VA FileMan 22.2

Technical Manual

November 2021

Department of Veterans Affairs (VA)

Office of Information and Technology (OIT)

Development, Security, and Operations (DSO)

Revision History

Date	Revision	Description	Author
11/10/2021	1.8	Added DDE* components to the following sections: • Section 4.1, "Routines and Callable Entry Points:" Added **DDE*** routine descriptions and added **DDEPRT** routine to Table 5. • Section 4.6, "VA FileMan Options:" Added **DDE*** options attached to the **Other Options** [DIOTHER] menu branch in Figure 3: ○ **Entity Mapping** [DDE ENTITY MAPPING] Menu. ○ **Auto Gen Entity for a DD #** [DDE AUTO GEN ENTITY FOR A DD #] Option. ○ **Entity Enter/Edit** [DDE ENTITY ENTER/EDIT] Option. ○ **Inquire to Entity File** [DDE ENTITY INQUIRE] Option.	VistA Kernel (VistA Infrastructure [VI]) Development Team
02/04/2019	1.7	Updated Section 4.5 to add menu option numbers.	VistA Kernel (VistA Infrastructure [VI]) Development Team
01/30/2019	1.6	Changes for Patch DI*22.2*9: • Added ENTITY (#1.5) file global in Table 3 and Section 10. • Added ENTITY (#1.5) file and description in Table 4. • Added DDE routines to Table 5. • Updated Section 4.5 to add DDE menu and submenu options. • Added Section 5.18 with a list of cross-references for the ENTITY (#1.5) file.	VistA Kernel (VistA Infrastructure [VI]) Development Team
10/01/2018	1.5	Reviewer Feedback Edits: • Added the **DDERR** routine to Table 5. • Added field numbers to Table 23.	VistA Kernel (VistA Infrastructure [VI]) Development Team

Date	Revision	Description	Author
05/16/2018	1.4	Updated Figure 3: VA FileMan Exported Options Diagrams in Section 4.5 - VA FileMan with Kernel. Removed references to the following options, since they have been deleted with Patch DI*22.2*10: • DI DATA TYPE OPTIONS • DI DATA TYPE FILE • DI DATA TYPE METHOD FILE • DI DATA TYPE PROPERTY FILE	VistA Kernel (VistA Infrastructure [VI]) Development Team
08/07/2017	1.3	Changes for patch DI*22.2*6 (Data Sync functionality)	VistA Kernel (VistA Infrastructure [VI]) Development Team
08/07/2017	1.2	Tech edits for patch DI*22.2*8, Data Access Control (DAC): • Added new Data Access Control (DAC) files to the "Files" section. • Added new DAC routines to the "Routines, Application Programming Interfaces (APIs), and Options" section. • Added new DAC options to the "VA FileMan Options" section. • Reformatted display of file and field names throughout; moved file/field number immediately following the file/field name.	VistA Kernel (VistA Infrastructure [VI]) Development Team

Date	Revision	Description	Author
01/17/2017	1.1	Changes for patch DI*22.2*2: • Updated Table 3: VA FileMan Routine Global References in Orientation section. Added ^DIT. • Updated Table 4: VA FileMan File List in Section 3 – Files. Added .86, .87, 1.71 and 1.72., and updated .9. • Updated Figure 2: VA FileMan Pointer Map in Section 3.1 – Pointer Map. Added .86 and .87. • Updated Table 5: VA FileMan Routines and Callable Entry Points in Section 4 – Routines and Callable Routines/Entry Points/Application Programming Interfaces (APIs). Added DDPA2, DDSRP, DICATTD8, DICATTUD, DIETLIB, DIFMEDT1, DITIME, DIUTC, and updated DDD and DIALOGZ. • Updated Figure 3: VA FileMan Exported Options Diagrams in Section 4.5 - VA FileMan with Kernel. Added DI DATA TYPE OPTIONS. • Update global list in Section 10 Globals. Added **^DIT**.	VA FileMan 23.0 Development Team
08/03/2016	1.0	Initial release of VA FileMan 22.2 Release Notes.	VA FileMan 22.2 Development Team

REF: For the current patch history related to this software, see the Patch Module (i.e., Patch User Menu [A1AE USER]) on FORUM.

Table of Contents

List of Figures

List of Tables

Orientation

What is VA FileMan?

VA FileMan is the database management system for the Veterans Health Information Systems and Technology Architecture user (VistA) environment. VA FileMan creates and maintains a database management system that includes features such as:

- Report writer
- Data dictionary manager
- Scrolling and screen-oriented data entry
- Text editors
- Programming utilities
- Tools for sending data to other systems
- File archiving

VA FileMan can be used as a:

- Standalone database
- Set of interactive or "silent" routines
- Set of application utilities; in all modes

It is used to define, enter, and retrieve information from a set of computer-stored files, each of which is described by a data dictionary.

VA FileMan is a public domain software package that is developed and maintained by the Department of Veterans Affairs (VA). It is widely used by VA medical centers and in clinical, administrative, and business settings in this country and abroad.

CAUTION: Programmer access in VistA is defined as DUZ(0)="@". It grants the privilege to become a developer in VistA. Programmer access allows you to work outside many of the security controls enforced by VA FileMan, enables access to all VA FileMan files, access to modify data dictionaries, etc. It is important to *proceed with caution* when having access to the system in this way.

How to Use this Manual

The *VA FileMan Technical Manual* provides information about the technical structure of VA FileMan. It includes the following information about VA FileMan:

- Implementation and Maintenance
- Files
- Routines, Application Programming Interfaces (APIs), and Options
- Cross-References
- Archiving and Purging
- External Relationships
- Internal Relationships
- Package-Wide Variables
- Globals
- Security

REF: For VA FileMan installation instructions in the VistA environment, see the *VA FileMan Installation Guide* and any national patch description of the patch being released.

Intended Audience

The intended audience of this manual is all key stakeholders. The stakeholders include the following: It also contains material specifically intended for VA's Veterans Health Information Systems and Technology Architecture (VistA) systems managers and application developers.

- System Administrators—System administrators at Department of Veterans Affairs (VA) sites who are responsible for computer management and system security on the VistA M Servers.
- Enterprise Program Management Office (EPMO)—VistA development teams.
- Product Support (PS).

Disclaimers

Software Disclaimer

This software was developed at the Department of Veterans Affairs (VA) by employees and contractors of the Federal Government in the course of their official duties with significant input from the larger open source community. Pursuant to title 17 Section 105 of the United States Code this software is *not* subject to copyright protection and is in the public domain. VA assumes no responsibility whatsoever for its use by other parties, and makes no guarantees, expressed or implied, about its quality, reliability, or any other characteristic. We would appreciate acknowledgement if the software is used. This software can be redistributed and/or modified freely provided that any derivative works bear some notice that they are derived from it, and any modified versions bear some notice that they have been modified.

CAUTION: To protect the security of VistA systems, distribution of this software for use on any other computer system by VistA sites is prohibited. All requests for copies of this software for *non*-VistA use should be referred to the VistA site's local Office of Information Field Office (OIFO).

Documentation Disclaimer

This manual provides an overall explanation of VA FileMan and the functionality contained in VA FileMan 22.2; however, no attempt is made to explain how the overall VistA programming system is integrated and maintained. Such methods and procedures are documented elsewhere. We suggest you look at the various VA Internet and Intranet Websites for a general orientation to VistA. For example, visit the Office of Information and Technology (OIT) VistA Development Intranet website.

DISCLAIMER: The appearance of any external hyperlink references in this manual does *not* constitute endorsement by the Department of Veterans Affairs (VA) of this Website or the information, products, or services contained therein. The VA does *not* exercise any editorial control over the information you find at these locations. Such links are provided and are consistent with the stated purpose of this VA Intranet Service.

Documentation Conventions

This manual uses several methods to highlight different aspects of the material:

- Various symbols are used throughout the documentation to alert the reader to special information. Table 1 gives a description of each of these symbols:

Table 1: Documentation Symbol Descriptions

Symbol	Description
	**NOTE / REF:** Used to inform the reader of general information including references to additional reading material.
	**CAUTION / RECOMMENDATION / DISCLAIMER:** Used to caution the reader to take special notice of critical information.

- Descriptive text is presented in a proportional font (as represented by this font).
- Conventions for displaying TEST data in this document are as follows:
  - The first three digits (prefix) of any Social Security Numbers (SSN) begin with either "**000**" or "**666**".
  - Patient and user names are formatted as follows:
    - <*Application Name/Abbreviation/Namespace*>PATIENT,[*N*] and
    - <*Application Name/Abbreviation/Namespace*>USER,[*N*]

    Where "<*Application Name/Abbreviation/Namespace*>" is defined in the Approved Application Abbreviations document and "*N*" represents the first name as a number value or spelled out and incremented with each new entry. For example, in VA FileMan (FM) test patient and user names would be documented as follows:
    - FMPATIENT,ONE; FMPATIENT,TWO; FMPATIENT,THREE; FMPATIENT,14, etc.
    - FMUSER,ONE; FMUSER,TWO; FMUSER,THREE; FMUSER,14, etc.
- "Snapshots" of computer online displays (i.e., screen captures/dialogues) and computer source code, if any, are shown in a *non*-proportional font and enclosed within a box.
  - User's responses to online prompts are **bold** typeface and highlighted in yellow (e.g., **<Enter>**).
  - Emphasis within a dialogue box is **bold** typeface and highlighted in blue (e.g., STANDARD LISTENER: RUNNING).
  - Some software code reserved/key words are **bold** typeface with alternate color font.

- References to "**<Enter>**" within these snapshots indicate that the user should press the **Enter** key on the keyboard. Other special keys are represented within < > angle brackets. For example, pressing the **PF1** key can be represented as pressing **<PF1>**.
- Author's comments are displayed in italics or as "callout" boxes.

  NOTE: Callout boxes refer to labels or descriptions usually enclosed within a box, which point to specific areas of a displayed image.

- All uppercase is reserved for the representation of M code, variable names, or the formal name of options, field/file names, and security keys (e.g., DIEXTRACT).

  NOTE: Other software code (e.g., Delphi/Pascal and Java) variable names and file/folder names can be written in lower or mixed case (e.g., CamelCase).

Documentation Navigation

This document uses Microsoft® Word's built-in navigation for internal hyperlinks. To add **Back** and **Forward** navigation buttons to your toolbar, do the following:

1. Right-click anywhere on the customizable Toolbar in Word (*not* the Ribbon section).
2. Select **Customize Quick Access Toolbar** from the secondary menu.
3. Select the drop-down arrow in the "Choose commands from:" box.
4. Select **All Commands** from the displayed list.
5. Scroll through the command list in the left column until you see the **Back** command (green circle with arrow pointing left).
6. Select/Highlight the **Back** command and select **Add** to add it to your customized toolbar.
7. Scroll through the command list in the left column until you see the **Forward** command (green circle with arrow pointing right).
8. Select/Highlight the Forward command and select **Add** to add it to your customized toolbar.
9. Select **OK**.

You can now use these **Back** and **Forward** command buttons in your Toolbar to navigate back and forth in your Word document when clicking on hyperlinks within the document.

NOTE: This is a one-time setup and is automatically available in any other Word document once you install it on the Toolbar.

VA FileMan Coding Conventions

Non-Standard M Features

Z-commands and **Z**-functions are avoided throughout VA FileMan routines. For certain purposes (e.g., allowing terminal breaking and spooling to a Standard Disk Processor [SDP] disk device), VA FileMan executes lines of *non*-standard M code out of the MUMPS OPERATING SYSTEM (#.7) file. The *non*-standard code used (if any) depends on the answer to the prompt:

Figure 1: Type of M System Prompt

```
TYPE OF MUMPS SYSTEM YOU ARE USING:
```

This prompt appears during the **DINIT** initialization routine. Answering **OTHER** to this question ensures that VA FileMan uses only standard M code.

NOTE: When installed with the VA's KIDS build, use of the Caché operating is assumed. You will not see the TYPE OF MUMPS SYSTEM YOU ARE USING: prompt.

VA FileMan also makes use of *non*-standard M code that is stored in the **%ZOSF** global:

- If VA FileMan is installed on a system that contains Kernel, it uses the **%ZOSF** global created by Kernel.
- If it is being used without Kernel (i.e., standalone), the necessary **%ZOSF** nodes are set for many operating systems by running **DINZMGR** in the Manager account.

REF: For details, see the "System Management" section in the *VA FileMan Advanced User Manual*.

String-valued subscripts (up to **30** characters long) are used extensively but only in the **$ORDER** collating sequence approved by the MUMPS Development Committee (MDC). Non-negative integer and fractional canonic numbers collate ahead of all other strings.

The **$ORDER** function is used at several points in VA FileMan's code. VA FileMan routines assume that reference to an undefined global subscript level sets the naked indicator to that level, rather than leaving it undefined. In all other respects, the VA FileMan code conforms to the 1995 ANSI Standard for the M language with Type **A** extensions.

Routine, Variable, and Global Names

In keeping with the convention that all programs that are a part of the same application or utility package should be namespaced, all VA FileMan routine names begin with **DI**, **DD**, or **DM**. (The "Device Handling for Standalone VA FileMan" section in the VA *FileMan Advanced User Manual* explains that some **DI*** routines are renamed in the Manager account.) The **DINIT** routine initializes VA FileMan. The **DINIT** routine is run automatically with no user interaction during the KIDS install. The **DI** routine itself is the main option reader.

REF: For more information on the DI routine, see the "^DI: Programmer Access" section in the *VA FileMan Developer's Guide*.

Except in **DI**, the routines do *not* contain unargumented or exclusive **KILL** commands. Most multi-character local variable names created by VA FileMan routines begin with **%** or the letter **D** or consist of one uppercase letter followed by one numeral [except that **IO(0)**, by convention, contains the **$I** value of the signon device]. Since VA FileMan uses single character variable names extensively, do *not* use them in code that is executed from within VA FileMan programming hooks unless their use is documented in the hook's description or you NEW them. Also, do *not* expect single character variables to return unchanged after calling a VA FileMan entry point.

Table 2 lists the local variables that are of special importance in the VA FileMan routines:

Table 2: VA FileMan Routine Variables and Default Values

Variable	Description	Default Value
DT	If defined, it is assumed to be the current date. For example: June 1, 1987 is DT=2870601.	Today's date; derived from **$H**
DTIME	If defined, it is the integer value of the number of seconds the user *must* respond to a timed read.	**300**
DUZ	If defined, it is assumed to be the User Number; a positive number uniquely identifying the current user.	**0**
DUZ(0)	If defined, it is assumed to be the FileMan Access Code, which is a character string describing the user's security clearance with regard to files, templates, and data fields within a file. **REF:** See the "Data Security" section in the *VA FileMan Advanced User Manual*. Setting **DUZ(0)** equal to the at-sign (**@**) overrides all security checks and allows special programmer features that are described later. If the user's M	""

Variable	Description	Default Value
	implementation supports terminal break, a developer is allowed to break execution at any point, whereas a user who does *not* have programmer access can only break during output routines.	
U	If defined, it is equal to a single caret (**^**) character.	**^**

VA FileMan routines explicitly refer to the globals in Table 3:

Table 3: VA FileMan Routine Global References

Global	Description
^DD	All attribute dictionaries, Keys, Functions, and MUMPS operating systems.
^DDA	Data dictionary audit trail.
^DDD	Meta Data Dictionary.
^DDE	Entity File
^DI	Data types, Languages, Dialogs.
^DIA	Data audit trail.
^DIAR	Archival activity and Filegrams.
^DIBT	Sort templates and the results of file searches.
^DIC	Dictionary of files.
^DIE	Input templates.
^DIPT	Print templates and Filegram templates.
^DIST	ScreenMan forms and blocks, Import Templates, Foreign Formats, and Alternate Editors.
^DISV	Most recent lookup value in any file or subfile (by **DUZ**).
^DIT	Files needed for UTC Data Type.
^DIZ	Default location for new data files as they are created.
^DOPT	Option lists.
^DOSV	Statistical results.
^%ZOSF	M vendor-specific executable code.

The routines use the **^UTILITY** and **^TMP** globals for temporary scratch space. The **^XUTL** global is also used if you are running some M implementations.

Delimiters within Strings

The caret (^) character is conventionally used to delimit data elements that are strung together to be stored in a single global node. A corollary of this rule is that the routines almost never allow input data to contain carets; the user types a caret (^) to change or terminate the sequence of questions being asked. Within ^-pieces, semicolons (**;**) are usually used as secondary delimiters, and colons (**:**) as tertiary delimiters.

VA FileMan routines use the local variable **U** as equal to the single caret (^) character.

Canonic Numbers

VA FileMan recognizes only canonic numbers. A canonic number is a number that does *not* begin or end with meaningless zeroes. For example, **7** is a canonic number, whereas **007** and **7.0** are *not* canonic numbers.

How to Obtain Technical Information Online

Exported VistA M Server-based software file, routine, and global documentation can be generated through the use of Kernel, MailMan, and VA FileMan utilities.

NOTE: Methods of obtaining specific technical information online are indicated where applicable under the appropriate section.

Help at Prompts

VistA M Server-based software provides online help and commonly used system default prompts. Users are encouraged to enter question marks at any response prompt. At the end of the help display, you are immediately returned to the point from which you started. This is an easy way to learn about any aspect of the software.

Obtaining Data Dictionary Listings

Technical information about VistA M Server-based files and the fields in files is stored in data dictionaries (DD). You can use the List File Attributes option [DILIST] on the Data Dictionary Utilities menu [DI DDU] in VA FileMan to print formatted data dictionaries.

REF: For details about obtaining data dictionaries and about the formats available, see the "List File Attributes" section in the "File Management" section in the *VA FileMan Advanced User Manual*.

Assumptions

This manual is written with the assumption that the reader is familiar with the following:

- VistA computing environment:
  - Kernel—VistA M Server software
  - VA FileMan data structures and terminology—VistA M Server software
- Microsoft® Windows environment
- M programming language

Reference Materials

Readers who wish to learn more about VA FileMan should consult the following documents:

- *VA FileMan Release Notes*
- *VA FileMan Installation Guide*
- *VA FileMan Technical Manual* (this manual)
- *VA FileMan User Manual* (PDF and HTML format)
- *VA FileMan Advanced User Manual* (PDF and HTML format)
- *VA FileMan Developer's Guide* (PDF and HTML format)

REF: Zip files of the VA FileMan documentation in HTML format are located on the VA FileMan Intranet Product website and VDL at: http://www.va.gov/vdl/application.asp?appid=5.

Using a Web browser, open the **HTML** documents "table of contents" page (i.e., index.shtml). The *VA FileMan User Manual*, the *VA FileMan Advanced User Manual*, and the *VA FileMan Developer's Guide* are all linked together.

VistA documentation is made available online in Microsoft® Word format and in Adobe® Acrobat Portable Document Format (PDF). The PDF documents *must* be read using the Adobe® Acrobat Reader, which is freely distributed by Adobe® Systems Incorporated at: http://www.adobe.com/

VistA software documentation can be downloaded from the VA Software Document Library (VDL) at: http://www.va.gov/vdl/

REF: VA FileMan manuals are located on the VDL at: http://www.va.gov/vdl/application.asp?appid=5

VistA documentation and software can also be downloaded from the Product Support (PS) Anonymous Directories.

1 Introduction

VA FileMan is a database management system (DBMS) consisting of computer routines written in American National Standards Institute (ANSI) Standard M, along with associated files. Developed with portability as a goal, VA FileMan runs on all major implementations of ANSI M and on hardware platforms ranging from PCs to mainframes.

Developers and non-developers use VA FileMan alike. VA FileMan can be used as a standalone database or as a set of application utilities. In either mode, it is used to define, enter, and retrieve information from a set of computer-stored files, each of which is described by the data dictionary.

VA FileMan is a public domain software package and is widely used in clinical, administrative, and business settings in the United States and abroad.

2 Implementation and Maintenance

VA FileMan 22.2 is initialized by an install using the Kernel Distribution and Installation system (KIDS) as directed in the *VA FileMan Installation Guide*. In previous versions **DINIT** was used to initialize VA FileMan. Now, **DINIT** is run automatically with no user intervention during the KIDS install. **DINIT** should *not* be run from the command line after the KIDS install is done. Standalone VA FileMan installs on systems without Kernel is not addressed by this documentation.

VA FileMan routines and globals occupy approximately **3.5 MB** of disk space. The size of the globals, particularly those that store application data, increases when VA FileMan is used.

Since VA FileMan provides the DBMS upon which all files in Veterans Health Information Systems and Technology Architecture (VistA) are based, it *must* be present on all VistA systems. The current version of VA FileMan is designed for complete backward compatibility; files and applications developed under prior versions remain usable.

If used with Kernel, all or part of the VA FileMan options can be given to users. Those who are able to use programmer mode can also invoke the main menu from the M prompt. Anyone can use applications developed with VA FileMan, whether or not direct access to VA FileMan itself is allowed.

REF: For more information on programmer mode, see the "^DI: Programmer Access" section in the "Developer's Tools" section in the *VA FileMan Developer's Guide*.

When used with Kernel, VA FileMan allows the user to print multiple copies. In order to do this, a temporary storage location *must* be allocated on the system with a corresponding DEVICE (#3.5) file entry that uses a sequential disk processor (SDP) device type.

REF: The *Kernel 8.0 and Kernel Toolkit 7.3 Systems Management Guide* contains specific instructions on how to set up an SDP device for different operating systems.

The **^DISV** global contains the most recent lookup value for files and subfiles; it is used to process **<Spacebar><Enter>** input. The **^DOSV** global contains results of statistical operations. These globals can grow to considerable size and should be monitored. It is safe to periodically **KILL** these globals. Users should *not* be logged on to the system when the globals are **KILL**ed in order to minimize inconvenience and avoid data corruption.

The site manager *must* monitor the proliferation of routines with names like **^DISZ*nnnn*** where "***nnnn***" is a four-digit number with leading zeros. These routines are created when compiled sorts are run. Ordinarily, they are deleted after the sort completes, but, if the system goes down or the job fails with an error, they can remain. When users are *not* on the system, the routine ENRLS^DIOZ can be run to clean up these routines and to release the "***nnnn***" numbers for reuse.

REF: For more information on the ENRLS^DIOZ utility, see the "COMPILED ROUTINE File Cleanup: ENRLS^DIOZ( )" section in the "System Management" section in the "Tools" section in the *VA FileMan Advanced User Manual*.

3 Files

This section lists all the VA FileMan files, file numbers, global locations, and a brief description of each. Data exported with VA FileMan 22.2 is described for some files:

- VA FileMan uses files numbered between **0** and **2**.
- VA FileMan files should *not* be altered, per VHA Directive 6402.

Table 4: VA FileMan File List (listed by file number)

File #	File Name	Global Location	Description
.11	INDEX	^DD("IX",	The INDEX file stores information about New-Style cross-references defined on a file. Whereas Traditional cross-references are stored under the **1** nodes of the ^DD for a particular field, New-Style cross-references are stored in this file and can consist of one field (simple cross-references), as well as more than one field (compound cross-references).
.2	DESTINATION	^DIC(.2	The DESTINATION file documents the location where data is used.
.31	KEY	^DD("KEY",	The KEY file stores information about keys on a file or subfile. A key is a set of one or more fields that uniquely identifies a record in a file. If more than one set of fields can uniquely identify a record, one of those sets should be designated the primary key; all others should be designated secondary keys. The primary key is the principal means of identifying records in the file. To allow VA FileMan to enforce key uniqueness, the database designer *must* define a regular index that consists of all the fields that make up the key. This index is called the uniqueness index. All key fields *must* have values. They cannot be **NULL**.
.4	PRINT TEMPLATE	^DIPT(	The PRINT TEMPLATE file stores VA FileMan PRINT templates. Exported PRINT templates include: • CAPTIONED

File #	File Name	Global Location	Description
			• FILE SECURITY CODES • DI-PKG-DEFAULT-DEFINITION • DDXP FORMAT DOC • DDXP FORMAT DOC HDR
.401	SORT TEMPLATE	^DIBT(	The SORT TEMPLATE file stores VA FileMan SORT, SEARCH, and INQUIRE templates.
.402	INPUT TEMPLATE	^DIE(	The INPUT TEMPLATE file stores VA FileMan INPUT templates.
.403	FORM	^DIST(.403	The FORM file stores forms used by VA FileMan to display screens. The DDXP FF FORM1 and various forms used by ScreenMan's Form Editor utility are exported.
.404	BLOCK	^DIST(.404	The BLOCK file stores blocks used to build forms for screen display. Blocks are exported for use with the forms sent with VA FileMan.
.44	FOREIGN FORMAT	^DIST(.44	The FOREIGN FORMAT file holds specifications for sending data to an application outside of M. Several Foreign Formats are exported.
.46	IMPORT TEMPLATE	^DIST(.46,	The IMPORT TEMPLATE file holds specifications for importing information from an application outside of M into a VA FileMan file.
.5	FUNCTION	^DD("FUNC"	The FUNCTION file stores the computed functions available in VA FileMan. The functions described in the *VA FileMan Advanced User Manual* are exported. **REF:** For more information on functions, see the "VA FileMan Functions" section in the "Tools" section in the VA *FileMan Advanced User Manual*.
.6	DD AUDIT	^DDA(	The DD AUDIT file stores the changes made to data dictionaries.
.7	MUMPS OPERATING SYSTEM	^DD("OS"	The MUMPS OPERATING SYSTEM file stores the operating systems

File #	File Name	Global Location	Description
			recognized by VA FileMan along with operating system-specific data. This data is exported.
.81	DATA TYPE	^DI(.81	The DATA TYPE file stores information about the DATA TYPEs known to VA FileMan. Several DATA TYPEs are exported.
.83	COMPILED ROUTINE	^DI(.83	The COMPILED ROUTINE file contains a list of numbers (to be used to create compiled sort routines) and a flag to indicate whether a number is currently in use.
.84	DIALOG	^DI(.84	The DIALOG file contains text used to “talk” to the user (error messages, help text, prompts). Entries under IEN **10,000** are exported by VA FileMan and are used in VA FileMan routines.
.85	LANGUAGE	^DI(.85	The LANGUAGE file is used to reference data dictionary elements and subentries in the DIALOG file for user dialogue in foreign languages and contains M code used to perform data transformations for such things as dates and numbers to non-English formats. All the languages in ISO 639-2:1998 (as revised 11/21/2012; International Organization for Standardization) are exported.
.86	DATA TYPE PROPERTY	^DI(.86	The DATA TYPE PROPERTY file stores the names of different kinds of STRINGS that describe data.
.87	DATA TYPE METHOD	^DI(.87	The DATA TYPE METHOD file stores the names of different kinds of lines of MUMPS code that are used in the definitions of DATA TYPES.
.9	META DATA DICTIONARY	^DDD(	The META DATA DICTIONARY file stores the file and field definitions of all files and fields in a VA FileMan instance.
1	FILE	^DIC(	The FILE file stores the name, number, global name or location, package name, security access, and

File #	File Name	Global Location	Description
			developer of VA FileMan created files. Data for the VA FileMan files is exported.
1.1	AUDIT	^DIA(	The AUDIT file stores the date and time, user's name, and old and new data values of changes made to audited fields.
1.11	ARCHIVAL ACTIVITY	^DIAR(1.11	The ARCHIVAL ACTIVITY file stores information about and status of archiving and extract activities.
1.12	FILEGRAM HISTORY	^DIAR(1.12	The FILEGRAM HISTORY file stores information and status of Filegrams.
1.13	FILEGRAM ERROR LOG	^DIAR(1.13	The FILEGRAM ERROR LOG file stores information about Filegram errors and the text of the affected Filegram.
1.2	ALTERNATE EDITOR	^DIST(1.2	The ALTERNATE EDITOR file stores information about the editors that can be used to edit VA FileMan's WORD-PROCESSING-type fields. Data for the Line Editor and the Screen Editor is exported.
1.5	ENTITY	^DDE(	The ENTITY file maps VistA data to entities or resources, which can be exposed RESTfully to standard web methods and formats. It can support various data models; for example: • Fast Healthcare Interoperability Resources (FHIR) • InterSystems' Summary Document Architecture (SDA)
1.521	SQLI_SCHEMA	^DMSQ("S",	The SQLI_SCHEMA file stores a set of tables and domains; a subset of catalog and environment.
1.52101	SQLI_KEY_WORD	^DMSQ("K",	The SQLI_KEY_WORD file stores the SQL identifiers that *cannot* be used for column and table names. SQL, ODBC, and vendors all have lists of restricted words, which should be put in this table before SQLI table generation.

File #	File Name	Global Location	Description
1.5211	SQLI_DATA_TYPE	^DMSQ(“DT”,	The SQLI_DATA_TYPE file stores a set of values from which all domains of that type can be drawn: • PRIMARY_KEY—Set of all primary keys (in SQLI_TABLE_ELEMENT [#1.5216] file, type P). • CHARACTER—Set of all character strings of length less than 256. • INTEGER—Set of all cardinal numbers. • NUMERIC—Set of all real numbers. • DATE—Set of all date valued tokens. • TIME—Set of all time valued tokens. • MOMENT—Set of all tokens that have both a date and a time value. • BOOLEAN—Set of all tokens that evaluate to true or false only. • MEMO—Set of all character strings of length greater than 255.
1.5212	SQLI_DOMAIN	^DMSQ(“DM”,	The SQLI_DOMAIN file stores the set from which all objects of that domain *must* be drawn. In SQLI, all table elements (SQLI_TABLE_ELEMENT [#1.5216] file) have a domain that restricts them to their domain set. For each DATA TYPE there is a domain of the same name, representing the same set. Other domains have different set membership restrictions. Each domain has a DATA TYPE, which determines the rules for comparing values from different domains, and the operators that can be used on them. The PRIMARY_KEY DATA TYPE and domain is unique to SQLI. It is

File #	File Name	Global Location	Description
			used to relate primary keys to foreign keys unambiguously. **REF:** For information on table elements, see the SQLI_TABLE_ELEMENT (#1.5216) file.
1.5213	SQLI_KEY_FORMAT	^DMSQ(“KF”,	The SQLI_KEY_FORMAT file stores strategies for converting base values into key values. Soundex and uppercase conversion are common examples. This implies that comparisons of key values with base values *must* be preceded by conversion of the base value to a key value. Key formats are frequently lossy; they *cannot* be converted uniquely back to base format.
1.5214	SQLI_OUTPUT_FORMAT	^DMSQ(“OF”,	The SQLI_OUTPUT_FORMAT file stores strategies for converting base values to external values. In VA FileMan, they are used to convert references to pointers to their text values. They are also used for the SET OF CODES type. SQLI projects POINTER TO A FILE and SET OF CODES as calls to $$GET1^DIQ, VARIABLE-POINTERs into calls to $$EXTERNAL^DILFD. Vendors and other users of SQLI can implement their own conversions to improve performance.
1.5215	SQLI_TABLE	^DMSQ(“T”,	The SQLI_TABLE file stores the descriptor of a set of table elements, which includes name and file number (see the SQLI_TABLE_ELEMENT [#1.5216] file). Each ^DD(DA) represents a table in a relational model of VA FileMan. Further, each index represents a table. Each schema contains multiple tables. Each table contains just one primary key, but multiple columns, foreign keys and indices.

File #	File Name	Global Location	Description
1.5216	SQLI_TABLE_ELEMENT	^DMSQ("E",	The SQLI_TABLE_ELEMENT file contains the names and domains of primary keys, columns, and foreign keys. Each represents the relational concept of an attribute; whose essential characteristics are a name (unique by relation) and a domain.  **REF:** For more information, see the SQLI_PRIMARY_KEY, SQLI_COLUMN, and SQLI_FOREIGN KEY files.
1.5217	SQLI_COLUMN	^DMSQ("C",	The SQLI_COLUMN file stores a set of formatting and physical structure specifications. Each column specification has a column type table element (SQLI_TABLE_ELEMENT file) that contains the relational specifications, name, and domain. The column specification contains those attributes required to locate the value in the global structure and to project the value to the user.  **REF:** For information on table elements, see the SQLI_TABLE_ELEMENT (#1.5216) file.
1.5218	SQLI_PRIMARY_KEY	^DMSQ("P",	The SQLI_PRIMARY_KEY file stores a chosen set of columns that uniquely identify a table. In the relational model (as in set theory) the columns of a primary key are *not* ordered. In SQLI, they *must* be, in order to map to the quasi-hierarchical model of M globals. VA FileMan subfiles (Multiples) have a primary key element for each parent plus one for the subfile. Each contains a pointer to its primary key table element (SQLI_TABLE-ELEMENT file), a sequence, and a column in the local base table (SQLI_COLUMN file).

File #	File Name	Global Location	Description
			REF: For information, see the SQLI_TABLE_ELEMENT and SQLI_COLUMN files above.
1.5219	SQLI_FOREIGN_KEY	^DMSQ("F",	The SQLI_FOREIGN_KEY file stores a set of columns in a table that match the primary key of another table. They represent an explicit join of the two tables. Each foreign key element points to its table element (SQLI_TABLE_ELEMENT [#1.5216] file), a column in the local table (SQLI_COLUMN file), and a primary key element of a foreign table (SQLI_PRIMARY_KEY file). The primary key table element of the foreign table has the domain of that table, which makes the connection. **REF:** For more information, see the SQLI_TABLE_ELEMENT, SQLI_COLUMN, and SQLI_PRIMARY_KEY files.
1.52191	SQLI_ERROR_TEXT	^DMSQ("ET",	The SQLI_ERROR_TEXT file stores a numbered list of error messages, auto-generated by ERR^DMSQU.
1.52192	SQLI_ERROR_LOG	^DMSQ("EX",	The SQLI_ERROR_LOG file stores a log of all errors encountered while compiling SQLI. It generates the error text table (SQLI_ERROR_TEXT file) on a LAYGO basis; errors are added only when they occur. If DBS errors triggered the error, the DIALOG file reference is also saved. **REF:** For more information, see the SQLI_ERROR_TEXT and DIALOG files.
1.6	POLICY	^DIAC(1.6,	The POLICY file is a self-referring, namespaced file, which is similar to the OPTION (#19) file. Rules are stored in a sub-file, much like menu items, and evaluated in sequence. If more complex policies are needed,

File #	File Name	Global Location	Description
			policy sets can be created by grouping other policies or sets, drilling down the levels in sequence like a menu tree.
1.61	APPLICATION ACTION	^DIAC(1.6,	The APPLICATION ACTION file stores the list of actions that can be taken on a file or sub-file (e.g., read, cancel, sign, etc.). Each action can be mapped to a policy that is evaluated when that kind of access to data is requested.
1.62	POLICY FUNCTION	^DIAC(1.6,	Supporting M code for policies is implemented as M functions and stored as entries in the POLICY FUNCTION file.
1.71	WORLD TIMEZONES	^DIT(1.71,	The WORLD TIMEZONES file stores time zone designations used throughout the world.
1.72	WORLD DAYLIGHT SAVINGS	^DIT(1.72,	The WORLD DAYLIGHT SAVINGS file tracks which countries have periods during the year in which they follow DAYLIGHT SAVING TIME, STANDARD TIME, or SUMMER TIME.
1.75	DATA SYNCHRONIZATION HISTORY	^DIFS(1.75	The DATA SYNCHRONIZATION HISTORY file is used to capture information from DATA SYNCHRONIZATION processing. Information logged allows an administrator to see if the process completed successfully or if there were issues and what errors were reported by the processing.

Installing the KIDS build for VA FileMan 22.2 loads the files listed in Table 4. Two files (LANGUAGE [#.85] and META DATA DICTIONARY [#.9]) are carried by the KIDS build in the standard fashion; the other files are installed when KIDS runs **DINIT**.

The PACKAGE (#9.4) file init routines (DIPKINIT) are no longer sent with VA FileMan 22.2. The PACKAGE (#9.4) file is necessary to build inits using **DIFROM**.

REF: For more information on DIFROM, see the "DIFROM" section in the "Developer's Tools" section in the VA *FileMan Developer's Guide*.

CAUTION: The Kernel Installation and Distribution System (KIDS) replaced the use of DIFROM as the method of exporting software packages in the VA. The version of DIFROM released with VA FileMan 22.2 will transport the new Key and Index structures.

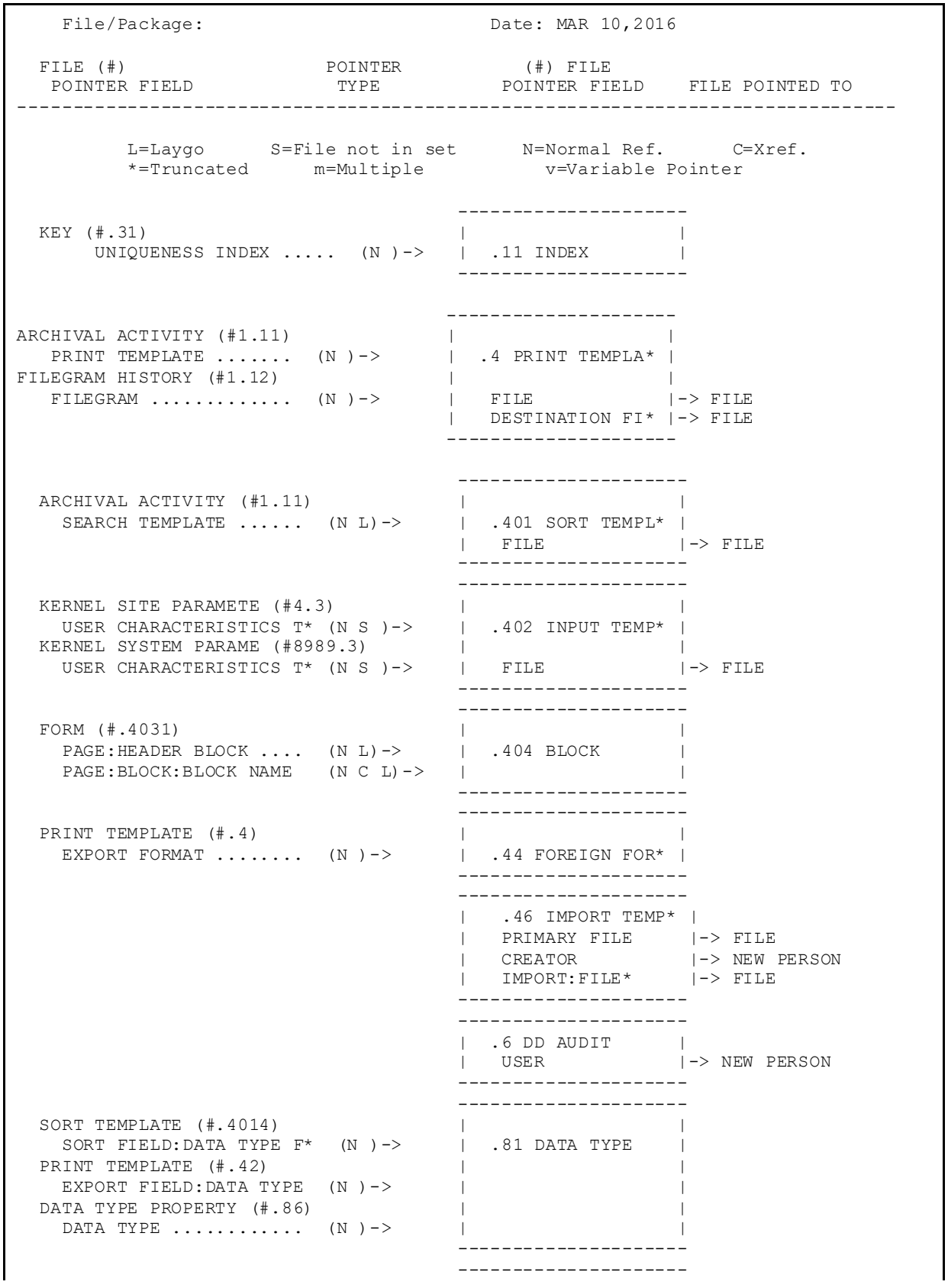
3.1 Pointer Map

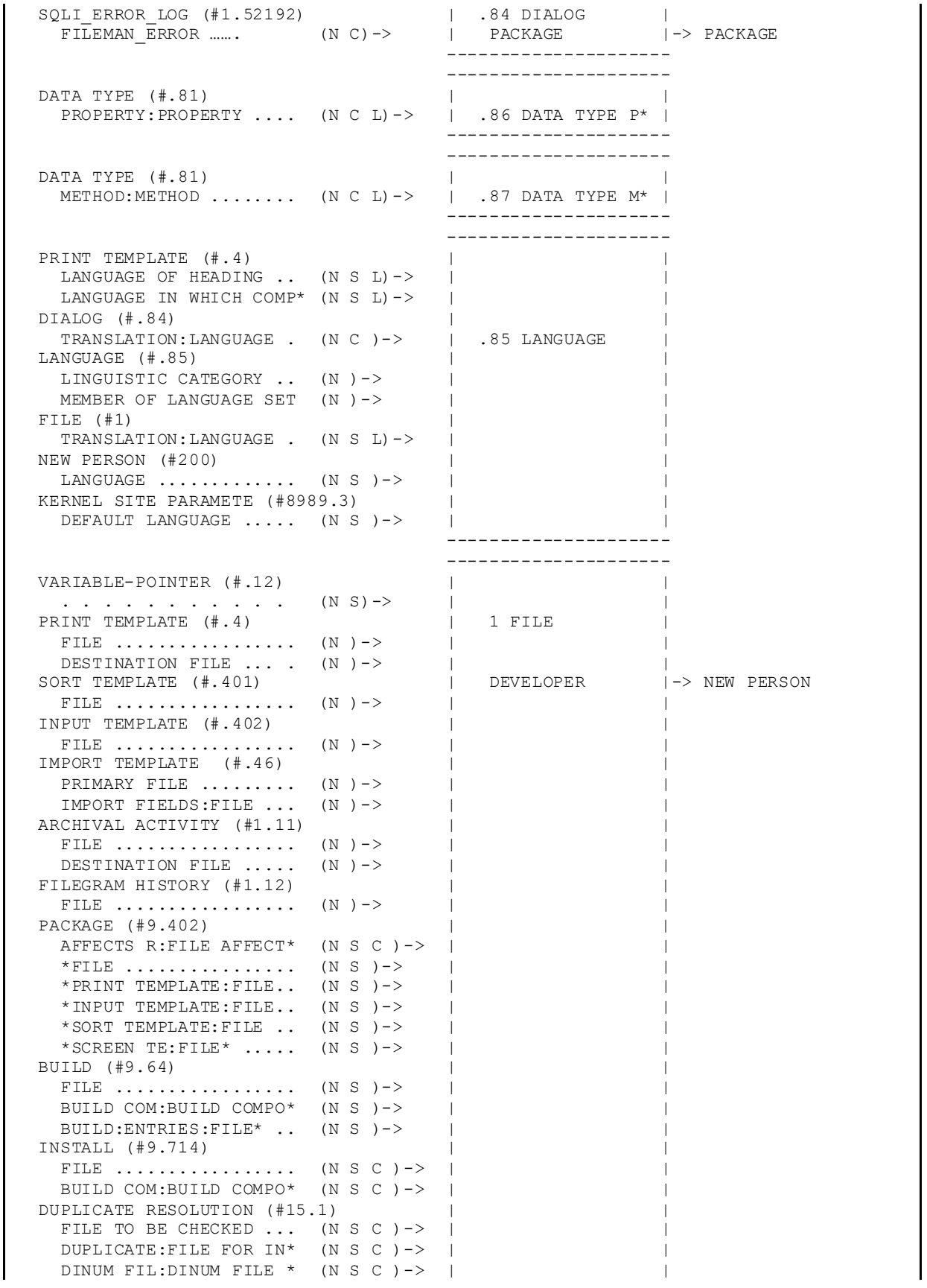
Figure 2 is a diagram of the pointer relationships between fields in VA FileMan's files. This pointer map reflects the relationships that exist in a VA FileMan environment running Kernel 8.0. As files are added to a system, new pointer relationships can be created; thus, the actual map for different operational systems can vary.

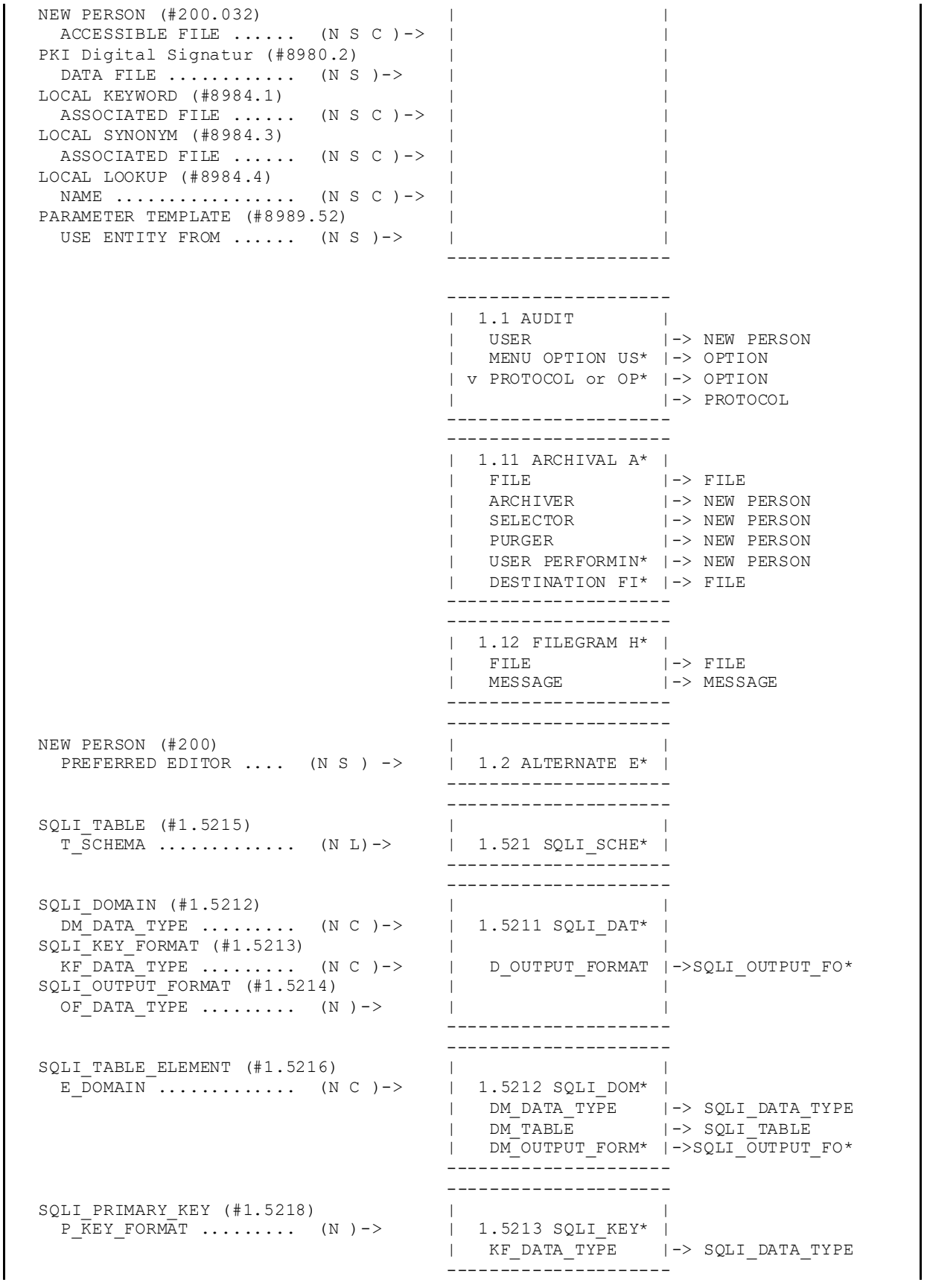
The diagram in Figure 2 was created using the Map Pointer Relations option on the Data Dictionary Utilities submenu.

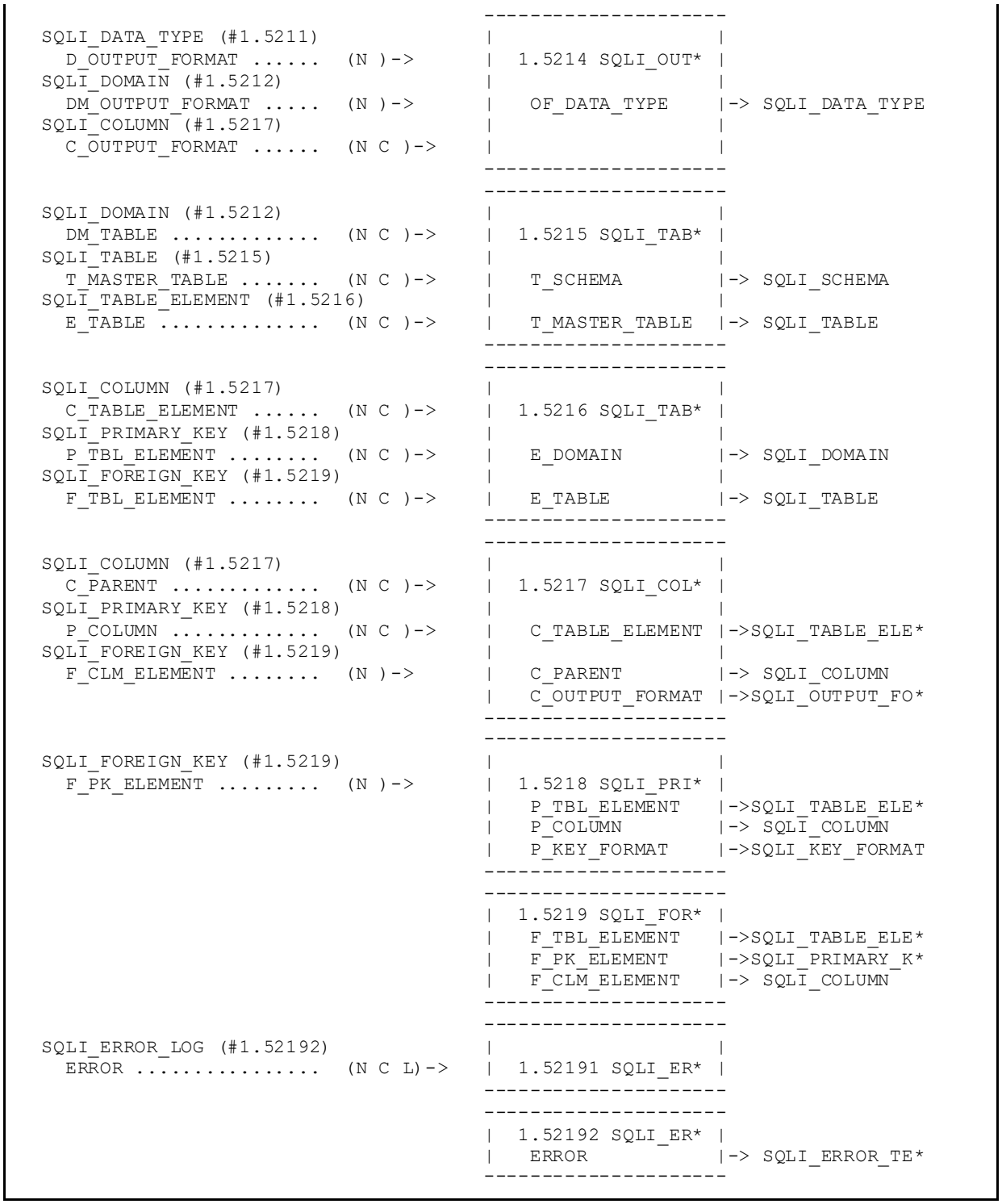
REF: For more information about creating and reading this map, see the "Map Pointer Relations option" section in the "List File Attributes" section in the "File Management" section in the *VA FileMan Advanced User Manual*.

Figure 2: VA FileMan Pointer Map




SQLI_ERROR_LOG (#1.52192)
FILEMAN_ERROR …….
(N C)->
.84 DIALOG
PACKAGE
-> PACKAGE
DATA TYPE (#.81)
PROPERTY:PROPERTY ....
(N C L)->
.86 DATA TYPE P*
DATA TYPE (#.81)
METHOD:METHOD ....
(N C L)->
.87 DATA TYPE M*
PRINT TEMPLATE (#.4)
LANGUAGE OF HEADING ..
(N S L)->
LANGUAGE IN WHICH COMP*
(N S L)->
DIALOG (#.84)
TRANSLATION:LANGUAGE .
(N C)->
.85 LANGUAGE
LANGUAGE (#.85)
LINGUISTIC CATEGORY ..
(N)->
MEMBER OF LANGUAGE SET
(N)->
FILE (#1)
TRANSLATION:LANGUAGE .
(N S L)->
NEW PERSON (#200)
LANGUAGE .....
(N S)->
KERNEL SITE PARAMETE (#8989.3)
DEFAULT LANGUAGE .....
(N S)->
VARIABLE-POINTER (#.12)
. . . . . .
(N S)->
PRINT TEMPLATE (#.4)
FILE .....
(N)->
DESTINATION FILE ... .
(N)->
1 FILE
SORT TEMPLATE (#.401)
FILE .....
(N)->
DEVELOPER
-> NEW PERSON
INPUT TEMPLATE (#.402)
FILE .....
(N)->
IMPORT TEMPLATE (#.46)
PRIMARY FILE .....
(N)->
IMPORT FIELDS:FILE ...
(N)->
ARCHIVAL ACTIVITY (#1.11)
FILE .....
(N)->
DESTINATION FILE .....
(N)->
FILEGRAM HISTORY (#1.12)
FILE .....
(N)->
PACKAGE (#9.402)
AFFECTS R:FILE AFFECT*
(N S C)->
*FILE .....
(N S)->
*PRINT TEMPLATE:FILE..
(N S)->
*INPUT TEMPLATE:FILE..
(N S)->
*SORT TEMPLATE:FILE ..
(N S)->
SCREEN TE:FILE .....
(N S)->
BUILD (#9.64)
FILE .....
(N S)->
BUILD COM:BUILD COMPO*
(N S)->
BUILD:ENTRIES:FILE* ..
(N S)->
INSTALL (#9.714)
FILE .....
(N S C)->
BUILD COM:BUILD COMPO*
(N S C)->
DUPLICATE RESOLUTION (#15.1)
FILE TO BE CHECKED ...
(N S C)->
DUPLICATE:FILE FOR IN*
(N S C)->
DINUM FIL:DINUM FILE *
(N S C)->
1 FILE
DEVELOPER
-> NEW PERSON


NEW PERSON (#200.032)
ACCESSIBLE FILE ..... (N S C)->
PKI Digital Signatur (#8980.2)
DATA FILE ..... (N S)->
LOCAL KEYWORD (#8984.1)
ASSOCIATED FILE ..... (N S C)->
LOCAL SYNONYM (#8984.3)
ASSOCIATED FILE ..... (N S C)->
LOCAL LOOKUP (#8984.4)
NAME ..... (N S C)->
PARAMETER TEMPLATE (#8989.52)
USE ENTITY FROM ..... (N S)->
1.1 AUDIT
USER
MENU OPTION US*
v PROTOCOL or OP*
-> NEW PERSON
-> OPTION
-> OPTION
-> PROTOCOL
1.11 ARCHIVAL A*
FILE
ARCHIVER
SELECTOR
PURGER
USER PERFORMIN*
DESTINATION FI*
-> FILE
-> NEW PERSON
-> NEW PERSON
-> NEW PERSON
-> NEW PERSON
-> FILE
1.12 FILEGRAM H*
FILE
MESSAGE
-> FILE
-> MESSAGE
NEW PERSON (#200)
PREFERRED EDITOR .... (N S) ->
1.2 ALTERNATE E*
SQLI_TABLE (#1.5215)
T_SCHEMA ..... (N L)->
1.521 SQLI_SCHE*
SQLI_DOMAIN (#1.5212)
DM_DATA_TYPE ..... (N C)->
SQLI_KEY_FORMAT (#1.5213)
KF_DATA_TYPE ..... (N C)->
SQLI_OUTPUT_FORMAT (#1.5214)
OF_DATA_TYPE ..... (N)->
1.5211 SQLI_DAT*
D_OUTPUT_FORMAT
->SQLI_OUTPUT_FO*
SQLI_TABLE_ELEMENT (#1.5216)
E_DOMAIN ..... (N C)->
1.5212 SQLI_DOM*
DM_DATA_TYPE
DM_TABLE
DM_OUTPUT_FORM*
-> SQLI_DATA_TYPE
-> SQLI_TABLE
->SQLI_OUTPUT_FO*
SQLI_PRIMARY_KEY (#1.5218)
P_KEY_FORMAT ..... (N)->
1.5213 SQLI_KEY*
KF_DATA_TYPE
-> SQLI_DATA_TYPE


SQLI_DATA_TYPE (#1.5211)
D_OUTPUT_FORMAT ..... (N)->
SQLI_DOMAIN (#1.5212)
DM_OUTPUT_FORMAT ..... (N)->
SQLI_COLUMN (#1.5217)
C_OUTPUT_FORMAT ..... (N C)->
1.5214 SQLI_OUT*
OF_DATA_TYPE
-> SQLI_DATA_TYPE
SQLI_DOMAIN (#1.5212)
DM_TABLE ..... (N C)->
SQLI_TABLE (#1.5215)
T_MASTER_TABLE ..... (N C)->
SQLI_TABLE_ELEMENT (#1.5216)
E_TABLE ..... (N C)->
1.5215 SQLI_TAB*
T_SCHEMA
-> SQLI_SCHEMA
T_MASTER_TABLE
-> SQLI_TABLE
SQLI_COLUMN (#1.5217)
C_TABLE_ELEMENT ..... (N C)->
SQLI_PRIMARY_KEY (#1.5218)
P_TBL_ELEMENT ..... (N C)->
SQLI_FOREIGN_KEY (#1.5219)
F_TBL_ELEMENT ..... (N C)->
1.5216 SQLI_TAB*
E_DOMAIN
-> SQLI_DOMAIN
E_TABLE
-> SQLI_TABLE
SQLI_COLUMN (#1.5217)
C_PARENT ..... (N C)->
SQLI_PRIMARY_KEY (#1.5218)
P_COLUMN ..... (N C)->
SQLI_FOREIGN_KEY (#1.5219)
F_CLM_ELEMENT ..... (N)->
1.5217 SQLI_COL*
C_TABLE_ELEMENT
->SQLI_TABLE_ELE*
C_PARENT
-> SQLI_COLUMN
C_OUTPUT_FORMAT
->SQLI_OUTPUT_FO*
SQLI_FOREIGN_KEY (#1.5219)
F_PK_ELEMENT ..... (N)->
1.5218 SQLI_PRI*
P_TBL_ELEMENT
->SQLI_TABLE_ELE*
P_COLUMN
-> SQLI_COLUMN
P_KEY_FORMAT
->SQLI_KEY_FORMAT
1.5219 SQLI_FOR*
F_TBL_ELEMENT
->SQLI_TABLE_ELE*
F_PK_ELEMENT
->SQLI_PRIMARY_K*
F_CLM_ELEMENT
-> SQLI_COLUMN
SQLI_ERROR_LOG (#1.52192)
ERROR ..... (N C L)->
1.52191 SQLI_ER*
1.52192 SQLI_ER*
ERROR
-> SQLI_ERROR_TE*

4 Routines, Application Programming Interfaces (APIs), and Options

4.1 Routines and Callable Entry Points

Table 5 lists and briefly describes the VA FileMan routines and Application Programming Interfaces (APIs; aka callable routines and entry points).

CAUTION VA FileMan routines should *not* be altered, per Veterans Health Administration (VHA) Directive 6402.

The Application Programming Interfaces (APIs; aka callable routines and entry points) for those VA FileMan routines that can be invoked from other applications are shown in the "Callable Entry Point" column in Table 5.

REF: The APIs, ScreenMan, and Database Server (DBS) calls are described in detail (including their function, required variables, and any restrictions) in the *VA FileMan Developer's Guide*:

APIs—See the "Major APIs" and "Other APIs" sections in the *VA FileMan Developer's Guide*.

ScreenMan—See the "ScreenMan" section in the *VA FileMan Developer's Guide*.

Database Server (DBS) calls—See the "Database Server (DBS)" section in the "Major APIs" section in the *VA FileMan Developer's Guide*.

REF: The Direct mode utilities, which can only be called directly from M and ScreenMan-specific utilities, are listed in Sections 4.2 and 4.3, and are also described in the *VA FileMan Developer's Guide*.

REF: Routine mapping is described in Section 4.4.

Table 5: VA FileMan Routines and Callable Entry Points

Routine	Callable Entry Point	Description
%DT		See DIDT for callable entry points and description.
%DTC		See DIDTC for callable entry points and description.
%RCR		See DIRCR for callable entry points and description.
DDBR	EN^DDBR WP^DDBR BROWSE^DDBR	Routines responsible for displaying ASCII text on a terminal screen, for viewing only.

Routine	Callable Entry Point	Description
	DOCLIST^DDBR	
DDBR0 **DDBR1** **DDBR2** **DDBR3** **DDBR4** **DDBRAHT** **DDBRAHTE** **DDBRAHTJ** **DDBRAHTR** **DDBRAP** **DDBRGE** **DDBRP** **DDBRS**		
DDBRT	$$TEST^DDBRT	
DDBRU **DDBRU2** **DDBRWB**		
DDBRZIS	CLOSE^DDBRZIS OPEN^DDBRZIS POST^DDBRZIS	
DDD	^DDD FILELIST^DDD PARTIAL1^DDD PARTIAL2^DDD	Routine that creates a full META DATA DICTIONARY (#.9) file. Other entry points to be used to update partial portions of the META DATA DICTIONARY (#.9) file.
DDE*	$$GET1^DDE GET^DDE	Entity main driver. Routines used to enter/edit entries in the ENTITY (#1.5) file, VA FileMan entity mapping menu options, and other DD utilities.
DDE1A		Entity Enter/Edit via VA FileMan.
DDEG		Entity **GET** Extract.
DDEGET		Entity **GET** Handler.
DDEMAP		Auto-Generate Data Mapping.
DDEOPT		DDE Options.
DDEPRT		Entity Print Utilities.
DDERR		Entity Error Handler.
DDEX		Entity Data Dictionary Utilities.

Routine	Callable Entry Point	Description
DDFIX		Routine that checks nodes in the data dictionary and the FILE (#1) file.
DDGF **DDGF0** **DDGF1** **DDGF2** **DDGF3** **DDGF4** **DDGFADL** **DDGFAPC** **DDGFASUB** **DDGFBK** **DDGFBSEL** **DDGFEL** **DDGFFLD** **DDGFFLDA** **DDGFFM** **DDGFH** **DDGFHBK** **DDGFLOAD** **DDGFORD** **DDGFPG** **DDGFSV** **DDGFU** **DDGFUPDB** **DDGFUPDP**		Routines used to create and edit ScreenMan forms.
DDGLBXA **DDGLBXA1** **DDGLCBOX** **DDGLIB0** **DDGLIBH** **DDGLIBW** **DDGLIBW1**		Routines that manage the screen for VA FileMan's screen-oriented utilities.
DDIOL	EN^DDIOL	Routine that any of the following: • Writes text to the screen. • Writes text in ScreenMan's Command Area. • Loads text into an array, depending on the environment in which it is called.

Routine	Callable Entry Point	Description
DDMAP **DDMAP1** **DDMAP2**		Routines that generate a graphic display of the pointer relationships among a specified group of package files to an output device.
DDMOD	DELIX^DDMOD DELIXN^DDMOD CREIXN^DDMOD FILESEC^DDMOD	Routine supporting calls for modifying DD attributes.
DDMP **DDMP1** **DDMP2** **DDMPSM** **DDMPSM1** **DDMPU**	FILE^DDMP	Routines used by the Import Tool.
DDPA2		Routine finds any sort templates that have a sort field with a range that is **FROM** or **TO** a *non*-canonic number.
DDR **DDR0** **DDR1** **DDR2** **DDR3** **DDR4**		Routines that contain the RPCs for the VA FileMan Delphi components.
DDS **DDS0** **DDS01** **DDS02** **DDS1** **DDS10** **DDS11** **DDS2** **DDS3** **DDS4** **DDS41** **DDS5** **DDS6** **DDS7** **DDSBOX** **DDSCAP**	DDS	Routines used to compile and run forms for data viewing and editing—ScreenMan.

Routine	Callable Entry Point	Description
DDSCLONE **DDSCLONF** **DDSCOM** **DDSCOMP** **DDSDBLK** **DDSDEL** **DDSDFRM** **DDSFO** **DDSIT** **DDSLIB** **DDSM** **DDSM1** **DDSMSG** **DDSOPT** **DDSPRNT** **DDSPRNT1** **DDSPRNT2** **DDSPTR** **DDSR** **DDSR1** **DDSRP** **DDSRSEL** **DDSRUN** **DDSSTK** **DDSU**		
DDSUTL	MSG^DDSUTL REFRESH^DDSUTL REQ^DDSUTL UNED^DDSUTL	
DDSVAL	$$GET^DDSVAL PUT^DDSVAL	
DDSVALF	$$GET^DDSVALF PUT^DDSVALF	
DDSVALM **DDSWP** **DDSZ** **DDSZ1** **DDSZ2** **DDSZ3**		

Routine	Callable Entry Point	Description
DDU **DDUCHK** **DDUCHK1** **DDUCHK2** **DDUCHK3** **DDUCHK4** **DDUCHK5**		Routines responsible for running the data dictionary checking utility.
DDW **DDW1** **DDW2** **DDW3** **DDW4** **DDW5** **DDW6** **DDW7** **DDW8** **DDW9** **DDWC** **DDWC1** **DDWF** **DDWG** **DDWH** **DDWK** **DDWT1**		Routines responsible for full screen text editing.
DDXP **DDXP1** **DDXP2** **DDXP3** **DDXP31** **DDXP32** **DDXP33** **DDXP4** **DDXP41** **DDXP5** **DDXPLIB**		Routines responsible for the data export to a Foreign Format tool.
DI		Routine for direct entry into VA FileMan.
DI222ENV **DI222POS**		These routines are removed after the install.

Routine	Callable Entry Point	Description
DI222PRE		
DIA **DIA1** **DIA2** **DIA3**		Routines responsible for gathering fields to be edited.
DIAC	DIAC	Routine that determines file access.
DIAC1	$$CANDO^DIAC1	Data Access Control (DAC): Policy Evaluation API.
DIAC1T		Data Access Control (DAC): Test utility for Policies.
DIACLM		Data Access Control (DAC): Policy Editor driver.
DIACLM1		Data Access Control (DAC): Policy Editor actions.
DIACOPT		Data Access Control (DAC): Data Access Control Options.
DIACP		Data Access Control (DAC): Print Policy Reports.
DIACX		Data Access Control (DAC): Policy utilities.
DIALOG	BLD^DIALOG $$EZBLD^DIALOG	Routines to build VA FileMan dialogues and their functions.
DIALOGU		
DIALOGZ	LANG^DIALOGZ	Routine that creates and uses foreign-language additions to the data dictionary.
DIAR **DIARA** **DIARB** **DIARCALC** **DIARR** **DIARR1** **DIARR2** **DIARR3** **DIARR4** **DIARR5** **DIARR6** **DIARU** **DIARX**		Routines responsible for VA FileMan archiving.
DIAU **DIAUTL**		Routines used for auditing.
DIAX **DIAXD**	EN^DIAXU	Routines responsible for extracting data to a VA FileMan file.

Routine	Callable Entry Point	Description
DIAXERR **DIAXF** **DIAXM** **DIAXM1** **DIAXM2** **DIAXM3** **DIAXMS** **DIAXP** **DIAXT** **DIAXU**		
DIB	EN^DIB	Routine that creates a new file.
DIBT **DIBT1** **DIBTEDT**		Routine that stores a SORT template.
DIC	DIC FIND^DIC $$FIND1^DIC IX^DIC LIST^DIC	Routines that perform VA FileMan lookups or return an ordered list of records.
DIC0		
DIC1	MIX^DIC1 DO^DIC1	
DIC11 **DIC2** **DIC3** **DIC4** **DIC5**		
DICA **DICA1** **DICA2** **DICA3**		Routines responsible for DBS Updater functions.
DICATT **DICATT0** **DICATT1** **DICATT2** **DICATT22** **DICATT3**		Routines responsible for the Modify File Attributes option.

Routine	Callable Entry Point	Description
DICATT4 **DICATT5** **DICATT6**		
DICATTA		Routine responsible for data dictionary audits.
DICATTD **DICATTD0** **DICATTD1** **DICATTD2** **DICATTD3** **DICATTD4** **DICATTD5** **DICATTD6** **DICATTD7** **DICATTD8** **DICATTD9** **DICATTDD** **DICATTDE** **DICATTDK** **DICATTDM** **DICATTUD**		Routines responsible for Modify File Attributes option in Screen oriented format.
DICD	WAIT^DICD	Routine for selecting, displaying, editing, or deleting a cross-reference.
DICE **DICE0** **DICE1** **DICE2** **DICE3** **DICE4** **DICE7**		Routines responsible for creating cross-references.
DICF **DICF0** **DICF1** **DICF2** **DICF3** **DICF4** **DICF5** **DICFIX** **DICFIX1**		Routines responsible for DBS Finder functions.

Routine	Callable Entry Point	Description
DICL **DICL1** **DICL10** **DICL2** **DICL3** **DICLGFT** **DICLIB** **DICLIX** **DICLIX0** **DICLIX1**		Routines responsible for DBS Lister functions.
DICM **DICM0** **DICM1** **DICM2** **DICM3**		Routines responsible for performing transforms on the lookup value to attempt to find a match on the lookup indexes. For example, transforms date to internal format.
DICN	FILE^DICN YN^DICN	Routines that allow adding a new entry to a file.
DICN0 **DICN1**		
DICOMP **DICOMP0** **DICOMP1** **DICOMPU** **DICOMPV** **DICOMPW** **DICOMPX** **DICOMPY** **DICOMPZ**		Routines that evaluate computed field expressions.
DICQ **DICQ1**	DQ^DICQ	Routines responsible for help on lookups.
DICR		Routine responsible for recursive calls for cross-references on triggered fields.
DICRW **DICRW1**	DT^DICRW	Routines that select a file.
DICU **DICU1** **DICU11**		Routines containing utilities used during lookups.

Routine	Callable Entry Point	Description
DICU2 **DICUF** **DICUIX** **DICUIX1** **DICUIX2**		
DID	EN^DID FIELD^DID FIELDLST^DID FILE^DID FILELST^DID $$GET1^DID	Routines for data dictionary listings.
DID1		Standard data dictionary listing.
DID2		Modified data dictionary listing.
DIDC		Condensed data dictionary listing.
DIDG		Global Map data dictionary listing.
DIDGFTPT		Find pointers into a file utility.
DIDH		Headers for the data dictionary listings.
DIDH1		
DIDT	%DT DD^%DT	Routine responsible for the Date/Time validation. *Must* be stored in the Manager account as **%DT**.
DIDTC	%DTC C^%DTC NOW^%DTC H^%DTC DW^%DTC YMD^%DTC COMMA^%DTC S^%DTC YX^%DTC HELP^%DTC	Routine responsible for the Date/Time operations. *Must* be stored in the Manager account as **%DTC**.
DIDU **DIDU1** **DIDU2**		Routines responsible for data dictionary functions.
DIDX		Brief data dictionary listing.
DIE	DIE CHK^DIE	Routines responsible for the Enter or Edit File Entries option and for DBS filing and help retrieval functions.

Routine	Callable Entry Point	Description
	FILE^DIE HELP^DIE $$KEYVAL^DIE UPDATE^DIE VAL^DIE VALS^DIE WP^DIE	
DIE0 **DIE1** **DIE17** **DIE2** **DIE3** **DIE9** **DIED** **DIEF** **DIEF1** **DIEFU** **DIEFW** **DIEH** **DIEH1** **DIEKMSG** **DIEQ** **DIEQ1**		
DIENV **DIENVSTP** **DIENVWRN**		Environment check routines.
DIET **DIETED**		Routine that displays an INPUT template and performs VA FileMan auditing function.
DIETLIB		Library of APIs for user-defined data types.
DIETLIBF		Library for field attributes.
DIEV **DIEV1** **DIEVK** **DIEVK1** **DIEVS**		Routines responsible for data validation functions.
DIEZ **DIEZ0** **DIEZ1**	DIEZ EN^DIEZ	Routines that compile INPUT templates.

Routine	Callable Entry Point	Description
DIEZ2 **DIEZ3** **DIEZ4**		
DIFG **DIFG0** **DIFG0A** **DIFG0B** **DIFG1** **DIFG2** **DIFG3** **DIFG3A** **DIFG4** **DIFG4A** **DIFG5** **DIFG6** **DIFG7** **DIFGA** **DIFGA1** **DIFGB**	DIFG	Routines responsible for Filegrams.
DIFGG	EN^DIFGG	
DIFGG2		
DIFGG4 **DIFGGI** **DIFGGSB** **DIFGGSB1** **DIFGGSB2** **DIFGGU** **DIFGO** **DIFGSRV**		
DIFMEDT1	ENP81^DIFMEDT1 ENP86^DIFMEDT1 ENP87^DIFMEDT1	Routine to enter/edit entries in the following files: • DATA TYPE (#.81) • DATA TYPE PROPERTY (#.86) • DATA TYPE METHOD (#.87)
DIFROM **DIFROM0** **DIFROM1** **DIFROM11** **DIFROM12**	DIFROM	Routines responsible for generating init packages for export and supporting Kernel's KIDS functions.

Routine	Callable Entry Point	Description
DIFROM2 **DIFROM3** **DIFROM4** **DIFROM41** **DIFROM42** **DIFROM5** **DIFROM6** **DIFROM7** **DIFROMH** **DIFROMH1** **DIFROMS** **DIFROMS1** **DIFROMS2** **DIFROMS3** **DIFROMS4** **DIFROMS5** **DIFROMSB** **DIFROMSC** **DIFROMSD** **DIFROMSE** **DIFROMSI** **DIFROMSK** **DIFROMSL** **DIFROMSO** **DIFROMSP** **DIFROMSR** **DIFROMSS** **DIFROMSU** **DIFROMSV** **DIFROMSX** **DIFROMSY**		
DIFSBLD	JSON1^DIFSBLD	Routine to accept a JSON formatted file, process the data in it, and insert the data into a FileMan file, logging the results into a new Data Synchronization History file (#1.75).
DIG		Routine responsible for the Scattergram option on the Statistics submenu.
DIH		Routine responsible for the Histogram option on the Statistics submenu.

Routine	Callable Entry Point	Description
DII **DII1**		Routines responsible for the main menu in standalone VA FileMan and for the Inquire to File Entries option.
DIIS **DIISC** **DIISS**		Routines responsible for device selection for standalone VA FileMan. Stored in the Manager account as **%ZIS**, **%ZISC**, and **%ZISS**.
DIK	DIK IXALL^DIK IX^DIK IX1^DIK ENALL^DIK EN^DIK EN1^DIK	Routines that perform file re-indexing and entry deletion.
DIK1		
DIKC **DIKC1** **DIKC2** **DIKCBLD** **DIKCDD** **DIKCFORM** **DIKCP** **DIKCP1** **DIKCP2** **DIKCP3** **DIKCR** **DIKCU** **DIKCU1** **DIKCU2** **DIKCUTL** **DIKCUTL1** **DIKCUTL2** **DIKCUTL3** **DIKD** **DIKD1** **DIKD2**	DIKCBLD	Routines responsible for defining, deleting, printing, and executing the logic for New-Style indices.
DIKK **DIKK1** **DIKK2** **DIKKDD**		Routines responsible for defining, printing, and verifying the integrity of Keys.

Routine	Callable Entry Point	Description
DIKKFORM **DIKKP** **DIKKUTL** **DIKKUTL1** **DIKKUTL2** **DIKKUTL3** **DIKKUTL4**		
DIKZ **DIKZ0** **DIKZ1** **DIKZ11** **DIKZ2**	DIKZ EN^DIKZ	Routines responsible for VA FileMan's cross-reference compiler.
DIL **DIL0** **DIL1** **DIL11** **DIL2** **DILL**		Routines responsible for processing PRINT templates or fields.
DILF	CLEAN^DILF $$CREF^DILF DA^DILF DT^DILF FDA^DILF $$IENS^DILF $$OREF^DILF $$VALUE1^DILF VALUES^DILF	Routine that contains VA FileMan's library of functions.
DILFD	$$EXTERNAL^DILFD $$FLDNUM^DILFD PRD^DILFD RECALL^DILFD $$ROOT^DILFD $$VFIELD^DILFD $$VFILE^DILFD	
DILIBF		
DIM **DIM1** **DIM2**	DIM	Routines responsible for the M syntax checker.

Routine	Callable Entry Point	Description
DIM3 **DIM4**		
DINIT		Routines that initialize VA FileMan.
DINIT*		Numerous routines starting with “DINIT” are used in the initialization process.
DINVGTM **DINVGUX** **DINVONT** **DINZONT**		Routines containing operating system specific code.
DIO **DIO0** **DIO1**		Routines responsible for building sort logic, executing the sort, and performing output functions.
DIO2	DT^DIO2	
DIO3 **DIO4** **DIOS** **DIOS1**		
DIOC		Routine responsible for checking code to check query conditions.
DIOQ		Routine responsible for determining sort (query) optimization numbers.
DIOU		Routines responsible for generic VA FileMan code generation utilities.
DIOZ	^DIOZ	Routines responsible for compiling SORT templates.
DIP **DIP0** **DIP1** **DIP10** **DIP100** **DIP11** **DIP12** **DIP2** **DIP21** **DIP22** **DIP23** **DIP3** **DIP31**	EN1^DIP	Routines that: process sorting specifications, edit SORT templates, process the **FROM** and **TO** sort range, edit PRINT templates, process PRINT templates, and initialize the printing process.

Routine	Callable Entry Point	Description
DIP4 **DIP5**		
DIPT	DIPT DIBT^DIPT	Routine that displays PRINT and SORT templates.
DIPTED		Routine used for the ScreenMan-based PRINT template editor.
DIPZ	DIPZ EN^DIPZ	Routines that compile PRINT templates.
DIPZ0 **DIPZ1** **DIPZ2**		
DIQ	EN^DIQ Y^DIQ D^DIQ DT^DIQ $$GET1^DIQ GETS^DIQ	Routines that retrieve data and support DBS Retriever and DD Retriever functions.
DIQ1	EN^DIQ1	
DIQG **DIQGDD** **DIQGDD0** **DIQGDDF** **DIQGDDT** **DIQGDDU** **DIQGQ** **DIQGU** **DIQGU0**		
DIQQ **DIQQ1** **DIQQQ**		Routines that provide help on various subjects.
DIR **DIR0** **DIR01** **DIR02** **DIR03** **DIR0H** **DIR0K**	DIR	Routines responsible for the standard reader used in VA FileMan.

Routine	Callable Entry Point	Description
DIR0W **DIR1** **DIR2** **DIR3** **DIRQ**		
DIRCR	XY^%RCR	Routine that moves arrays. *Must* be stored in the Manager account as **%RCR**.
DIS	EN^DIS	Routines responsible for the Search File Entries option.
DIS0 **DIS1** **DIS2** **DIS3**		
DISZ*		Temporary routines compiled for SORT templates and deleted after use (*not* exported with VA FileMan routines).
DIT **DIT0** **DIT1** **DIT2** **DIT3** **DITP** **DITR** **DITR1**		Routines responsible for the Transfer Entries option. Also used by the Compare/Merge option and by **DIFROM**.
DITC **DITC0** **DITC1** **DITC2** **DITC3**		Routines responsible for allowing a user to select data values during the compare/merge process.
DITCP **DITCP0** **DITCPL**		Routines enabling comparison of data and data dictionaries across environments.
DITIME		Input Transform for “TIME” Data Type.
DITM **DITM1** **DITM2** **DITMGM1** **DITMGM2**		Routines used to compare/merge two records located within a single file.

Routine	Callable Entry Point	Description
DITMGM2A **DITMGM2B** **DITMGM2C** **DITMGMRG** **DITMGMRI** **DITMU1** **DITMU2** **DITMU3** **DITMU4**		
DITP		Routine responsible for transferring pointers.
DIU **DIU0** **DIU1**		Routines responsible for the Utility Functions option.
DIU2	EN^DIU2	
DIU20 **DIU21** **DIU3** **DIU31** **DIU4** **DIU5**		
DIUCANON		Routine containing utilities for Canonic Templates.
DIUTC	$$UTC^DIUTC	Routine to convert a VA FileMan date/time into Coordinated Universal Time (UTC).
DIUTL		General utility routines used internally by VA FileMan.
DIV **DIVC** **DIVR** **DIVR1** **DIVU**		Routines that verify field data.
DIVRE **DIVRE1**		Routine that checks for required field data.
DIVRPTR	DIVRPTR	Routine called from programmer mode to check pointers.
DIWE	EN^DIWE	Routines responsible for VA FileMan's Line Editor and display of word processing output. They also provide for use of Alternate Editors.
DIWE1		

Routine	Callable Entry Point	Description
DIWE11 **DIWE12** **DIWE2** **DIWE3** **DIWE4** **DIWE5**		
DIWF	DIWF EN1^DIWF EN2^DIWF	Routine used for printing forms.
DIWP **DIWW**	DIWP DIWW	Routines responsible for display of word processing output.
DIX **DIXC**		Routines used for the Statistics option. Routine used for the Descriptive Statistics option.
DMSQ **DMSQD** **DMSQE** **DMSQF** **DMSQF1** **DMSQF2** **DMSQP** **DMSQP1** **DMSQP2** **DMSQP3** **DMSQP4** **DMSQP5** **DMSQP6** **DMSQS** **DMSQT** **DMSQT1** **DMSQU**		Routines used to build and maintain an SQL mapping to VA FileMan data. Allows access to VA FileMan data using an SQL interface.

REF: For details on all VA FileMan callable routines/entry points/APIs, see the *VA FileMan Developer's Guide*.

4.2 Direct Mode Utilities

In addition to the callable entry points shown in Table 5, there are a few other entry points into VA FileMan routines. Unlike the callable entry points, these entries ***cannot be used within application programs***. Only users with programmer access can invoke the following direct mode utilities from the M prompt:

- **C^DI**
- **D^DI**
- **P^DI**
- **Q^DI**

REF: For more information on these direct mode utilities, see the "^DI: Programmer Access" section in the "Developer Tools" section in the *VA FileMan Developer's Guide*.

4.3 ScreenMan-Specific Utilities

The following are ScreenMan-specific utilities:

- ^DDGF
- CLONE^DDS
- PRINT^DDS
- RESET^DDS

REF: For more information on these ScreenMan-specific utilities, see the "Programmer Mode Utilities" section in the "ScreenMan Forms" section in the "ScreenMan" section in the *VA FileMan Developer's Guide.*

4.4 Mapping Routines

No VA FileMan-specific routine mapping actions are needed in the VA environment.

4.5 Direct Mode VA FileMan

The exported menu structure of VA FileMan is displayed in Figure 3.

The following options are accessible from the MUMPS command prompt using the calls described in Section 4.2, "Direct Mode Utilities

- ENTER OR EDIT FILE ENTRIES
- PRINT FILE ENTRIES
- SEARCH FILE ENTRIES
- MODIFY FILE ATTRIBUTES
- INQUIRE TO FILE ENTRIES

UTILITY FUNCTIONS:

- VERIFY FIELDS
- CROSS-REFERENCE A FIELD OR FILE
- IDENTIFIER
- RE-INDEX FILE
- INPUT TRANSFORM (SYNTAX)
- EDIT FILE
- OUTPUT TRANSFORM
- TEMPLATE EDIT
- UNEDITABLE DATA
- MANDATORY/REQUIRED FIELD CHECK
- KEY DEFINITION

OTHER OPTIONS:

- FILEGRAMS:
  - CREATE/EDIT FILEGRAM TEMPLATE
  - DISPLAY FILEGRAM TEMPLATE
  - GENERATE FILEGRAM
  - VIEW FILEGRAM

- SPECIFIERS
- INSTALL/VERIFY FILEGRAM

- ARCHIVING:
  - SELECT ENTRIES TO ARCHIVE
  - ADD/DELETE SELECTED ENTRIES
  - PRINT SELECTED ENTRIES
  - CREATE FILEGRAM ARCHIVING TEMPLATE
  - WRITE ENTRIES TO TEMPORARY STORAGE
  - MOVE ARCHIVED DATA TO PERMANENT STORAGE
  - PURGE STORED ENTRIES
  - CANCEL ARCHIVAL SELECTION
  - FIND ARCHIVED ENTRIES

- AUDITING:
  - FIELDS BEING AUDITED
  - MONITOR A USER
  - PURGE DATA AUDITS
  - PURGE DD AUDITS
  - TURN DATA AUDIT ON/OFF

- SCREENMAN:
  - EDIT/CREATE A FORM
  - RUN A FORM
  - DELETE A FORM
  - PURGE UNUSED BLOCKS
  - PRINT A FORM
  - CUSTOMIZE COLORS
  - CLONE A FORM

- STATISTICS:
  - DESCRIPTIVE STATISTICS
  - SCATTERGRAM
  - HISTOGRAM

- EXTRACT DATA TO FILEMAN FILE:
  - SELECT ENTRIES TO EXTRACT
  - ADD/DELETE SELECTED ENTRIES
  - PRINT SELECTED ENTRIES
  - MODIFY DESTINATION FILE
  - CREATE EXTRACT TEMPLATE
  - UPDATE DESTINATION FILE
  - PURGE EXTRACTED ENTRIES
  - CANCEL EXTRACT SELECTION
  - VALIDATE EXTRACT TEMPLATE

- DATA EXPORT TO FOREIGN FORMAT:
  - DEFINE FOREIGN FILE FORMAT
  - SELECT FIELDS FOR EXPORT
  - CREATE EXPORT TEMPLATE
  - EXPORT DATA
  - PRINT FORMAT DOCUMENTATION

- IMPORT DATA
- BROWSER
- DATA ACCESS CONTROL:
  - SET UP APPLICATION ACTIONS
  - EDIT/CREATE AN ACTION POLICY
  - TEST A POLICY
  - DISABLE A POLICY
  - DELETE A POLICY
  - PRINT ACTIONS/POLICIES
  - POLICY FUNCTIONS

- DATA TYPE OPTIONS:
  - ENTER OR EDIT DATA TYPE FILE
  - ENTER OR EDIT DATA TYPE METHOD FILE

- ENTER OR EDIT DATA TYPE PROPERTY FILE

- ENTITY MAPPING:
  - ENTITY ENTER/EDIT
  - AUTO GEN ENTITY FOR A DD #
  - INQUIRE TO ENTITY FILE

DATA DICTIONARY UTILITIES:

- LIST FILE ATTRIBUTES
- MAP POINTER RELATIONS
- CHECK/FIX DD STRUCTURE
- FIND POINTERS INTO A FILE
- UPDATE THE META DATA DICTIONARY

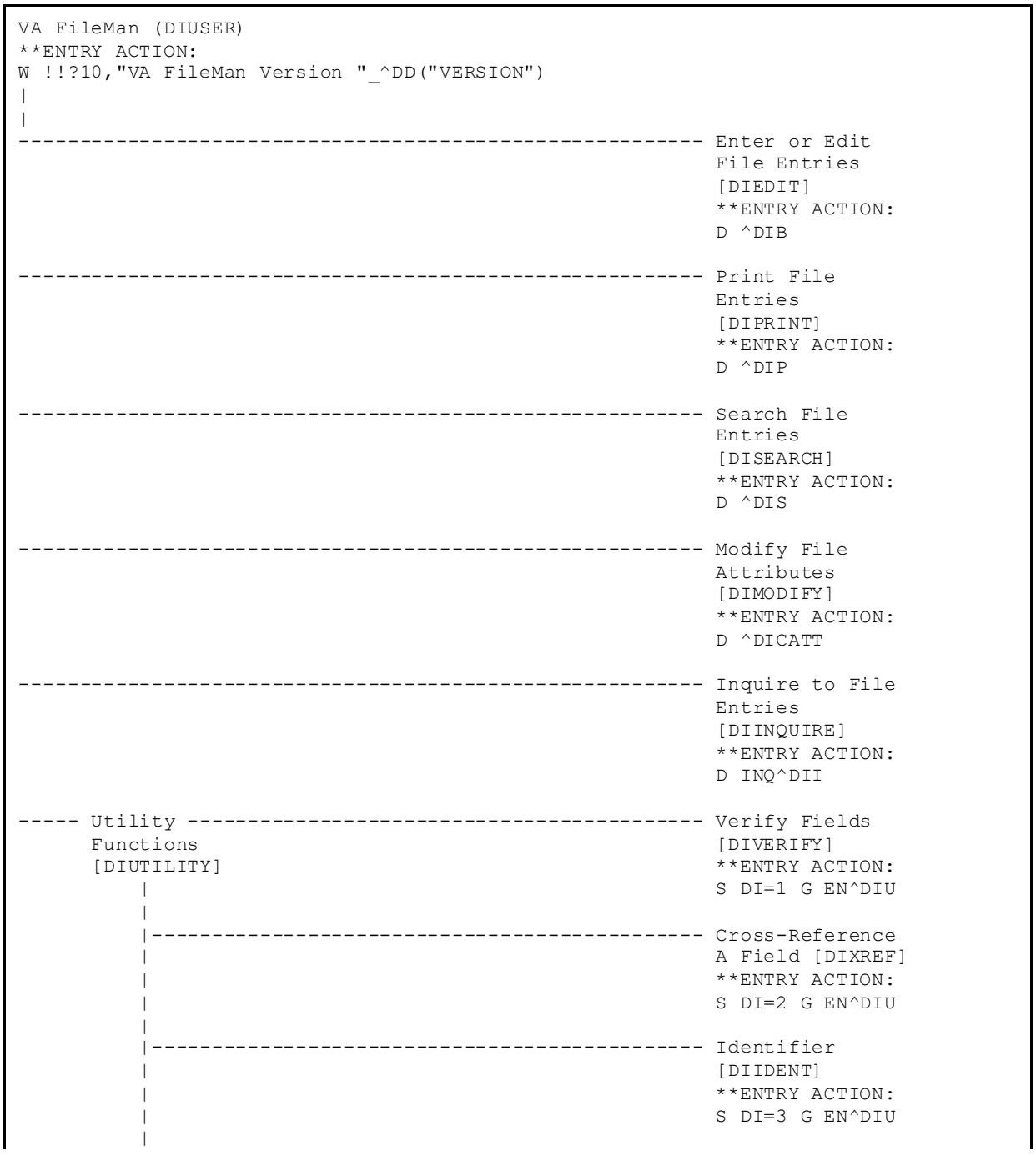
TRANSFER ENTRIES:

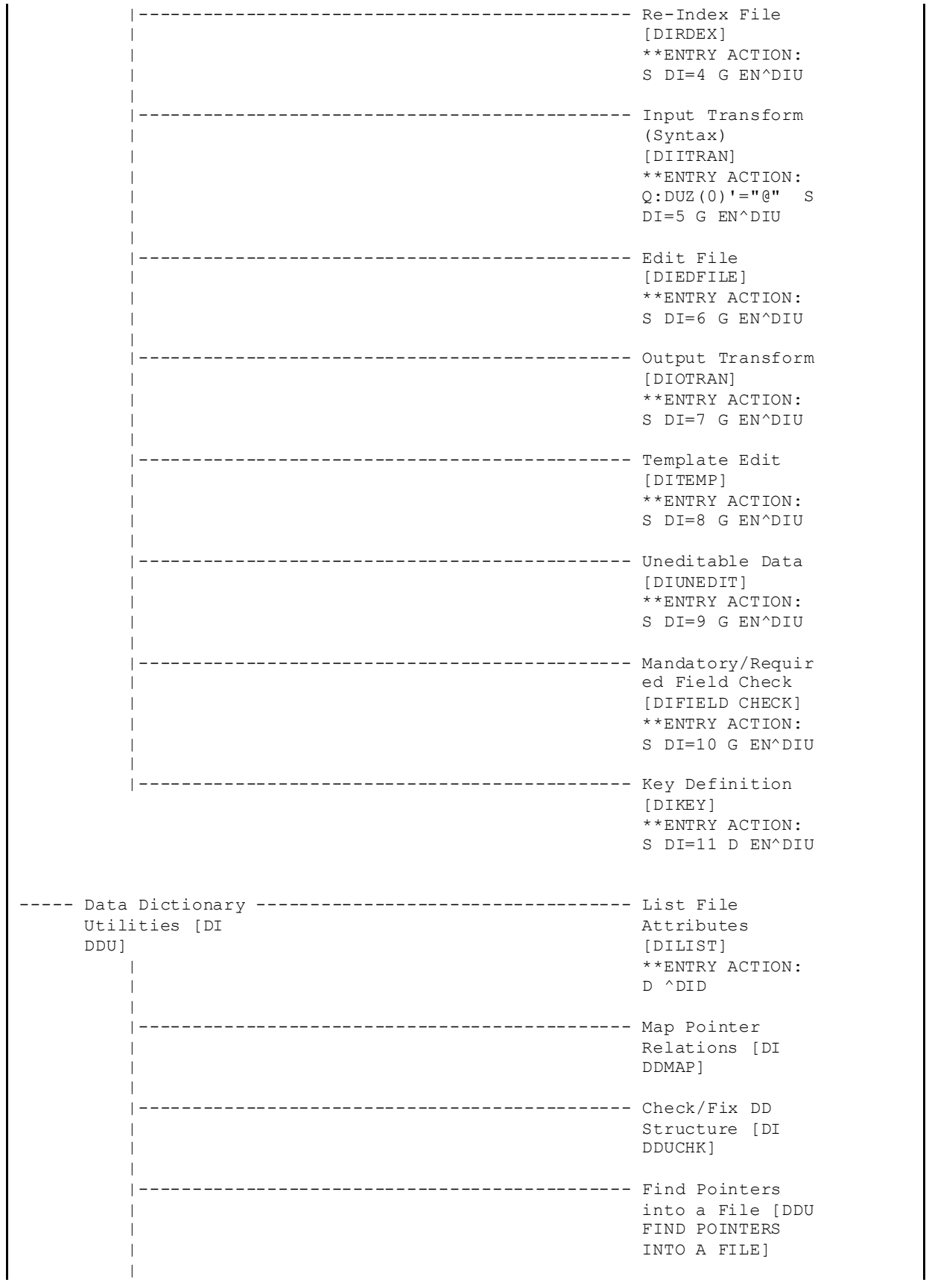
- TRANSFER FILE ENTRIES
- COMPARE/MERGE FILE ENTRIES
- NAMESPACE COMPARE

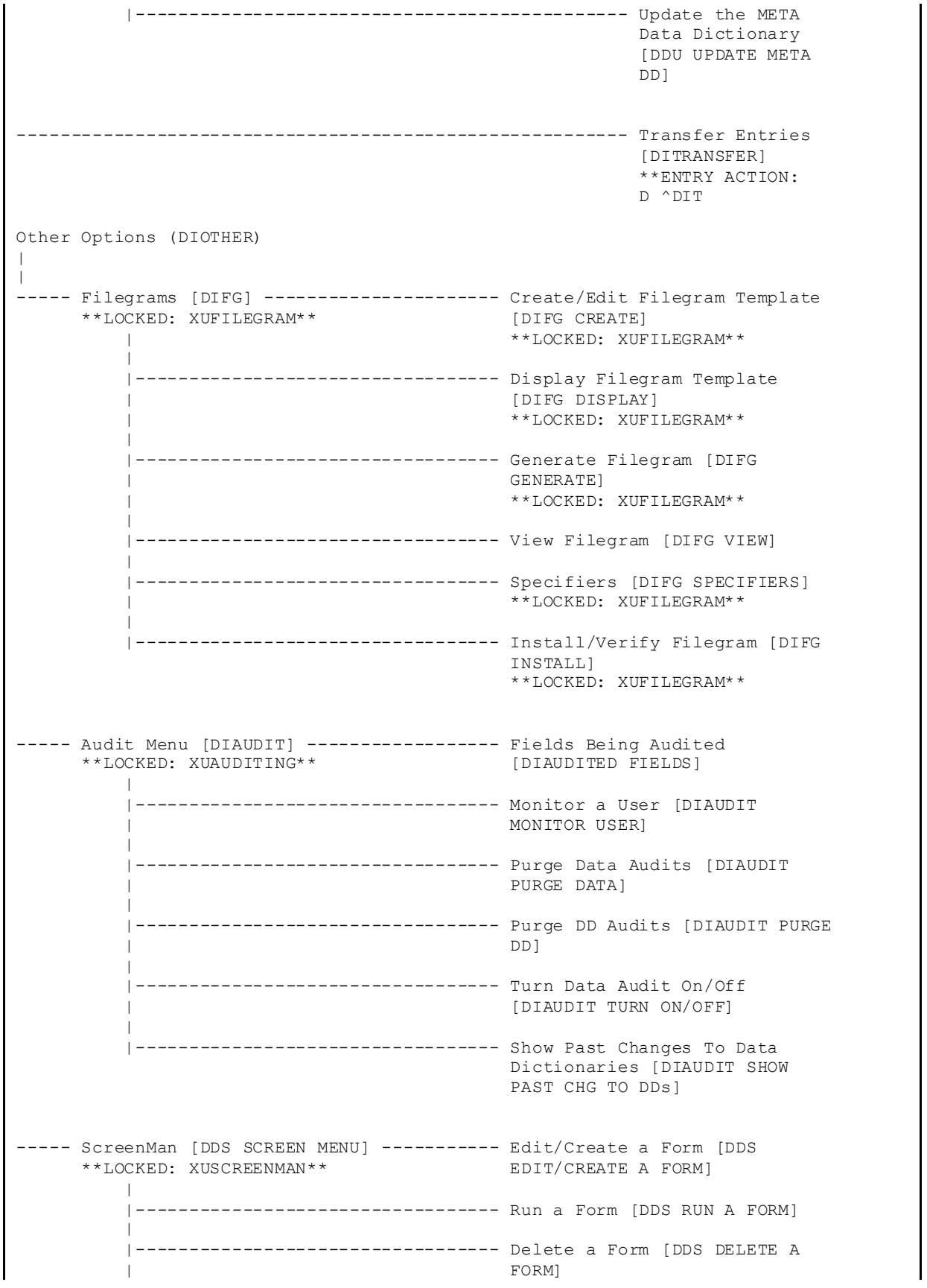
4.6 VA FileMan Options

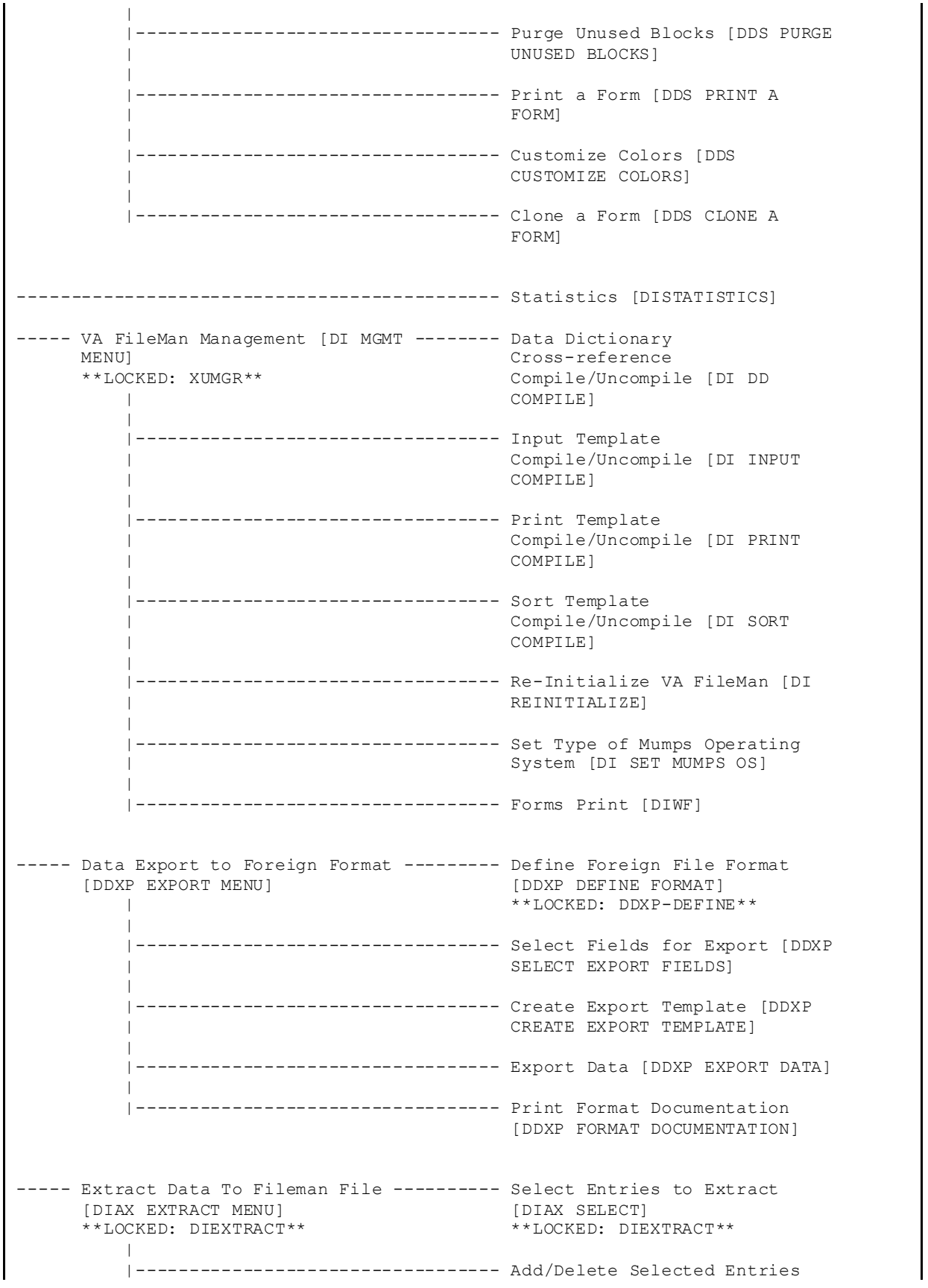
VA FileMan exports the options listed in Figure 3. They are installed during the KIDS install. The top-level **VA FileMan** [DIUSER] menu option can be found on Kernel's EVE menu. The top-level menu option, DMSQ MENU, is *not* attached to any other existing menu; it is standalone and can be assigned as needed.

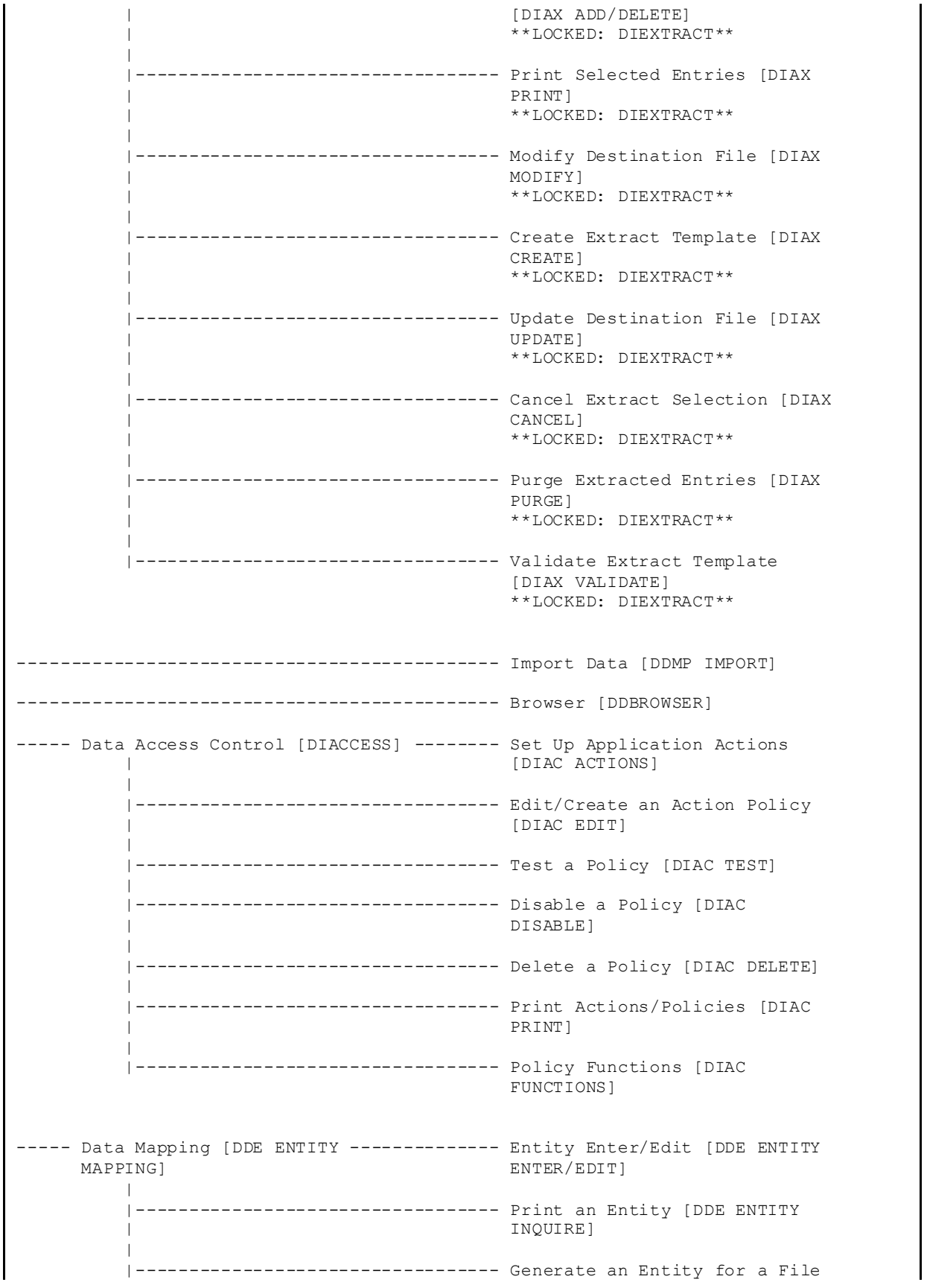
Figure 3: VA FileMan Exported Options Diagrams

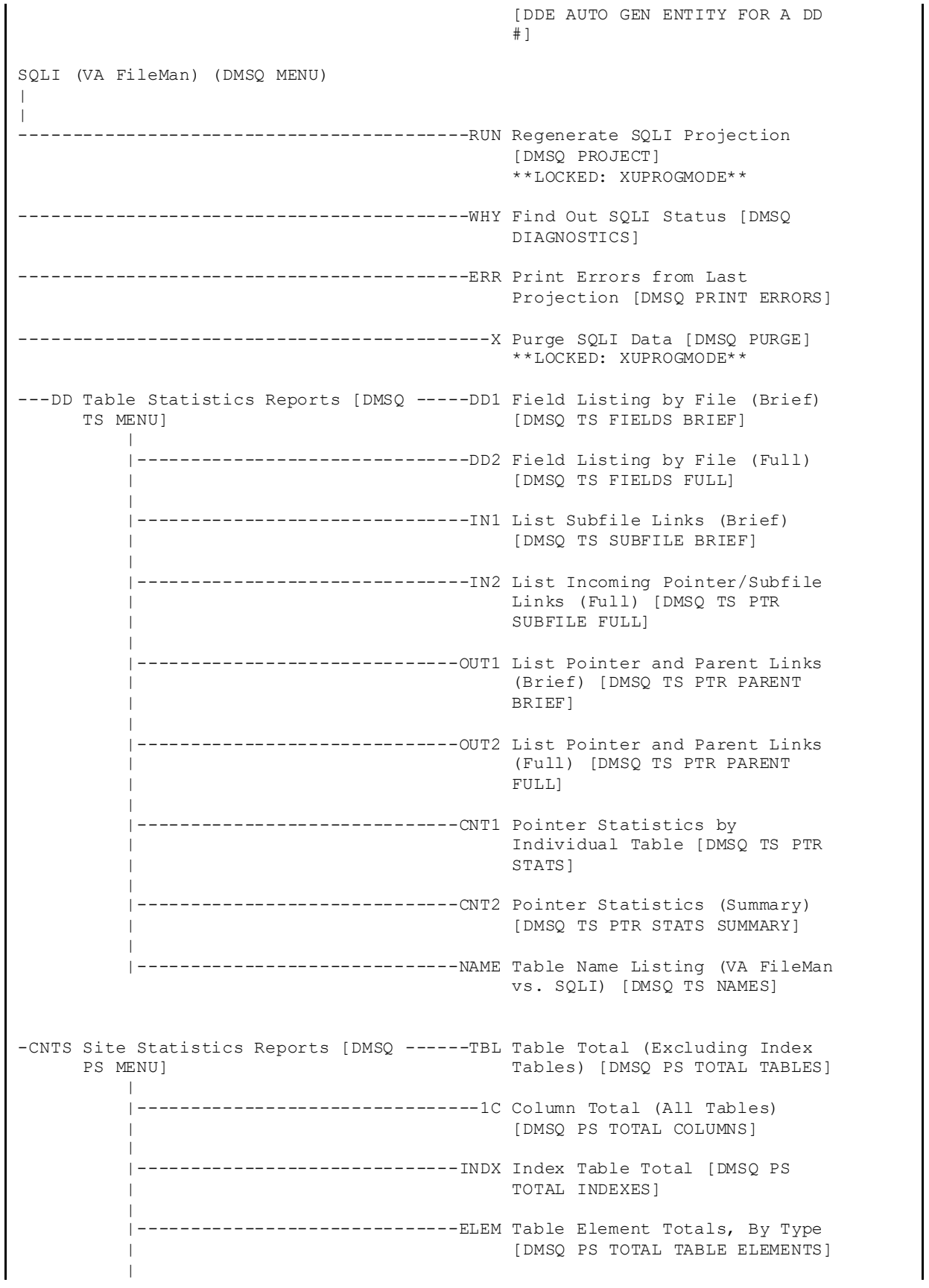



|---------- Re-Index File [DIRDEX] **ENTRY ACTION: S DI=4 G EN^DIU
|---------- Input Transform (Syntax) [DIITRAN] **ENTRY ACTION: Q:DUZ(0)'="@" S DI=5 G EN^DIU
|---------- Edit File [DIEDFILE] **ENTRY ACTION: S DI=6 G EN^DIU
|---------- Output Transform [DIOTRAN] **ENTRY ACTION: S DI=7 G EN^DIU
|---------- Template Edit [DITEMP] **ENTRY ACTION: S DI=8 G EN^DIU
|---------- Uneditable Data [DIUNEDIT] **ENTRY ACTION: S DI=9 G EN^DIU
|---------- Mandatory/Required Field Check [DIFIELD CHECK] **ENTRY ACTION: S DI=10 G EN^DIU
|---------- Key Definition [DIKEY] **ENTRY ACTION: S DI=11 D EN^DIU
----- Data Dictionary Utilities [DIDDU] ---------- List File Attributes [DILIST] **ENTRY ACTION: D ^DID
|---------- Map Pointer Relations [DIDDMAP]
|---------- Check/Fix DD Structure [DIDDUCHK]
|---------- Find Pointers into a File [DDU FIND POINTERS INTO A FILE]


|---------- Update the META Data Dictionary [DDU UPDATE META DD]
---------- Transfer Entries [DITRANSFER] **ENTRY ACTION: D ^DIT
Other Options (DIOTHER)
----- Filegrams [DIFG] ---------- Create/Edit Filegram Template [DIFG CREATE] **LOCKED: XUFILEGRAM**
LOCKED: XUFILEGRAM
|---------- Display Filegram Template [DIFG DISPLAY] **LOCKED: XUFILEGRAM**
|---------- Generate Filegram [DIFG GENERATE] **LOCKED: XUFILEGRAM**
|---------- View Filegram [DIFG VIEW]
|---------- Specifiers [DIFG SPECIFIERS] **LOCKED: XUFILEGRAM**
|---------- Install/Verify Filegram [DIFG INSTALL] **LOCKED: XUFILEGRAM**
----- Audit Menu [DIAUDIT] ---------- Fields Being Audited [DIAUDITED FIELDS] **LOCKED: XUAUDITING**
|---------- Monitor a User [DIAUDIT MONITOR USER]
|---------- Purge Data Audits [DIAUDIT PURGE DATA]
|---------- Purge DD Audits [DIAUDIT PURGE DD]
|---------- Turn Data Audit On/Off [DIAUDIT TURN ON/OFF]
|---------- Show Past Changes To Data Dictionaries [DIAUDIT SHOW PAST CHG TO DDs]
----- ScreenMan [DDS SCREEN MENU] ---------- Edit/Create a Form [DDS EDIT/CREATE A FORM] **LOCKED: XUSCREENMAN**
|---------- Run a Form [DDS RUN A FORM]
|---------- Delete a Form [DDS DELETE A FORM]


Purge Unused Blocks [DDS PURGE UNUSED BLOCKS]
Print a Form [DDS PRINT A FORM]
Customize Colors [DDS CUSTOMIZE COLORS]
Clone a Form [DDS CLONE A FORM]
Statistics [DISTATISTICS]
VA FileMan Management [DI MGMT MENU]
LOCKED: XUMGR
Data Dictionary
Cross-reference
Compile/Uncompile [DI DD COMPILE]
Input Template Compile/Uncompile [DI INPUT COMPILE]
Print Template Compile/Uncompile [DI PRINT COMPILE]
Sort Template Compile/Uncompile [DI SORT COMPILE]
Re-Initialize VA FileMan [DI REINITIALIZE]
Set Type of Mumps Operating System [DI SET MUMPS OS]
Forms Print [DIWF]
Data Export to Foreign Format [DDXP EXPORT MENU]
Define Foreign File Format [DDXP DEFINE FORMAT]
LOCKED: DDXP-DEFINE
Select Fields for Export [DDXP SELECT EXPORT FIELDS]
Create Export Template [DDXP CREATE EXPORT TEMPLATE]
Export Data [DDXP EXPORT DATA]
Print Format Documentation [DDXP FORMAT DOCUMENTATION]
Extract Data To Fileman File [DIAX EXTRACT MENU]
LOCKED: DIEXTRACT
Select Entries to Extract [DIAX SELECT]
LOCKED: DIEXTRACT
Add/Delete Selected Entries


[DIAX ADD/DELETE]
LOCKED: DIEXTRACT
Print Selected Entries [DIAX PRINT]
LOCKED: DIEXTRACT
Modify Destination File [DIAX MODIFY]
LOCKED: DIEXTRACT
Create Extract Template [DIAX CREATE]
LOCKED: DIEXTRACT
Update Destination File [DIAX UPDATE]
LOCKED: DIEXTRACT
Cancel Extract Selection [DIAX CANCEL]
LOCKED: DIEXTRACT
Purge Extracted Entries [DIAX PURGE]
LOCKED: DIEXTRACT
Validate Extract Template [DIAX VALIDATE]
LOCKED: DIEXTRACT
Import Data [DDMP IMPORT]
Browser [DDBROWSER]
Data Access Control [DIACCESS]
Set Up Application Actions [DIAC ACTIONS]
Edit/Create an Action Policy [DIAC EDIT]
Test a Policy [DIAC TEST]
Disable a Policy [DIAC DISABLE]
Delete a Policy [DIAC DELETE]
Print Actions/Policies [DIAC PRINT]
Policy Functions [DIAC FUNCTIONS]
Data Mapping [DDE ENTITY MAPPING]
Entity Enter/Edit [DDE ENTITY ENTER/EDIT]
Print an Entity [DDE ENTITY INQUIRE]
Generate an Entity for a File


[DDE AUTO GEN ENTITY FOR A DD #]
SQLI (VA FileMan) (DMSQ MENU)
RUN Regenerate SQLI Projection [DMSQ PROJECT] **LOCKED: XUPROGMODE**
WHY Find Out SQLI Status [DMSQ DIAGNOSTICS]
ERR Print Errors from Last Projection [DMSQ PRINT ERRORS]
X Purge SQLI Data [DMSQ PURGE] **LOCKED: XUPROGMODE**
---DD Table Statistics Reports [DMSQ TS MENU]
DD1 Field Listing by File (Brief) [DMSQ TS FIELDS BRIEF]
DD2 Field Listing by File (Full) [DMSQ TS FIELDS FULL]
IN1 List Subfile Links (Brief) [DMSQ TS SUBFILE BRIEF]
IN2 List Incoming Pointer/Subfile Links (Full) [DMSQ TS PTR SUBFILE FULL]
OUT1 List Pointer and Parent Links (Brief) [DMSQ TS PTR PARENT BRIEF]
OUT2 List Pointer and Parent Links (Full) [DMSQ TS PTR PARENT FULL]
CNT1 Pointer Statistics by Individual Table [DMSQ TS PTR STATS]
CNT2 Pointer Statistics (Summary) [DMSQ TS PTR STATS SUMMARY]
NAME Table Name Listing (VA FileMan vs. SQLI) [DMSQ TS NAMES]
-CNTS Site Statistics Reports [DMSQ PS MENU]
TBL Table Total (Excluding Index Tables) [DMSQ PS TOTAL TABLES]
1C Column Total (All Tables) [DMSQ PS TOTAL COLUMNS]
INDX Index Table Total [DMSQ PS TOTAL INDEXES]
ELEM Table Element Totals, By Type [DMSQ PS TOTAL TABLE ELEMENTS]

```
          |--------------------------------2C Column Totals, by Table [DMSQ
          |                                   PS TOTAL TABLE COLS]
          |
          |--------------------------------3C Column Totals, by Table
          |                                   (Ordered by # of Columns)
          |                                   [DMSQ PS TOTAL TABLE COLS A]
          |
          |--------------------------------4C Columns in Regular Tables
          |                                   Total [DMSQ PS TOTAL COLUMNS
          |                                   REG]
          |
          |-------------------------------FLDS Columns in Regular Tables,
          |                                   Excluding ID Columns [DMSQ PS
          |                                   COLUMNS REG NOID]
          |
          |--------------------------------DOM Columns by Domain [DMSQ PS
                                              COLUMNS BY DOMAIN]


----------------------------------------GRP Suggest Table Groupings [DMSQ
                                            SUGGEST TABLE GROUPINGS]
```

5 Cross-References

This section contains a description of the MUMPS-type cross-references that exist on fields in VA FileMan files. There are no bulletin or trigger cross-references in these files. All other cross-references are regular types used for lookup or sorting, or both.

The cross-references are grouped by file. The field affected is identified along with the cross-reference's name (or subscript location if there is no name) and a brief description. Many of these cross-references are described in more detail in the data dictionaries. Standard "B" cross-references are not shown. New-Style Indexes are identified by as asterisk (*). No Regular cross-references are shown for the SQLI files (1.521-1.52192).

5.1 INDEX (#.11) File

Table 6: INDEX (#.11) File—Cross-References

Field (Subfile: Field)	X-Ref ID	Description
ROOT FILE	**AC**	VA FileMan finds indexes defined on fields from a particular file.
FILE, NAME	**BB***	The BB index, on the key of the INDEX (#.11) file, lets VA FileMan test potential key values for uniqueness. It is a regular compound index with two fields, the **.01** (FILE) and **.02** (NAME).
NAME	**IX***	This "Regular" index on the NAME (#.02) field allows users to select an index by its name.
CROSS-REFERENCE VALUES: SUBSCRIPT NUMBER	**AC***	VA FileMan finds cross reference values by subscript.
CROSS-REFERENCE VALUES: ORDER NUMBER	**BB***	The uniqueness index of the CROSS-REFERENCE VALUES Multiple field of the INDEX (#.11) file.
CROSS-REFERENCE VALUES: FILE, FIELD	**F**	The **F** index is a whole file compound cross-reference on two fields in the CROSS-REFERENCE VALUES Multiple: FILE (#2) and FIELD (#3).

5.2 KEY (#.31) File

Table 7: KEY (#.31) File—Cross-References

Field	X-Ref ID	Description
FILE, PRIORITY	**AP***	VA FileMan determines the primary key of a file.
UNIQUENESS INDEX	**AU***	VA FileMan determines whether an index is a uniqueness index for a key.
FILE, NAME	**BB***	The **BB** index, the uniqueness index for the Key file's key, lets VA FileMan test potential key values for uniqueness. It is a regular compound index with two fields, the **.01** (File) and **.02** (Key Name).
FIELD: FIELD	**Trigger**	The FILE (.01) of the parent record is triggered into FILE (.02) when FIELD (.01) is edited.
FIELD: FIELD, FILE	**BB***	The **BB** index, on the key of the FIELDS Multiple of the KEY (#.31) file, allows VA FileMan to test potential key values for uniqueness. It is a regular compound index with two fields.
FIELD: FILE, FIELD	**F***	The **F** index, a whole file compound cross-reference on the key of the FIELDS Multiple of the KEY (#.31) file, allows VA FileMan to determine the keys of which a field is part. This is essential for identifying the key value uniqueness tests that must be done when a field value changes.
FIELD: SEQUENCE NUMBER, FIELD, FILE	**S***	The **S** index, a compound index on all fields of the FIELDS Multiple of the KEY (#.31) file, allows VA FileMan to step through the key fields in sequence. This is essential for prompting, returning values, as well as for the generation of each key's uniqueness index.

5.3 PRINT TEMPLATE (#.4) File

Table 8: PRINT TEMPLATE (#.4) File—Cross-References

Field	X-Ref ID	Description
NAME	**F_file#**	This cross-reference is used to quickly find all PRINT templates associated with a particular file.
	AF	This cross-reference sets up an "**AF**" cross-reference node for each field in a compiled PRINT template. The cross-reference has the form: `^DIPT("AF",file#,field#,print template#)=""`
FILE	**F_file#**	This cross-reference is used to quickly find all PRINT templates associated with a particular file.
TEMPLATE TYPE	**FG**	This cross-reference is used to do a quick lookup of FILEGRAM-type of PRINT templates.
	EX	This cross-reference is used to do a quick lookup of EXTRACT-type PRINT templates.
CANONIC FOR THIS FILE	**CANONIC**	This cross-reference is used to identify files that have a Canonic Print Template assigned. The structure of the cross-reference is: **^DIPT("CANONIC", File#, IEN)** Where **File#** identifies the file that has a Canonic PRINT template and IEN is the internal entry number of the Canonic PRINT template assigned to that file.

5.4 SORT TEMPLATE (#.401) File

Table 9: SORT TEMPLATE (#.401) File—Cross-References

Field	X-Ref ID	Description
NAME	**F_file#**	This cross-reference is used to quickly find all SORT templates associated with a particular file.
FILE	**F_file#**	This cross-reference is used to quickly find all SORT templates associated with a particular file.
CANONIC FOR THIS FILE	**CANONIC**	This cross-reference is used to identify files that have a Canonic Sort Template assigned. The structure of the cross-reference is: **^DIBT("CANONIC", File#, IEN)** Where **File#** identifies the file that has a Canonic SORT template and IEN is the internal entry number of the Canonic SORT template assigned to that file.

5.5 INPUT TEMPLATE (#.402) File

Table 10: INPUT TEMPLATE (#.402) File—Cross-References

Field	X-Ref ID	Description
NAME	**F_file#**	This cross-reference is used to quickly find all INPUT templates associated with a particular file.
	AF	This cross-reference sets up an "**AF**" cross-reference node for each field in a compiled INPUT template. The cross-reference has the form: `^DIE("AF",file#,field#,input template#)=""`
FILE	**F_file#**	This cross-reference is used to quickly find all INPUT templates associated with a particular file.
CANONIC FOR THIS FILE	**CANONIC**	This cross-reference is used to identify files that have a Canonic Edit Template assigned. The structure of the cross-reference is: **^DIE("CANONIC", File#, IEN)** Where **File#** identifies the file that has a Canonic EDIT template and IEN is the internal entry number of the Canonic EDIT template assigned to that file.

5.6 FORM (#.403) File

Table 11: FORM (#.403) File—Cross-References

Field	X-Ref ID	Description
NAME	**F1**	This cross-reference is used to quickly find all ScreenMan forms associated with a particular file.
	AY	This cross-reference merely documents the existence of data stored under **^DIST(.403,form IEN,"AY")**. This is where the compiled data for a form is stored.
PAGE NAME (subfield of PAGE Multiple)	**C**	This cross-reference stores the PAGE NAME converted to uppercase characters.
PRIMARY FILE	**F**	This cross-reference is used to quickly find all ScreenMan forms associated with a particular file.
PAGE: IS THIS A POP UP PAGE?		This MUMPS cross-references ensures that no Header block is present if it is a pop-up page.
PAGE: HEADER BLOCK	**AC**	This cross-reference ensures that no header block, next page, or previous page is associated with a pop-up page.
PAGE: BLOCK: BLOCK NAME	**AB**	This cross-reference facilitates identifying the Forms on which a Block is used.
PAGE: BLOCK: BLOCK ORDER	**AC**	This cross-reference ensures that Block Order Numbers are unique within a page.

5.7 BLOCK (#.404) File

Table 12: BLOCK (#.404) File—Cross-References

Field	X-Ref ID	Description
CAPTION (subfield of FIELD Multiple)	**C**	This cross-reference is used for lookup of fields by CAPTION. It is also used for ^-jumping.
UNIQUE NAME (subfield of FIELD Multiple)	**D**	This cross-reference stores the UNIQUE NAME converted to uppercase characters.

5.8 FOREIGN FORMAT (#.44) File

Table 13: FOREIGN FORMAT (#.44) File—Cross-References

Field	X-Ref ID	Description
OTHER NAME FOR FORMAT: OTHER NAME FOR FORMAT	**C**	This cross-reference allows look-ups for formats based on OTHER NAME FOR FORMAT.

5.9 IMPORT TEMPLATE (#.46) File

Table 14: IMPORT TEMPLATE (#.46) File—Cross-References

Field	X-Ref ID	Description
NAME	**F1**	Creates an index under **F_file#** that is used for lookup when the file number is known.
PRIMARY FILE	**F**	Same as **F1**.

5.10 DD AUDIT (#.6) File

Table 15: DD AUDIT (#.6) File—Cross-References

Field	X-Ref ID	Description
DATE UPDATED	**D**	A regular cross-reference supporting lookups on the DATE UPDATED field.
USER	**E**	A regular cross-reference supporting lookups on the USER field.

5.11 DATA TYPE (#.81) File

Table 16: DATA TYPE (#.81) File—Cross-References

Field	X-Ref ID	Description
INTERNAL REPRESENTATION	**C**	A regular cross-reference supporting lookups on the INTERNAL REPRESENTATION field.

5.12 COMPILED ROUTINE (#.83) File

Table 17: COMPILED ROUTINE (#.83) File—Cross-References

Field	X-Ref ID	Description
IN USE	**C**	This cross-reference is used to control when a routine number is available for use in creating a compiled sort routine, during the FileMan sort/print option.

5.13 LANGUAGE (#.85) File

Table 18: LANGUAGE (#.85) File—Cross-References

Field	X-Ref ID	Description
TWO LETTER CODE	**C**	Regular new style index on two letter language codes
THREE LETTER CODE	**D**	Regular new-style index for three letter abbreviations for languages
ALTERNATE THREE LETTER CODE	**E**	This adds entries to the **D** index for the three-letter code a la the mnemonic style.
ALTERNATE NAME: ALTERNATE NAME	**F**	Whole file cross-reference for ALTERNATE NAME Multiple allowing look-up by ALTERNATE NAME.

5.14 META DATA DICTIONARY (#.9) File

Table 19: META DATA DICTIONARY (#.9) File—Cross-References

Field	X-Ref ID	Description
DATA DICTIONARY NUMBER	**AFF**	The **AFF** cross-reference is a multi-field MUMPS cross-reference based on the DATA DICTIONARY NUMBER and FIELD NUMBER fields. It stores data into the same location as the **AFF2** cross-reference on the FIELD NUMBER field. Its structure is: **^DDD("AFF",file_number,field_number,IEN)**
FIELD NUMBER	**AFF2**	The **AFF2** cross-reference is a multi-field MUMPS cross-reference based on the DATA DICTIONARY NUMBER and FIELD NUMBER fields. It stores data into the same location as the **AFF** cross-reference on the DATA DICTIONARY NUMBER field. Its structure is: **^DDD("AFF",file_number,field_number,IEN)**

Field	X-Ref ID	Description
LOOKUP TERM	**C**	The **C** cross-reference is a regular cross-reference on the LOOKUP TERM field, supporting lookups on field labels.

5.15 FILE (#1) of Files

Table 20: FILE (#1) of Files—Cross-References

Field	X-Ref ID	Description
NAME	**AD**	This cross-reference sets and kills the "**GL**" node for the file. This node has the form: `^DIC(file#,0,"GL")=file's global location`
	AE	This cross-reference sets and kills the "**NM**" node for the file. This node has the form: `^DIC(file#,0,"NM")=file's name`
APPLICATION GROUP: APPLICATION GROUP	**AC**	This whole file cross-reference allows file look-ups by Application Group (Package).
TRANSLATION: TRANSLATION	**ALANG**	This cross-reference facilitates checking if a particular language has a translation of the file name. Its structure is: **^DIC("ALANAG"_LanguageFileIEN,Translation,FileNumber)**

5.16 AUDIT (#1.1) File

Table 21: AUDIT (#1.1) File—Cross-References

Field	X-Ref ID	Description
DATE/TIME RECORDED	**C**	The cross-reference allows looking up an Audit record by date and time.
USER	**D**	The cross-reference allows looking up an Audit record by user.

5.17 ARCHIVAL ACTIVITY (#1.11) File

Table 22: ARCHIVAL ACTIVITY (#1.11) File—Cross-References

Field	X-Ref ID	Description
FILE	**C**	This cross-reference allows looking up an Archive by File name.

5.18 ENTITY (#1.5) File

Table 23: ENTITY (#1.5) File—Cross-References

Field	X-Ref ID	Description
ENTITY (#.08)	**AD**	The cross-reference allows looking up an ENTITY record for Input Transform, to look back up the tree and ensure item is *not* an ancestor.
NAME (#.01)	**B**	The cross-reference allows looking up an ENTITY record by name.
NUMBER (#.02)	**F**	The cross-reference allows finding entities by primary file number.
DISPLAY NAME (#.1)	**FHIR**	Compound cross-reference. Retrieves FHIR entities by display name and file number.
DEFAULT FILE NUMBER (#.02)		
DISPLAY NAME (#.1)	**SDA**	Compound cross-reference. Retrieves SDA entities by display name and file number.
DEFAULT FILE NUMBER (#.02)		

5.19 SQLI_TABLE_ELEMENT (#1.5216) File

Table 24: SQLI_TABLE_ELEMENT (#1.5216) File—Cross-References

Field	X-Ref ID	Description
E_TABLE	**G**	Table element by table, by name.
E_TYPE	**F**	Table element by table, by type.

5.20 SQLI_COLUMN (#1.5217) File

Table 25: SQLI_COLUMN (#1.5217) File—Cross-References

Field	X-Ref ID	Description
C_FIELD	**D**	Column by VA FileMan file number, by field number.

5.21 SQLI_PRIMARY_KEY (#1.5218) File

Table 26: SQLI_PRIMARY_KEY (#1.5218) File—Cross-References

Field	X-Ref ID	Description
P_SEQUENCE	**C**	Primary key by table, by sequence.

6 Archiving and Purging

6.1 Archiving

There are no package-specific archiving procedures in VA FileMan.

The generic archiving tool for VistA is a part of VA FileMan. It is described in the *VA FileMan Advanced User Manual*.

REF: For more information on archiving, see the "Archiving" section in the VA *FileMan Advanced User Manual*.

The Extract Tool provides a means of archiving data into a VA FileMan file. It is also described in the *VA FileMan Advanced User Manual*.

REF: For more information on the Extract Tool, see the "Extract Tool" section in the "Archiving" section in the VA *FileMan Advanced User Manual*.

6.2 Purging

Within VA FileMan, the only files that might grow large enough to require purging of data are the audit files:

- AUDIT (#1.1)
- DD AUDIT (#.6)

These files capture information about changes to data and to data dictionaries, respectively. The user audit is started and stopped by using the Monitor a User option on the Auditing submenu. Starting with VA FileMan 22.2, the data dictionary audit will always be on. The amount of data accumulated is dependent both on the scope of the audit and its duration. Options are available to purge the AUDIT (#1.1) (Purge Data Audits) and the DD AUDIT (#.6) file (Purge DD Audits). Purging the audit files is optional. Decisions to purge *must* be made based on the size of the files and any need to retain the audit data.

REF: For instructions on the use of the Auditing options, see the "Auditing" section in the VA *FileMan Advanced User Manual*.

The Purge Stored Entries option on the Archiving submenu removes the data archived from the primary file and from the ARCHIVAL ACTIVITY (#1.11) file when the archiving process is complete. The Purge Stored Entries option should be run when each archiving action is finished in order to remove the archived data and clean up the files.

The Purge Extracted Entries option on the Extract Tool submenu removes extracted data from the primary file and from the ARCHIVAL ACTIVITY (#1.11) file when the extract process is complete. This option should be run when using the Extract Tool for archiving purposes to remove extracted data.

7 External Relationships

As distributed with a Kernel Installation and Distribution System (KIDS) build, VA FileMan 22.2 is dependent on a pre-existing installation of Kernel. The VA FileMan 22.2 Installation Guide does not describe how to install VA FileMan without the Kernel. In other words, a so-called standalone installation is not explicitly supported. However, almost all of the functionality of VA FileMan can be implemented without Kernel by installing the VA FileMan 22.2 routines and running **^DINIT**. Describing how to accomplish a standalone install is beyond the scope of this documentation set.

VA FileMan must be installed on a system running an implementation of ANSI Standard M. The KIDS distribution described here assumes installation on a Caché system. Information in the MUMPS OPERATING SYSTEM (#.07) file and Kernel-supplied **%ZOSF** nodes is used to perform functions that are operating-system dependent. Operating Systems other than Caché can be accommodated based on entries in the MUMPS OPERATING SYSTEM (#.07) file. Again, processes for running VA FileMan on operating systems other than Caché are beyond the scope of these documents.

REF: For details of installing VA FileMan, see the *VA FileMan 22.2 Installation Guide.*

Although not part of VA FileMan, the Kernel's PACKAGE (#9.4) file *must* be present on your system to use the DIFROM routines to export software packages. The Package file installation is *not* included in this distribution of VA FileMan 22.2

CAUTION: The Kernel Installation and Distribution System (KIDS) replaced the use of DIFROM as the method of exporting software packages in the VA. The version of DIFROM released with VA FileMan 22.2 will transport the new Key and Index structures.

VA FileMan's capability is augmented when it is installed with Kernel and MailMan. Specifically, VA FileMan 22.2 is designed to work with Kernel 8.0 or later. For example, the following additional functionality is available when VA FileMan is installed with Kernel:

- User security via the NEW PERSON (#200) file
- Control of file access
- More sophisticated menu presentation
- Device control

- Queuing

The following additional functionality is available when VA FileMan is installed with MailMan:

- Bulletins, one of VA FileMan's cross-references, become operational when MailMan is installed to deliver the messages.
- Filegram options also require MailMan.

Kernel allows networking two CPUs with different operating systems. Kernel provides this ability by retrieving the type of operating system from **^%ZOSF("OS")**. This global does *not* have to be replicated or translated; thus, a separate copy of the global can be stored on each CPU. When running standalone VA FileMan, the type of operating system is retrieved either from the second piece of **^%ZOSF("OS")**, if the **DINZMGR** was run, or from **^DD("OS")**. **^DD("OS")** is the global location of the MUMPS OPERATING SYSTEM (#.7) file. The **^DD** global *must* always be either replicated or translated across systems. In any case, VA FileMan uses the local **DISYS** variable to store the value of the current operating system. VA FileMan finds some operating system-specific code in nodes descending from **^DD("OS",DISYS)**; other code is found in **^%ZOSF** nodes.

VA FileMan exports options and security keys with the **DI** and **DD** namespace for use by Kernel.

NOTE: Throughout the VA FileMan manuals, specific reference is made to Kernel or MailMan when either is needed for a function to work.

7.1 DBA Approvals and Database Integration Control Registrations (ICRs)

The Database Administrator (DBA) maintains a list of Integration Control Registrations (ICRs) or mutual agreements between software developers allowing the use of internal entry points or other software-specific features that are *not* available to the general programming public.

7.1.1 ICRs—Current List for VA FileMan as Custodian

To obtain the current list of ICRs, if any, to which the VA FileMan software (DI) is a custodian, perform the following procedures:

1. Sign onto the **FORUM** system (forum.va.gov).
2. Go to the **DBA** menu [DBA].
3. Select the **Integration Agreements Menu** option [DBA IA ISC].
4. Select the **Custodial Package Menu** option [DBA IA CUSTODIAL MENU].
5. Choose the **ACTIVE by Custodial Package** option [DBA IA CUSTODIAL].
6. When this option prompts you for a package, enter **VA FILEMAN** or **DI**.

7. All current ICRs to which the VA FileMan software is a custodian are listed.

7.1.2 ICRs—Detailed Information

To obtain detailed information on a specific integration control registration, perform the following procedures:

1. Sign onto the **FORUM** system (forum.va.gov).
2. Go to the **DBA** menu [DBA].
3. Select the **Integration Agreements Menu** option [DBA IA ISC].
4. Select the **Inquire** option [DBA IA INQUIRY].
5. When prompted for "INTEGRATION REFERENCES," enter the specific integration control registrations number of the ICR you would like to display.
6. The option then lists the full text of the ICR you requested.

7.1.3 ICRs—Current List for VA FileMan as Subscriber

To obtain the current list of ICRs, if any, to which the VA FileMan software (DI) is a subscriber, perform the following procedures:

1. Sign onto the **FORUM** system (forum.va.gov).
2. Go to the **DBA** menu [DBA].
3. Select the **Integration Agreements Menu** option [DBA IA ISC].
4. Select the **Subscriber Package Menu** option [DBA IA SUBSCRIBER MENU].
5. Choose the **Print ACTIVE by Subscribing Package** option [DBA IA SUBSCRIBER].
6. When prompted with "START WITH SUBSCRIBING PACKAGE," enter **VA FILEMAN** (uppercase). When prompted with "GO TO SUBSCRIBING PACKAGE," enter **VA FILEMAN** (uppercase).
7. All current ICRs to which the VA FileMan software is a subscriber are listed.

8 Internal Relationships

All options can be independently invoked.

None of the options require any special setup in order to run successfully.

9 Package-Wide Variables

VA FileMan package-wide or key variables that can be assumed to be defined at all times are listed in Table 27:

Table 27: Package-Wide Variables

Variable	Description
DUZ	The internal entry number from the NEW PERSON (#200) file.
DUZ(0)	The variable defining the user's access.
DUZ("LANG")	If running Kernel 8.0 or later, this variable refers to the language of the current user.
DT	The current date in VA FileMan internal format.
DTIME	The integer value of the number of seconds the user has to respond to a timed read.
U	The up-arrow (caret).

In addition, the variable in Table 28 has a special meaning for VA FileMan although it is *not* always defined:

Table 28: Package-Wide Variables—DISY (Special Meaning)

Variable	Description
DISYS	The current M operating system—pointer to the MUMPS OPERATING SYSTEM (#.7) file contained in the first piece of **^DD("OS")** and, if using Kernel, in the second piece of **^%ZOSF("OS")**.

9.1 Standards and Conventions (SAC) Exemptions

Beginning January 1, 1995, VA FileMan has been granted exemptions from the following standards by the Programming Standards and Conventions Committee (SACC).

9.1.1 STANDARD SECTION: 4B–Package-wide variables

Beginning December 22, 1994, VA FileMan is exempted from **KILL**ing the listed variables in the following calls:

Table 29: List of Variables VA FileMan is Exempted from KILLing

Supported Reference	Variables
DIC	**DA**
FILE^DICN	**DA**
DIE	**%,D,D0,DI,DQ,X,D1,%X,%Y**
DIK	**%,DA,DIC, X, Y**
EN1^DIP	**X**
EN^DIQ1	**%,D0,I,J,X,Y,C**

9.1.2 STANDARD SECTION: 6D–FM compatibility

- The following globals are exempt from VA FileMan compatibility:
  - **^DISV**
  - **^DOSV**

- VA FileMan may set a *non*-VA FileMan compatible node [e.g., **^XXX(File#, IEN,-9)**] to record information about archival activity and may set *non*-VA FileMan compatible nodes **^(3)** and **^(2)** to store old and new values of any audited field.

10 Globals

VA FileMan's globals are listed below:

- **^DD**
- **^DDD**
- **^DDA**
- **^DDE**

- **^DI**
- **^DIA**
- **^DIAR**
- **^DIBT**
- **^DIC**
- **^DIE**
- **^DIPT**
- **^DIST**
- **^DISV**
- **^DIT**
- **^DIZ**
- **^DMSQ**
- **^DOPT**
- **^DOSV**
- **^TMP**
- **^UTILITY**
- **^%ZOSF**

REF: For a description of these globals, see Table 3.

The **^UTILITY** and **^TMP** globals are temporary globals used and then **KILL**ed by many VA FileMan options. If VA FileMan is used with Kernel, nodes in **^%ZOSF** are set up during Kernel's installation.

There is a supported entry point to the **^DD** global: **^DD("DD")**. Its use is explained in the "X ^DD("DD")—Another Way to Convert Dates" section in the "Date/Time Utilities" section found in the "Classic FileMan" section (listed by category) in the "Major APIs" section in the *VA FileMan Developer's Guide*.

REF: For specific information on **^%DT**, see the "^%DT" section in the "Classic FileMan API" section in the "Major APIs" section in the *VA FileMan Developer's Guide.*

^DD("VERSION") can be read to get the version number of the VA FileMan package that exists in the system.

10.1 Global Journaling, Translation, and Replication

No VA FileMan-specific actions are needed for global journaling, translation, or replication in the VA environment.

11 Security

VA FileMan (aka File Manager) is the database management system for Veterans Health Information Systems and Technology Architecture (VistA). As such, it provides security on a file, field, and template level. This security is based on a string of characters stored in the **DUZ(0)** local variable. You can find the details of the data security system imposed by VA FileMan in the *VA FileMan Advanced User Manual*. The security mechanisms described apply to the files and data sent with the VA FileMan software as well as to the files created by other applications and by users.

VA FileMan is a collection of routines written in MUMPS (M) that allow the user the capability of reading and writing to files. The routines are pre-written for users to access in creating APIs for access to data in their "namespace". The modifications were all pertaining to these routines and did *not* change the security boundary nor any methods of access to the data that did *not* already exist under an authority to operate (ATO) sustained by the Regions. VA FileMan experts extensively tested and verified all fixes and ran existing utilities, such as "**XINDEX**" to verify the validity of said routines.

REF: For specific information on VA FileMan's data security, see the "Data Security" section in the "Security" section in the *VA FileMan Advanced User Manual*.

When used with Kernel, other types of access control are available. If Kernel's File Access Security system has been implemented on your system, you can use it to control user access to files.

REF: Kernel's Sign-on/Security component is described in the *Kernel 8.0 and Kernel Toolkit 7.3 Systems Management Guide*.

When you use VA FileMan within the Kernel's menu system, you are subject to the Kernel's security requirements:

- You *must* enter correct Access and Verify codes.
- You can only use menus and options to which you have been granted access.
- You *must* have the proper security keys to use certain locked options.

Most VA FileMan options are accessed through the DIUSER menu. This menu is usually located on the EVE menu distributed with Kernel. SQLI-specific options are found on DMSQ menu.

REF: For a diagram of the complete menu tree for VA FileMan, see Figure 3 in the "VA FileMan Kernel Options" section.

11.1 Security Management

This software was developed at the Department of Veterans Affairs (VA) by employees of the Federal Government in the course of their official duties. Pursuant to title 17 Section 105 of the United States Code this software is *not* subject to copyright protection and is in the public domain. VA assumes no responsibility whatsoever for its use by other parties, and makes no guarantees, expressed or implied, about its quality, reliability, or any other characteristic. We would appreciate acknowledgement if the software is used. This software can be redistributed and/or modified freely provided that any derivative works bear some notice that they are derived from it, and any modified versions bear some notice that they have been modified.

11.2 Mail Groups and Alerts

VA FileMan does *not* make use of mail groups or alerts.

11.3 Remote Systems

VA FileMan does *not* transmit data to any remote system, facility, or database.

11.4 Interfacing

No *non*-VA products are embedded in or required by VA FileMan, other than those provided by the underlying operating systems.

11.5 Electronic Signatures

Electronic signatures are *not* used within VA FileMan.

11.6 Security Keys

VA FileMan options are locked with the security keys described in Table 30. The security keys in the XU namespace are distributed by Kernel; however, they lock VA FileMan options. The two remaining security keys are distributed by VA FileMan and are installed when DINIT is run:

Table 30: VA FileMan Security Keys

Security Key	Description
XUAUDITING	Use this security key to access the Auditing menu or to run any of the Auditing options.
XUFILEGRAM	Use this security key to access the Filegram menu or to run any of the Filegram options; except the View Filegram option, for which no security key is required.

Security Key	Description
XUMGR	Use this security key for users who act as site management staff. It is required in order to access the VA FileMan Management menu. It is also needed to access many Kernel options.
XUPROGMODE	Use this security key to access the SQLI Regenerate SQLI Projection and Purge SQLI Data options.
XUSCREENMAN	Use this security key to access the ScreenMan menu.
DDXP-DEFINE	Use this security key to access the Export Tool's Define Foreign File Format option.
DIEXTRACT	Use this security key to access the Extract Data to FileMan File menu.

11.7 File Security

Files with numbers less than **two** (**2**) belong to VA FileMan. In general, these files *cannot* be directly accessed. You can access them only through the menu options. Those users who are granted programmer access [**DUZ(0)="@"**] can directly read and manipulate data in VA FileMan files. However, it is *strongly recommended* that changes to data in such files only be made through documented VA FileMan utilities.

11.8 References

The following directive specifies that VA FileMan routines and files should *not* be altered:

Veterans Health Administration (VHA) Directive 6402

11.9 Official Policies

Modification of any part of the VA FileMan software is *not permitted* as per VHA Directive 6402.

Distribution of the VA FileMan software is unrestricted (see the "Software Disclaimer" section).

12 Troubleshooting

For product support, contact the National Help Desk.

12.1 How to Obtain Technical Information Online

Exported VistA M Server-based software file, routine, and global documentation can be generated through the use of Kernel, MailMan, and VA FileMan utilities.

NOTE: Methods of obtaining specific technical information online are indicated where applicable under the appropriate section.

12.2 Help at Prompts

VistA M Server-based software provides online help and commonly used system default prompts. Users are encouraged to enter question marks at any response prompt. At the end of the help display, you are immediately returned to the point from which you started. This is an easy way to learn about any aspect of the software.

Glossary

Table 31: Glossary

Term	Description
ANSI STANDARD MUMPS	American National Standards Institute (ANSI) computer language used by VA FileMan. Also called M. The acronym MUMPS stands for Massachusetts General Hospital Utility Multiprogramming System.
ARCHIVING	The storing of historical or little used data offline (often on tape).
AUDITING	The monitoring and recording of computer use. VA FileMan audits can log changes to data values in files and to the structure of the file itself.
BROWSER	An interactive application in VA FileMan that displays ASCII text on a terminal that supports a scroll region. The text can be in the form of a VA FileMan WORD-PROCESSING-type field or sequential local or global array. The user is allowed to navigate freely within the document.
CALLABLE ENTRY POINTS	Places in a VA FileMan routine that can be called from an application program.
CHECKSUM VALUE	A number computed for each routine in a package. The number is used to verify that the routine is uncorrupted and unchanged. Any coding change to a routine changes its checksum value.
CROSS-REFERENCE	In VA FileMan, an attribute of a field that identifies an action to take place when the value of the field is changed. Often, the action is the placement of the field's value into an index. Beginning in Version 22.0 of VA FileMan, the INDEX file allows creation of indexes that contain more than one data field. Thus, they become an attribute of the file, rather than of a single field. The action described in the INDEX file entry happens when any of the involved fields is changed.
DATA DICTIONARY	A data dictionary (DD) contains the definitions of a file's elements (fields or data attributes), relationships to other files, and structure or design.
DATABASE MANAGEMENT SYSTEM	A collection of software that handles the storage, retrieval and updating of records in a database.
DBS	Database Server: An Application Programming Interface (API) for VA FileMan that updates the database in a non-interactive mode. VA FileMan passes information that needs to be displayed to the user to the calling routine in arrays.
DBMS	Database Management System.

Term	Description
DEVICE	A terminal, printer, modem or other type of hardware or equipment associated with a computer. A Host file of an underlying operating system may be treated like a device in that it can be written to (e.g., for spooling).
DHCP	The Decentralized Hospital Computer Program, see “VistA.”
DIRECT MODE UTILITY	An entry point into a routine that can only be called from programmer mode, see “Callable Entry Points.”
DSM FOR OPENVMS	The current name for **VAX DSM(V6)**. One of the M operating systems supported by VA FileMan.
ENTRY	For VA FileMan, an instance of a file; a set of logically related data in a file; a record.
FIELD	In an entry, a specified area used for the value of a data attribute. The data specifications of each VA FileMan field are documented in the file’s data dictionary.
FILE	A set of related records (or entries) treated as a unit.
FILEGRAMS	A VA FileMan feature that stores file information in a sequential format in preparation for archiving or for sending it to a corresponding database in another computing location.
GLOBAL	In M, global may refer to a variable stored on disk (“global variable”) or the array to which the global variable may belong (“global array”).
HELP FRAMES	Online screens of documentation made possible by the Kernel’s Help Processor.
IMPLICITING	Term used by M/SQL operating system for global translation.
INIT	A step in the installation process that builds VA FileMan files from a set of routines (the “init routines”). Shortened form for “initialization.”
INDEX	A part of the data global whose subscripts are one or more fields from a single record in the file, along with the internal entry number (or numbers) that locate the record. An ordered list of all or a subset of the records in the file used to facilitate lookup and sorting.
INDEX FILE	This file was introduced with VA FileMan 22.0. Contains the information that describes an index on a file. Old-style index information is stored descendent from the description of the indexed field in the data dictionary. The INDEX file allows the creation of more complex indexes.
JOURNALING	The capturing of changes to files in order to facilitate the restoring of files from a known prior state.

Term	Description
KERNEL	A set of VistA software utilities that function as an intermediary between the host operating system and VistA application packages (e.g., Laboratory, Pharmacy, IFCAP, etc.). Kernel provides a standard and consistent user and programmer interface between application packages and the underlying M implementation.
KEY	A group of one or more fields that together uniquely identifies a record in a file. Each key field *must* have a value, and fields that make up a key *must* in combination be unique for all records in the file. VA FileMan enforces key integrity.
KEY VARIABLE	See "Package-Wide Variable" below.
LAYGO ACCESS	A user's authorization to create a new entry when editing a computer file. **L**earn **A**s **Y**ou **GO**: the ability to create new entries.
MAILMAN	An electronic mail system (e-mail) that allows you to send messages to and receive them from other users via the computer. It is part of VistA.
MAPPING	See "Routine Mapping.
OPERATING SYSTEM	A basic program that runs on the computer, controls the peripherals, allocates computing time to each user, and communicates with terminals. Some M implementations take over the functions of an operating system completely; others run on top of another host operating system.
PACKAGE	The set of programs, files, documentation, online help, and installation procedures required for a given software application package identified by a unique namespace. Elements include routines, files, and file entries from the OPTION, KEY, HELP FRAME, BULLETIN, FUNCTION, SORT TEMPLATE, PRINT TEMPLATE, INPUT TEMPLATE, FORM, and BLOCK files. Packages are transported using VA FileMan's DIFROM routine, which creates initialization (init) routines to bundle the files and entries for export.
PACKAGE-WIDE VARIABLE	For VistA, a variable that, for a particular application package, has a standard and documented meaning. Some package-wide variables may need to be defined at all times during package use. Also called Key Variable.
POINTER RELATIONSHIPS	In VA FileMan, links between files that are created by use of the POINTER TO A FILE or VARIABLE-POINTER DATA TYPEs.
PROGRAMMER ACCESS	The ability to utilize VA FileMan features that are reserved for application developers. Referred to as "having the at-sign (**@**)", because **@** is the **DUZ(0)** value that grants programmer access.
PROGRAMMER MODE	Entry into VA FileMan directly from the M prompt instead of from Kernel's menu system (e.g., by entering **D P^DI** at the M prompt).

Term	Description
REPLICATION (OF GLOBALS)	The practice of keeping and maintaining identical copies of the same global in different physical locations.
ROUTINE	A program or a sequence of instructions called by a program that may have some general or frequent use. M routines are groups of program lines that are saved, loaded, and called as a single unit via a specific name.
ROUTINE MAPPING	The placement of routines into main memory. Frequently used routines are mapped to reduce disk access and thereby increase efficiency.
SAC EXEMPTION	An exception specifically granted by the Standards and Conventions Committee of the Programming Standards and Conventions requirements.
SCREENMAN	A VA FileMan screen-oriented utility that supports creation, alteration, and presentation of screens for data editing and data display.
SDP SPACE	Sequential Disk Processor space is an area on disk set aside for temporary storage of data during copying of the data. SDP is implemented by some M systems.
SPACEBAR RETURN or SPACEBAR ENTER	The use of the key combination **<Spacebar><Return>** or **<Spacebar><Enter>** at a prompt. VA FileMan retrieves the user's last response to that prompt.
STANDALONE	Referring to VA FileMan, the use of VA FileMan without the complete Kernel. The rest of Kernel adds functionality; however, VA FileMan can be used alone.
TEMPLATE	A means of storing report formats, data entry formats, and sorted entry sequences. A template is a permanent place to store selected field specifications for use at a later time.
TRANSLATION (OF GLOBALS)	The pointing to a physical disk storage location in another UCI for location of a global. Allows the same globals to be accessed from multiple UCIs.
VISTA	The Veterans Health Information Systems and Technology Architecture, within the Department of Veterans Affairs, is the component of the Veterans Health Administration that develops software and installs, maintains, and updates compatible computer systems in VA medical facilities. (Previously known as the Decentralized Hospital Computer Program [DHCP].)

Index

E

F

G

H

I

K

L

M

N

O

P

Q

R

S

T

U

V

W

X