAIT " "
  MDISKSPACE = DISKSPACE()
ENDDO
SET DEFAULT TO &MDRIVE
COPY FILE \PAVEDB\FILES\LOCATION.DBF TO &MBACKDRV\LOCATION.DBF

? "Please wait.  Backing up LAYER files . . ."
IF FILE('\PAVEDB\FILES\BACKUP\LAYER.DBF')
  DELE FILE \PAVEDB\FILES\BACKUP\LAYER.DBF
ENDIF
COPY FILE \PAVEDB\FILES\LAYER.DBF TO \PAVEDB\FILES\BACKUP\LAYER.DBF
SET DEFAULT TO &MBACKDRV
MDISKSPACE = DISKSPACE()
DO WHILE MDISKSPACE < MLAYER
  ? "Not enough space on diskette in drive " + MBACKDRV
  ? "Please replace with another diskette and press any key to continue"
  WAIT " "
  MDISKSPACE = DISKSPACE()
ENDDO
SET DEFAULT TO &MDRIVE
COPY FILE \PAVEDB\FILES\LAYER.DBF TO &MBACKDRV\LAYER.DBF

? "Please wait.  Backing up LAYTHICK files . . ."
IF FILE('\PAVEDB\FILES\BACKUP\LAYTHICK.DBF')
  DELE FILE \PAVEDB\FILES\BACKUP\LAYTHICK.DBF
ENDIF
COPY FILE \PAVEDB\FILES\LAYTHICK.DBF TO \PAVEDB\FILES\BACKUP\LAYTHICK.DBF
SET DEFAULT TO  &MBACKDRV
MDISKSPACE = DISKSPACE()
DO WHILE MDISKSPACE < MLAYTHICK
  ? "Not enough space on diskette in drive " + MBACKDRV
  ? "Please replace with another diskette and press any key to continue"
  WAIT " "
  MDISKSPACE = DISKSPACE()
ENDDO
SET DEFAULT TO  &MDRIVE
COPY FILE \PAVEDB\FILES\LAYTHICK.DBF TO &MBACKDRV\LAYTHICK.DBF

? "Please wait.  Backing up GEOSHO files . . ."
IF FILE('\PAVEDB\FILES\BACKUP\GEOSHO.DBF')
  DELE FILE \PAVEDB\FILES\BACKUP\GEOSHO.DBF
ENDIF
COPY FILE \PAVEDB\FILES\GEOSHO.DBF TO \PAVEDB\FILES\BACKUP\GEOSHO.DBF
SET DEFAULT TO  &MBACKDRV
MDISKSPACE = DISKSPACE()
DO WHILE MDISKSPACE < MGEOSHO
  ? "Not enough space on diskette in drive " + MBACKDRV
  ? "Please replace with another diskette and press any key to continue"
  WAIT " "
  MDISKSPACE = DISKSPACE()
```

```
ENDDO
SET DEFAULT TO  &MDRIVE
COPY FILE \PAVEDB\FILES\GEOSHO.DBF TO &MBACKDRV\GEOSHO.DBF

? "Please wait.  Backing up SURFACE files . . ."
IF FILE('\PAVEDB\FILES\BACKUP\SURFACE.DBF')
  DELE FILE \PAVEDB\FILES\BACKUP\SURFACE.DBF
ENDIF
COPY FILE \PAVEDB\FILES\SURFACE.DBF TO \PAVEDB\FILES\BACKUP\SURFACE.DBF
SET DEFAULT TO &MBACKDRV
MDISKSPACE = DISKSPACE()
DO WHILE MDISKSPACE < MSURFACE
  ? "Not enough space on diskette in drive " + MBACKDRV
  ? "Please replace with another diskette and press any key to continue"
  WAIT " "
  MDISKSPACE = DISKSPACE()
ENDDO
SET DEFAULT TO &MDRIVE
COPY FILE \PAVEDB\FILES\SURFACE.DBF TO &MBACKDRV\SURFACE.DBF

? "Please wait.  Backing up SUBGRADE files . . ."
IF FILE('\PAVEDB\FILES\BACKUP\SUBGRADE.DBF')
  DELE FILE \PAVEDB\FILES\BACKUP\SUBGRADE.DBF
ENDIF
COPY FILE \PAVEDB\FILES\SUBGRADE.DBF TO \PAVEDB\FILES\BACKUP\SUBGRADE.DBF
SET DEFAULT TO &MBACKDRV
MDISKSPACE = DISKSPACE()
DO WHILE MDISKSPACE < MSUBGRADE
  ? "Not enough space on diskette in drive " + MBACKDRV
  ? "Please replace with another diskette and press any key to continue"
  WAIT " "
  MDISKSPACE = DISKSPACE()
ENDDO
SET DEFAULT TO &MDRIVE
COPY FILE \PAVEDB\FILES\SUBGRADE.DBF TO &MBACKDRV\SUBGRADE.DBF

? "Please wait.  Backing up VISUAL files . . ."
IF FILE('\PAVEDB\FILES\BACKUP\VISUAL.DBF')
  DELE FILE \PAVEDB\FILES\BACKUP\VISUAL.DBF
ENDIF
COPY FILE \PAVEDB\FILES\VISUAL.DBF TO \PAVEDB\FILES\BACKUP\VISUAL.DBF
SET DEFAULT TO &MBACKDRV
MDISKSPACE = DISKSPACE()
DO WHILE MDISKSPACE < MVISUAL
  ? "Not enough space on diskette in drive " + MBACKDRV
  ? "Please replace with another diskette and press any key to continue"
  WAIT " "
  MDISKSPACE = DISKSPACE()
ENDDO
SET DEFAULT TO &MDRIVE
COPY FILE \PAVEDB\FILES\VISUAL.DBF TO &MBACKDRV\VISUAL.DBF

? "Please wait.  Backing up DYNAFLECT files . . ."
IF FILE('\PAVEDB\FILES\BACKUP\DYNAFLLD.DBF')
```

```
  DELE FILE \PAVEDB\FILES\BACKUP\DYNAFLLD.DBF
ENDIF
COPY FILE \PAVEDB\FILES\DYNAFLLD.DBF TO \PAVEDB\FILES\BACKUP\DYNAFLLD.DBF
SET DEFAULT TO &MBACKDRV
MDISKSPACE = DISKSPACE()
DO WHILE MDISKSPACE < MDYNAFLLD
  ? "Not enough space on diskette in drive " + MBACKDRV
  ? "Please replace with another diskette and press any key to continue"
  WAIT " "
  MDISKSPACE = DISKSPACE()
ENDDO
SET DEFAULT TO &MDRIVE
COPY FILE \PAVEDB\FILES\DYNAFLLD.DBF TO &MBACKDRV\DYNAFLLD.DBF

? "Please wait.  Backing up SI files . . ."
IF FILE('\PAVEDB\FILES\BACKUP\SI.DBF')
  DELE FILE \PAVEDB\FILES\BACKUP\SI.DBF
ENDIF
COPY FILE \PAVEDB\FILES\SI.DBF TO \PAVEDB\FILES\BACKUP\SI.DBF
SET DEFAULT TO &MBACKDRV
MDISKSPACE = DISKSPACE()
DO WHILE MDISKSPACE < MSI
  ? "Not enough space on diskette in drive " + MBACKDRV
  ? "Please replace with another diskette and press any key to continue"
  WAIT " "
  MDISKSPACE = DISKSPACE()
ENDDO
SET DEFAULT TO &MDRIVE
COPY FILE \PAVEDB\FILES\SI.DBF TO &MBACKDRV\SI.DBF

? "Please wait.  Backing up TRAFFIC files . . ."
IF FILE('\PAVEDB\FILES\BACKUP\TRAFFIC.DBF')
  DELE FILE \PAVEDB\FILES\BACKUP\TRAFFIC.DBF
ENDIF
COPY FILE \PAVEDB\FILES\TRAFFIC.DBF TO \PAVEDB\FILES\BACKUP\TRAFFIC.DBF
SET DEFAULT TO &MBACKDRV
MDISKSPACE = DISKSPACE()
DO WHILE MDISKSPACE < MTRAFFIC
  ? "Not enough space on diskette in drive " + MBACKDRV
  ? "Please replace with another diskette and press any key to continue"
  WAIT " "
  MDISKSPACE = DISKSPACE()
ENDDO
SET DEFAULT TO &MDRIVE
COPY FILE \PAVEDB\FILES\TRAFFIC.DBF TO &MBACKDRV\TRAFFIC.DBF

? "Please wait.  Backing up WEATHER files . . ."
IF FILE('\PAVEDB\FILES\BACKUP\WEATHER.DBF')
  DELE FILE \PAVEDB\FILES\BACKUP\WEATHER.DBF
ENDIF
COPY FILE \PAVEDB\FILES\WEATHER.DBF TO \PAVEDB\FILES\BACKUP\WEATHER.DBF
SET DEFAULT TO &MBACKDRV
MDISKSPACE = DISKSPACE()
DO WHILE MDISKSPACE < MWEATHER
```

```
  ? "Not enough space on diskette in drive " + MBACKDRV
  ? "Please replace with another diskette and press any key to continue"
  WAIT " "
  MDISKSPACE = DISKSPACE()
ENDDO
SET DEFAULT TO &MDRIVE
COPY FILE \PAVEDB\FILES\WEATHER.DBF TO &MBACKDRV\WEATHER.DBF

? "Please wait.  Backing up FALLWGHT files . . ."
IF FILE('\PAVEDB\FILES\BACKUP\FALLWGHT.DBF')
  DELE FILE \PAVEDB\FILES\BACKUP\FALLWGHT.DBF
ENDIF
COPY FILE \PAVEDB\FILES\FALLWGHT.DBF TO \PAVEDB\FILES\BACKUP\FALLWGHT.DBF
SET DEFAULT TO &MBACKDRV
MDISKSPACE = DISKSPACE()
DO WHILE MDISKSPACE < MFALLWGHT
  ? "Not enough space on diskette in drive " + MBACKDRV
  ? "Please replace with another diskette and press any key to continue"
  WAIT " "
  MDISKSPACE = DISKSPACE()
ENDDO
SET DEFAULT TO  &MDRIVE
COPY FILE \PAVEDB\FILES\FALLWGHT.DBF TO &MBACKDRV\FALLWGHT.DBF

? "Please wait.  Backing up ENV files . . ."
IF FILE('\PAVEDB\FILES\BACKUP\ENV.DBF')
  DELE FILE \PAVEDB\FILES\BACKUP\ENV.DBF
ENDIF
COPY FILE \PAVEDB\FILES\ENV.DBF TO \PAVEDB\FILES\BACKUP\ENV.DBF
SET DEFAULT TO &MBACKDRV
MDISKSPACE = DISKSPACE()
DO WHILE MDISKSPACE < MENV
  ? "Not enough space on diskette in drive " + MBACKDRV
  ? "Please replace with another diskette and press any key to continue"
  WAIT " "
  MDISKSPACE = DISKSPACE()
ENDDO
SET DEFAULT TO &MDRIVE
COPY FILE \PAVEDB\FILES\ENV.DBF TO &MBACKDRV\ENV.DBF

? "Please wait.  Backing up SKID files . . ."
IF FILE('\PAVEDB\FILES\BACKUP\SKID.DBF')
  DELE FILE \PAVEDB\FILES\BACKUP\SKID.DBF
ENDIF
COPY FILE \PAVEDB\FILES\SKID.DBF TO \PAVEDB\FILES\BACKUP\SKID.DBF
SET DEFAULT TO &MBACKDRV
MDISKSPACE = DISKSPACE()
DO WHILE MDISKSPACE < MSKID
  ? "Not enough space on diskette in drive " + MBACKDRV
  ? "Please replace with another diskette and press any key to continue"
  WAIT " "
  MDISKSPACE = DISKSPACE()
ENDDO
SET DEFAULT TO  &MDRIVE
```

```
   COPY FILE \PAVEDB\FILES\SKID.DBF TO &MBACKDRV\SKID.DBF
ENDIF
SET TYPE TO 20
SET SAFETY OFF
CLOSE DATABASES
CLEAR
RETURN TO MASTER
```

APPENDIX A

County Name Table

File Name : CNTYTBL.DBF

Field	Key	Size/Type	Description
CNTYNUM	*	3N	County Number
CNTYNAME		13C	County Name

The table is sorted on the key field.

Distress vs. 18 KEAL File

File Layout

File Name : DISTRESS.DBF

Field	Key	Size/Type	File Source **	Description
SID_NO	*	4N		Section Identification Number
YEAR	*	2N		Year
A18KEAL		11N	T	Annual Cumulative 18 Keal one way
RUTT		1N	V	Rutting Area
ALLGCR		1N	V	Alligator Cracking Area
PATCH		1N	V	Patch
SIMEAN		7.5N	SI	Serviceability Index Mean
CONYEAR		2N	LI	Year of Last Major Construction
CONMON		2N	LI	Month of Last Major Construction
PAVETYP		2N	G	Pavement Type
OVRYEAR		2N	LI	Year of Last HMAC Overlay
OVRMON		2N	LI	Month of Last HMAC Overlay

** File Source

T - TRAFFIC FILE
V - VISUAL FILE
LI - LAYER IDENTIFICATION FILE (LAYER)
G - GEOMETRIC & SHOULDER INFORMATION FILE (GEOSHO)
SI - SERVICEABILITY INDEX FILE

District Temperature Constant Table

File Name : DISTTEMP.DBF

Field	Key	Size/Type	Description
DISTRICT	*	2N	District Number
TEMPCONS		2N	District Temperature Constant

The table is sorted on the key field.

Monitoring Data

Dynaflect Measurement File

File Name : DYNAFLLD.DBF

Field	Key	Size/Type	Description
SID_NO	*	4N	Section Identification Number
STRUCNUM	*	2N	Structure Number
LAYNUM	*	2N	Layer Identification Number
YEAR	*	2N	Year
MONTH	*	2N	Month
DAY		2N	Day
STATION	*	2N	Station
SENS1RD		5.3N	Sensor 1 Reading
SENS2RD		5.3N	Sensor 2 Reading
SENS3RD		5.3N	Sensor 3 Reading
SENS4RD		5.3N	Sensor 4 Reading
SENS5RD		5.3N	Sensor 5 Reading

Environmental Data

Environment Measurement File

File Name : ENV.DBF

Field	Key	Size/Type	Description
CNTYNUM	*	3N	County Number
THORNMN		S10.3N	Thornthwaite Index - Mean
THORNSD		S10.3N	Thornthwaite Index - Standard Deviation
THORNYRS		2N	Thornthwaite Index - No. Years Averaged

Monitoring Data

Falling Weight SSI File

File Name: FALLWGHT.DBF

Field	Key	Size/Type	Description
SID_NO	*	4N	Section Identification Number
STRUCNUM	*	2N	Structure Number
LAYNUM	*	2N	Layer Number
YEAR	*	2N	Year
MONTH	*	2N	Month
DAY	*	2N	Day
RWSSIAVG		4.1N	Average SSI for the Roadway
SSITEMP		3N	Temperature For the Roadway
SSIGP11		5.2N	SSI Reading 1 Geophone no. 1
SSIGP12		5.2N	SSI Reading 1 Geophone no. 2
SSIGP13		5.2N	SSI Reading 1 Geophone no. 3
SSIGP14		5.2N	SSI Reading 1 Geophone no. 4
SSIGP15		5.2N	SSI Reading 1 Geophone no. 5
SSIGP16		5.2N	SSI Reading 1 Geophone no. 6
SSIGP17		5.2N	SSI Reading 1 Geophone no. 7
SSIGP21		5.2N	SSI Reading 2 Geophone no. 1
SSIGP22		5.2N	SSI Reading 2 Geophone no. 2
SSIGP23		5.2N	SSI Reading 2 Geophone no. 3
SSIGP24		5.2N	SSI Reading 2 Geophone no. 4
SSIGP25		5.2N	SSI Reading 2 Geophone no. 5
SSIGP26		5.2N	SSI Reading 2 Geophone no. 6
SSIGP27		5.2N	SSI Reading 2 Geophone no. 7
SSIGP31		5.2N	SSI Reading 3 Geophone no. 1
SSIGP32		5.2N	SSI Reading 3 Geophone no. 2
SSIGP33		5.2N	SSI Reading 3 Geophone no. 3
SSIGP34		5.2N	SSI Reading 3 Geophone no. 4
SSIGP35		5.2N	SSI Reading 3 Geophone no. 5
SSIGP36		5.2N	SSI Reading 3 Geophone no. 6
SSIGP37		5.2N	SSI Reading 3 Geophone no. 7
SSIGP41		5.2N	SSI Reading 4 Geophone no. 1
SSIGP42		5.2N	SSI Reading 4 Geophone no. 2
SSIGP43		5.2N	SSI Reading 4 Geophone no. 3
SSIGP44		5.2N	SSI Reading 4 Geophone no. 4
SSIGP45		5.2N	SSI Reading 4 Geophone no. 5

Monitoring Data

Falling Weight SSI File (Continued)

Field	Key	Size/Type	Description
SSIGP46		5.2N	SSI Reading 4 Geophone no. 6
SSIGP47		5.2N	SSI Reading 4 Geophone no. 7
SSIGP51		5.2N	SSI Reading 5 Geophone no. 1
SSIGP52		5.2N	SSI Reading 5 Geophone no. 2
SSIGP53		5.2N	SSI Reading 5 Geophone no. 3
SSIGP54		5.2N	SSI Reading 5 Geophone no. 4
SSIGP55		5.2N	SSI Reading 5 Geophone no. 5
SSIGP56		5.2N	SSI Reading 5 Geophone no. 6
SSIGP57		5.2N	SSI Reading 5 Geophone no. 7

Functional Classification Table

File Name : FUNCLTBL.DBF

Field	Key	Size/Type	Description
CODE	*	1N	Functional Classification Code
DESCRIPT		40C	Description of Functional Classification

The table is sorted on the key field.

Inventory Data

Geometric and Shoulder Information File

File Name : GEOSHO.DBF

Field	Key	Size/Type	Description
SID_NO	*	4N	Section Identification Number
STRUCNUM	*	2N	Structure Number
PAVETYP		2N	Type of Pavement
LANEWID		4.1N	Lane Width (Feet)
OUTSHOWD		4.1N	Outside Shoulder Width (Feet)
SHOSFTYP		1N	Shoulder Surface Type
SHOBSTYP		2N	Shoulder Base Type
SHOSFTHK		5.2N	Shoulder Surface Thickness (Inch)
SHOBSTHK		5.2N	Shoulder Base Thichness (Inches)
WIDENFLG		1N	Flag for Type Of Widening

Inventory Data

Layer Identification File

File Name: LAYER.DBF

Field	Key	Size/Type	Description
SID_NO	*	4N	Section Identification Number
STRUCNUM	*	2N	Structure Number
LAYNUM	*	2N	Layer Number
LAYDESC		2N	Layer Description
CENTITHK		5.2N	Layer Center Thickness
LAYMATCL		2N	Material Type Classification
JOBCMPYR		2N	Year Job Completed
JOBCMPMO		2N	Month Job Completed
WIDENLYR		2N	Year Layer Widened
WIDENLMO		2N	Month Layer Widened

Layer Description Table

File Name : LAYERTBL.DBF

Field	Key	Size/Type	Description
CODE	*	2N	Layer Description Code
CODE_DESC		4C	Layer Description Abbreviation
CODE_LAYR		25C	Layer Description Explanation

The table is sorted on the key field.

Inventory Data

Layer Thickness Across the Road File

File Name : LAYTHICK.DBF

Field	Key	Size/Type	Description
SID_NO	*	4N	Section Identification Number
STRUCNUM	*	2N	Structure Number
LAYNUM	*	2N	Layer Number
FC3THK		5.2N	Thickness - 3rd Pos. from Center
FC2THK		5.2N	Thickness - 2nd Pos. from Center
FC1THK		5.2N	Thickness - 1st Pos. from Center
CENTTHK		5.2N	Thickness - Center
FC3DIS		4.1N	Distance from the Center/3rd Pos.
FC2DIS		4.1N	Distance from the Center/2nd Pos.
FC1DIS		4.1N	Distance from the Center/1st Pos.

Inventory Data

Location File

File Name: LOCATION.DBF

Field	Key	Size/Type	Description
SID_NO	*	4N	Section Identification Number
ACTVFLAG		1L	Active/Inactive Flag
HWYDIST		2N	SDHPT Highway District
CNTYNUM		3N	County Number
HPMSSAM		12C	HPMS Sample Number
HPMSSEC		1N	HPMS Section Subdivision
FUNCLAS		2N	Functional Classification
HWYPREFX		2C	Highway Number Prefix
HWYNUM		4N	Highway Number
HWYSUFFX		1C	Highway Number Suffix
BEGMPST		3N	Beginning Milepost
BDISSIGN		1C	Beginning Displacement Sign
BMPSTDIS		2N	Beginning Milepost Displacement
ENDMPST		3N	Ending Milepost
EDISSIGN		1C	Ending Displacement Sign
EMPSTDIS		2N	Ending Milepost Displacement
LANEID		1C	Lane Identification (R,L)
CONTROL		4N	Control
SECTION		2N	Section
BEGMPNT		6.3N	Beginning Milepoint
ENDMPNT		6.3N	Ending Milepoint
MPNTMO		2N	Milepoint Month
MPNTYR		2N	Milepoint Year
NUMLANES		2N	Number of Lanes
PREVSID		4N	Pointer to the Previous SID Number for this Location
NEXTSID		4N	Pointer to the Next SID Number for this Location
INACTYR		2N	Inactive Year
INACTMO		2N	Inactive Month
COMMENT		40C	Comments on SID Number

Material Type Table

File Name : MATLTBL.DBF

Field	Key	Size/Type	Description
MATCODE	*	2N	Code for Material Classification Type
MATDESC		20C	Description of Material Type
MATSHRT		4C	Short Form for Material Classification Type
LAYRDES		2C	Type of Layer (surface, subbase, base, subgrade)

The table is sorted on the key field.

Model File
File Layout

File Name: MODEL.DBF

Field Name	Key	Size/Type	Description
SID_NO	*	4N	Section Identification Number
HWYTYPE		2C	Highway Type
HWYDIST		2N	District
CNTYNUM		3N	County Number
PAVETYP		2N	Pavement Type
ALLG_RHO		7.4N	Alligator Rho Value
ALLG_BETA		7.4N	Alligator Beta Value
RUTT_RHO		7.4N	Rutting Rho Value
RUTT_BETA		7.4N	Rutting Beta Value
PSI_RHO		7.4N	PSI Rho Value
PSI_BETA		7.4N	PSI Beta Value
BASETYP		2N	Base Type
BASETHCK		7.2N	Base Thickness
TFTCMN		10.3N	Total Freeze Thaw Cycle Mean
PRECMN		10.3N	Inches of Precipitation
THORNMN		10.3N	Thornthwaite Index Mean
CURADT		6N	Current Average Daily Traffic
SUBGRDTY		2N	Subgrade Type
PPSV200		4.1N	Percent Passing #200 Sieve
LIQLIM		4.1N	Liquid Limit
SHOSFTYP		1N	Shoulder Surface Type
SURFTHCK		7.2N	Surface Thickness
ASAPPLRT		4.2N	Aggregate Application Rate
TEMPCONS		2N	District Temperature Constant
DYNAFLC1		5.3N	Average Dynaflect Measurement - Sensor 1
DYNAFLC2		5.3N	Average Dynaflect Measurement - Sensor 2
DYNAFLC3		5.3N	Average Dynaflect Measurement - Sensor 3
DYNAFLC4		5.3N	Average Dynaflect Measurement - Sensor 4
DYNAFLC5		5.3N	Average Dynaflect Measurement - Sensor 5
SSIGP1		5.2N	Average SSI for Geophone 1
SSIGP2		5.3N	Average SSI for Geophone 2
SSIGP3		5.3N	Average SSI for Geophone 3
SSIGP4		5.3N	Average SSI for Geophone 4
SSIGP5		5.3N	Average SSI for Geophone 5

Pavement Type Table

File Name : PAVETYPE.DBF

Field	Key	Size/Type	Description
PAVECODE	*	2N	Type of Pavement Code
BASETYPE		25C	Type of Base
BASETHK		17C	Surface Thickness
BASESEAL		12C	Surface Seal

The table is sorted on the key field.

Monitoring Data

Serviceability Index File

File Name: SI.DBF

Field	Key	Size/Type	Description
SID_NO	*	4N	Section Identification Number
STRUCNUM	*	2N	Structure Number
LAYNUM	*	2N	Layer Identification Number
YEAR	*	2N	Identifying Year
ACTMONTH		2N	Actual Month of Measurement
ACTDAY		2N	Actual Day of Measurement
ACTYEAR		2N	Actual Year of Measurement
SICOUNT		3N	Serviceability Index - Count of Observation
SIMEAN		7.5N	Serviceability Index - Mean
SISD		7.5N	Serviceability Index - Standard Deviation
SILOWVAL		3.1N	Serviceability Index - Low Value
SIHIVAL		3.1N	Serviceability Index - High Value

Monitoring Data

Skid Measuring File

File Name : SKID.DBF

Field	Key	Size/Type	Description
SID_NO	*	4N	Section Identification Number
STRUCNUM	*	2N	Structure number
LAYNUM	*	2N	Layer number
YEAR	*	2N	Year of Measurement
MONTH	*	2N	Month of Measurement
SKIDNUMM		2N	Skid Number - Mean
SKIDNUMH		2N	Skid Number - High
SKIDNUML		2N	Skid Number - Low

Inventory Data

Subgrade File

File Name : SUBGRADE.DBF

Field	Key	Size/Type	Description
SID_NO	*	4N	Section Identification Number
STRUCNUM	*	2N	Structure Number
LAYNUM	*	2N	Layer Number
PPSV200		4.1N	Percent Passing NO. 200 Sieve
PLASTIX		4.1N	Plasticity Index
LIQLIM		4.1N	Liquid Limit
TXTRIAXL		3.1N	Texas Triaxial Class
PERMIX		5.2N	Permeability Index

Inventory Data

Surface File

File Name : SURFACE.DBF

Field	Key	Size/Type	Description
SID_NO	*	4N	Section Identification Number
STRUCNUM	*	2N	Structure Number
LAYNUM	*	2N	Layer Number
AGAPPLRT		3N	Aggregate Application Rate
ADMXTYP		12C	Type Admixture
ADMXPER		5.2N	Percent Admixture
ASAPPLRT		4.2N	Asphalt Application Rate

Monitoring Data

Traffic Measurement File

File Name : TRAFFIC.DBF

Field	Key	Size/Type	Description
SID_NO	*	4N	Section Identification Number
YEAR	*	2N	Year of Measurement
AADT1WAY		6N	Annual Average Daily Traffic
A18KEAL		9N	Annual Cumulative 18KEAL one way
PCTTRK		4.1N	Percent Trucks

NOTE: 18KEAL is 18000 lbs. equivalent axle load. This figure is calculated using the AASHTO Equivalency factors which convert any weighted truck axle to a number of equivalent of 18000 lbs. single axles. The basics for the equivalency is the observed pavement damage done by different axle loads at the AASHTO Road Test (1959-1960).

Monitoring Data

Visual File

File Name: VISUAL.DBF

Field	Key	Size/Type	Description
SID_NO	*	4N	Section Identification Number
STRUCNUM	*	2N	Structure Number
LAYNUM	*	2N	Layer Number
YEAR	*	2N	Identifing Year
ACTYEAR		2N	Actual Year of Survey
ACTMONTH		2N	Actual Month of Survey
RUTTSL		1N	Rutting - Slight - Area Rating
RUTTMD		1N	Rutting / Moderate - Area Rating
RUTTSV		1N	Rutting / Severe - Area Rating
BLKCRKSL		1N	Block Crack / Slight - Area Rating
BLKCRKMD		1N	Block Crack / Moderate - Area Rating
BLKCRKSV		1N	Block Crack / Severe - Area Rating
ALLGCRSL		1N	Alligator Crack/Slight - Area Rating
ALLGCRMD		1N	AlligatorCrack/Mod- Area Rating
ALLGCRSV		1N	AlligatorCrack/Severe - Area Rating
LONGCRSL		1N	Longitudinal Crack/Slight-Area Rating
LONGCRMD		1N	Longitudinal Crack/Mod - Area Rating
LONGCRSV		1N	Longitudinal Crack/Severe-Area Rating
TRANCRSL		1N	Transverse Crack/Slight - Area Rating
TRANCRMD		1N	Transverse Crack/Mod - Area Rating
TRANCRSV		1N	Transverse Crack/Severe - Area Rating
SEALCRCD		1N	Cracks - Sealing Code
PATCHGD		1N	Patching / Good - Area Rating
PATCHFR		1N	Patching / Fair - Area Rating
PATCHPR		1N	Patching / Poor - Area Rating
FAILMILE		1N	Failures per Mile Occur. Rating
PRS		3N	Pavement Rating Score
PESPVIRS		4.2N	PES Rating Score
UVURS		4.2N	Unweighted Visual Utility Rating Score

Environmental Data

Weather Measurement File

File Name : WEATHER.DBF

Field	Key	Size/Type	Description
CNTYNUM	*	3N	County Number
MONTH	*	2N	Month number
PRECYRS		2N	Precipitation - No. Years Averaged
PRECMN		8.3N	Precipitation - Mean
PRECSD		8.3N	Precipitation - Standard Deviation
TFTCYRS		2N	Total Freeze Thaw Cycle - No. Years Averaged
TFTCMN		8.3N	Total Freeze Thaw Cycle - Mean
TFTCSD		8.3N	Total Freeze Thaw Cycle - Standard Deviation
WFTCYRS		2N	Wet Freeze Thaw Cycle - No. Years Averaged
WFTCMN		8.3N	Wet Freeze Thaw Cycle - Mean
WFTCSD		8.3N	Wet Freeze Thaw Cycle - Standard Deviation
MTMPYRS		2N	Maximum Temperature - No. Years Averaged
MTMPMN		8.3N	Maximum Temperature - Mean
MTMPSD		8.3N	Maximum Temperature - Standard Deviation
ATMPYRS		2N	Averaged Temperature - No. Years Averaged
ATMPMN		8.3N	Averaged Temperature - Mean
ATMPSD		8.3N	Averaged Temperature - Standard Deviation

Widening Flag Table

File Name : WIDENFLG.DBF

Field	Key	Size/Type	Description
WIDNCODE		1N	Code for Widening Flag
WIDNDESC		26C	Description for the Widening Flag Code

The table is sorted on the key field.

APPENDIX B

TEXAS FLEXIBLE PAVEMENT DATABASE
VARIABLE DESCRIPTION FOR ALL FILES

Variable Name	Source File	Explanation
A18KEAL	T	Traffic Measurement File
AADT1WAY	T	Traffic Measurement File
ACTDAY	S	Actual Day of Measurement
ACTMONTH	S	Actual Month of Measurement
ACTVFLAG	L	Active/Inactive Flag
ACTYEAR	S	Actual Year of Measurement
ADMXPER	R	Percent Admixture
ADMXTYP	R	Type Admixture
AGAPPLRT	R	Aggregate Application Rate
ASAPPLRT	R	Asphalt Application Rate
BDISSIGN	L	Beginning Displacement Sign
BEGMPNT	L	Beginning Milepoint
BEGMPST	L	Beginning Milepost
BMPSTDIS	L	Beginning Milepost Displacement
CENTTHK	HI	Thickness - Center
CNTYNUM	LWE	County Number
CONTROL	L	Control
DAY	FD	Day
EDISSIGN	L	Ending Displacement Sign
EMPSTDIS	L	Ending Milepost Displacement
ENDMPNT	L	Ending Milepoint
ENDMPST	L	Ending Milepost
FC1DIS	H	Distance from the Center/1st Pos.
FC1THK	H	Thickness - 1st Pos. from Center
FC2DIS	H	Distance from the Center/2nd Pos.
FC2THK	H	Thickness - 2nd Pos. from Center
FC3DIS	H	Distance from the Center/3rd Pos.
FC3THK	H	Thickness - 3rd Pos. from Center
FUNCLAS	L	Functional Classification
HPMSSAM	L	HPMS Sample Number
HPMSSEC	L	HPMS Section Subdivision
HWYDIST	L	SDHPT Highway District
HWYNUM	L	Highway Number
HWYPREFX	L	Highway Number Prefix
HWYSUFFX	L	Highway Number Suffix
INACTMO	L	Inactive Month
INACTYR	L	Inactive Year
JOBCMPMO	I	Month Job Completed
JOBCMPYR	I	Year Job Completed
LANEID	L	Lane Identification (R,L)
LANEWID	G	Lane Width (Feet)
LAYDESC	I	Layer Description
LAYMATCL	I	Material Type Classification
LAYNUM	FDHIRBSVK	Layer Number
LIQLIM	B	Liquid Limit

Variable Name	Source File	Explanation
MONTH	FDWK	Month
MPNTMO	L	Milepoint Month
MPNTYR	L	Milepoint Year
NEXTSID	L	Pointer to the Next SID Number for this location
NO_LANES	G	Number Of Lanes - One Way
NUMLANES	L	Number Of Lanes
OUTSHOWD	G	Outside Shoulder Width (Feet)
PAVETYP	G	Type of Pavement
PCTTRK	T	Traffic Measurement File
PERMIX	B	Permeability Index
PLASTIX	B	Plasticity Index
PPSV200	B	Percent Passing NO. 200 Sieve
PREVSID	L	Pointer to the Previous SID Number for this location
RWSSIAVG	F	Average SSI for the Roadway
SECTION	L	Section
SENS1RD	D	Sensor 1 Reading
SENS2RD	D	Sensor 2 Reading
SENS3RD	D	Sensor 3 Reading
SENS4RD	D	Sensor 4 Reading
SENS5RD	D	Sensor 5 Reading
SHOBSTHK	G	Shoulder Base Thickness (Inches)
SHOBSTYP	G	Shoulder Base Type
SHOSFTHK	G	Shoulder Surface Thickness (Inch)
SHOSFTYP	G	Shoulder Surface Type
SICOUNT	S	Serviceability Index - Count of Observation
SID_NO	FDHIGRBLSKT	Section Identification Number
SIHIVAL	S	Serviceability Index - High Value
SILOWVAL	S	Serviceability Index - Low Value
SIMEAN	S	Serviceability Index - Mean
SISD	S	Serviceability Index - Standard Deviation
SKIDNUMH	K	Skid Measurement File
SKIDNUML	K	Skid Measurement File
SKIDNUMN	K	Skid Measurement File
SSIGP11	F	SSI Reading 1 Geophone no. 1
SSIGP12	F	SSI Reading 1 Geophone no. 2
SSIGP13	F	SSI Reading 1 Geophone no. 3
SSIGP14	F	SSI Reading 1 Geophone no. 4
SSIGP15	F	SSI Reading 1 Geophone no. 5
SSIGP16	F	SSI Reading 1 Geophone no. 6
SSIGP17	F	SSI Reading 1 Geophone no. 7
SSIGP21	F	SSI Reading 2 Geophone no. 1
SSIGP22	F	SSI Reading 2 Geophone no. 2
SSIGP23	F	SSI Reading 2 Geophone no. 3
SSIGP24	F	SSI Reading 2 Geophone no. 4
SSIGP25	F	SSI Reading 2 Geophone no. 5
SSIGP26	F	SSI Reading 2 Geophone no. 6
SSIGP27	F	SSI Reading 2 Geophone no. 7

Variable Name	Source File	Explanation
SSIGP31	F	SSI Reading 3 Geophone no. 1
SSIGP32	F	SSI Reading 3 Geophone no. 2
SSIGP33	F	SSI Reading 3 Geophone no. 3
SSIGP34	F	SSI Reading 3 Geophone no. 4
SSIGP35	F	SSI Reading 3 Geophone no. 5
SSIGP36	F	SSI Reading 3 Geophone no. 6
SSIGP37	F	SSI Reading 3 Geophone no. 7
SSIGP41	F	SSI Reading 4 Geophone no. 1
SSIGP42	F	SSI Reading 4 Geophone no. 2
SSIGP43	F	SSI Reading 4 Geophone no. 3
SSIGP44	F	SSI Reading 4 Geophone no. 4
SSIGP45	F	SSI Reading 4 Geophone no. 5
SSIGP46	F	SSI Reading 4 Geophone no. 6
SSIGP47	F	SSI Reading 4 Geophone no. 7
SSIGP51	F	SSI Reading 5 Geophone no. 1
SSIGP52	F	SSI Reading 5 Geophone no. 2
SSIGP53	F	SSI Reading 5 Geophone no. 3
SSIGP54	F	SSI Reading 5 Geophone no. 4
SSIGP55	F	SSI Reading 5 Geophone no. 5
SSIGP56	F	SSI Reading 5 Geophone no. 6
SSIGP57	F	SSI Reading 5 Geophone no. 7
SSITEMP	F	Temperature for the Roadway
STATION	D	Station
STRUCNUM	FDHIGRBSVK	Structure Number
TXTRIAXL	B	Texas Triaxial Class
WIDENFLG	G	Type Of Widening Flag
WIDENLMO	I	Month Layer Widened
WIDENLYR	I	Year Layer Widened
YEAR	FDTK	Year
YEAR	SV	Identifing Year

** Source File Key

F - Falling Weight SSI
D - Dynaflect Measurement
H - Layer Thickness Across the Road
I - Layer Identification
G - Geometric & Shoulder
R - Surface
B - Subgrade
L - Location
W - Weather
E - Environment
S - Serviceability Index
V - Visual
T - Traffic
K - Skid

APPENDIX C

SYSTEM CONFIGURATION

The following are the requirements of the Texas Flexible Pavement System:

Hardware requirements:

- IBM Personal Computer XT or Compatible
- 640K bytes of RAM
- 360K Floppy Disk Drive
- 20 Megabyte Hard Disk
- Printer (able to condensed print - 133 columns)
- Monochrome or Color Monitor

Software requirements:

- DOS 3.3 or later
- dBASE III Plus (By Aston Tate)
- TURBO Pascal 3.0 (For source code modification)

Space requirements:

- 20 Megabytes of storage space.

APPENDIX D

INSTALLATION

The installation is accomplished by the program INSTALL.BAT. It creates the appropriate subdirectories for the Flexible Pavement System and then calls program SETUP.BAT. Program SETUP.BAT unarchives the files from 8 floppy diskettes onto the hard disk. The unarchive program PKXARC.COM by PKWARE INC. is used to unarchive programs. The program is activated from the hard disk prompt by typing "A:INSTALL".

The 8 floppy diskettes contain the following files:

DISK 1 - BACKUP.ARC
INQUIRY.ARC
REPORTS.ARC
PAVEDB.ARC
PKXARC.COM
FLEXPAVE.BAT
INSTALL.BAT
SETUP.BAT

DISK 2 - EDITUPDT.ARC
E_DIST.ARC

DISK 3 - E_TRAFFI.ARC

DISK 4 - E_PES.ARC

DISK 5 - APPLICAT.ARC
A_DIST.ARC
A_MODEL.ARC
A_GRAPH.ARC

DISK 6 - FILES1.ARC

DISK 7 - FILES2.ARC

DISK 8 - INDEXES.ARC

```
ECHO OFF
REM SUBSYSTEM:       PAVEDB
REM PROGRAM NAME:    INSTALL.BAT     10/27/88
REM PROJECT 2456 - TEXAS FLEXIBLE PAVEMENT DATABASE CONVERSION
REM TAMU/TTI
REM AUTHOR:          TREVOR X. PEREIRA
REM PURPOSE:         TO INSTALL THE FLEIBLE PAVEMENT DATABASE SYSTEM
REM                  ON THE HARD DISK
CLS
ECHO '
ECHO '
ECHO '
REM
REM asks user if he wants to install flexible pavement system
REM
ECHO You are about to install the Flexible Pavement system on
ECHO your hard disk under \PAVEDB.
ECHO If you already have the system installed on your computer,
ECHO stop this operation by pressing ctrl+break simultaneously.
ECHO '
ECHO '
ECHO If you do not want to continue, press ctrl+break simultaneously.
ECHO If you wish to continue, press any key.
PAUSE
CLS
ECHO '
ECHO '
ECHO '
ECHO '
ECHO '
ECHO '
ECHO '
ECHO '
ECHO '
ECHO Please wait.  Creating subdirectories.
CD \
MD \PAVEDB
MD \PAVEDB\APPLICAT
MD \PAVEDB\APPLICAT\DISTRESS
MD \PAVEDB\APPLICAT\GRAPH18K
MD \PAVEDB\APPLICAT\MODEL
MD \PAVEDB\BACKUP
MD \PAVEDB\EDITUPDT
MD \PAVEDB\EDITUPDT\BACKUP
MD \PAVEDB\EDITUPDT\DISTRESS
MD \PAVEDB\EDITUPDT\PES
MD \PAVEDB\EDITUPDT\TRAFFIC
MD \PAVEDB\FILES
MD \PAVEDB\FILES\BACKUP
MD \PAVEDB\INDEXES
```

```
MD \PAVEDB\INQUIRY
MD \PAVEDB\REPORTS
CD \

CLS
copy a:setup.bat
setup.bat
```

PROGRAM LISTING

```
REM
REM SUBSYSTEM:            PAVEDB
REM PROGRAM NAME:         SETUP.BAT      010/04/88
REM CALLED FROM:          INSTALL.BAT
REM PROJECT 2456 - TEXAS FLEXIBLE PAVEMENT DATABASE CONVERSION
REM TAMU/TTI
REM AUTHOR:               TREVOR X. PEREIRA
REM PURPOSE:              TO SET UP THE FLEXIBLE PAVEMENT SYSTEM ON THE
REM                       USERS HARD DISK.
REM

echo off
ECHO '
ECHO '
ECHO '
ECHO '
ECHO '
ECHO '
ECHO '
ECHO '
ECHO '
ECHO Please wait

CD \
COPY A:PKXARC.COM
CD \PAVEDB
PKXARC A:PAVEDB
CD \PAVEDB\INQUIRY
PKXARC A:INQUIRY
CD \PAVEDB\REPORTS
PKXARC A:REPORTS
CD \PAVEDB\BACKUP
PKXARC A:BACKUP

CLS
ECHO '
ECHO '
ECHO '
ECHO '
ECHO '
ECHO '
ECHO '
ECHO '
ECHO '
ECHO Please insert FLEXPAVE DISK 2
PAUSE
CD \PAVEDB\EDITUPDT
PKXARC A:EDITUPDT
CD \PAVEDB\EDITUPDT\DISTRESS
PKXARC A:E_DIST
```

```
CLS
ECHO '
ECHO '
ECHO '
ECHO '
ECHO '
ECHO '
ECHO '
ECHO '
ECHO '
ECHO Please insert FLEXPAVE DISK 3
PAUSE
CD \PAVEDB\EDITUPDT\TRAFFIC
PKXARC A:E_TRAFFIC

CLS
ECHO '
ECHO '
ECHO '
ECHO '
ECHO '
ECHO '
ECHO '
ECHO '
ECHO '
ECHO Please insert FLEXPAVE DISK 4
PAUSE
CD \PAVEDB\EDITUPDT\PES
PKXARC A:E_PES

CLS
ECHO '
ECHO '
ECHO '
ECHO '
ECHO '
ECHO '
ECHO '
ECHO '
ECHO '
ECHO Please insert FLEXPAVE DISK 5
PAUSE
CD \PAVEDB\APPLICAT
PKXARC A:APPLICAT
CD \PAVEDB\APPLICAT\DISTRESS
PKXARC A:A_DIST
CD \PAVEDB\APPLICAT\GRAPH18K
PKXARC A:A_GRAPH
CD \PAVEDB\APPLICAT\MODEL
PKXARC A:A_MODEL

CLS
ECHO '
```

```
ECHO '
ECHO '
ECHO '
ECHO '
ECHO '
ECHO '
ECHO '
ECHO '
ECHO Please insert FLEXPAVE DISK 6
PAUSE
CD \PAVEDB\FILES
PKXARC A:FILES1

CLS
ECHO '
ECHO '
ECHO '
ECHO '
ECHO '
ECHO '
ECHO '
ECHO '
ECHO '
ECHO Please insert FLEXPAVE DISK 7
PAUSE
PKXARC A:FILES2

CLS
ECHO '
ECHO '
ECHO '
ECHO '
ECHO '
ECHO '
ECHO '
ECHO '
ECHO '
ECHO Please insert FLEXPAVE DISK 8
PAUSE
CD \PAVEDB\INDEXES
PKXARC A:INDEXES

CD \
DEL PKXARC.COM
DEL \PAVEDB\INSTALL.MEM

CLS
ECHO '
ECHO '
ECHO '
ECHO '
ECHO '
ECHO '
ECHO '
```

```
ECHO '
ECHO '
ECHO INSTALLATION COMPLETE ! ! !
ECHO TYPE "FLEXPAVE" TO START SYSTEM AND SET DEFAULTS
cd \pavedb
ECHO ON
```