VA FileMan 22.0

Technical Manual



February 2016

Department of Veterans Affairs (VA)

Office of Information and Technology (OI&T)

Product Development (PD)

Revision History

Table . Document revision history

| Date | Revision | Description | Author |
| --- | --- | --- | --- |
| 02/17/2016 | 2.0 | Tech Edit:   * Converted document from Word .doc to Word **.docx**, since the VDL now allows the **.docx** format. * Reformatted document to follow latest documentation standards, styles, and formatting rules. * Formatted document for online presentation vs. print presentation (i.e., for double-sided printing). These changes include: * Revised section page setup. * Removed section headers. * Revised document footers. * Removed blank pages between sections. * Revised all heading style formatting. * Updated the “Revision History” section. * Added links to all internal section references. * New baseline document. * Updated document for Patch DI\*22\*167 (Remedy tickets: 447336 and 445925). * Replaced references from “*VA FileMan Getting Started Manual* to “*VA FileMan User Manual*,” since the next VA FileMan 22.*n* software version is creating a new “*VA FileMan Getting Started Manual*.” | Tech Writer: T. B. |
| 03/--/1999 | 1.0 | Initial document. Version 22.0 release. | VA FileMan 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.

Contents

[Revision History ii](#_Toc443482677)

[Figures and Tables v](#_Toc443482678)

[Orientation vi](#_Toc443482679)

[1 Introduction 1](#_Toc443482680)

[2 Implementation and Maintenance 1](#_Toc443482681)

[3 Files 3](#_Toc443482682)

[3.1 Pointer Map 9](#_Toc443482683)

[4 Routines and Callable Routines/Entry Points/Application Programming Interfaces (APIs) 14](#_Toc443482684)

[4.1 Direct Mode Utilities 32](#_Toc443482685)

[4.2 ScreenMan-Specific Utilities 33](#_Toc443482686)

[4.3 Mapping Routines 33](#_Toc443482687)

[5 Exported Options 35](#_Toc443482688)

[5.1 Standalone VA FileMan 35](#_Toc443482689)

[5.2 VA FileMan with Kernel 38](#_Toc443482690)

[6 Cross-References 44](#_Toc443482691)

[6.1 PRINT TEMPLATE File (#.4) 44](#_Toc443482692)

[6.2 SORT TEMPLATE File (#.401) 44](#_Toc443482693)

[6.3 INPUT TEMPLATE File (#.402) 45](#_Toc443482694)

[6.4 FORM File (#.403) 45](#_Toc443482695)

[6.5 BLOCK File (#.404) 46](#_Toc443482696)

[6.6 IMPORT TEMPLATE File (#.46) 46](#_Toc443482697)

[6.7 FILE of Files (#1) 46](#_Toc443482698)

[6.8 SQLI\_TABLE\_ELEMENT File (#1.5216) 47](#_Toc443482699)

[6.9 SQLI\_COLUMN File (#1.5217) 47](#_Toc443482700)

[6.10 SQLI\_PRIMARY\_KEY File (#1.5218) 47](#_Toc443482701)

[7 Archiving and Purging 48](#_Toc443482702)

[7.1 Archiving 48](#_Toc443482703)

[7.2 Purging 48](#_Toc443482704)

[8 External Relationships 49](#_Toc443482705)

[8.1 DBA Approvals and Database Integration Control Registrations (ICRs) 50](#_Toc443482706)

[8.1.1 ICRs—Current List for VA FileMan as Custodian 50](#_Toc443482707)

[8.1.2 ICRs—Detailed Information 50](#_Toc443482708)

[8.1.3 ICRs—Current List for VA FileMan as Subscriber 51](#_Toc443482709)

[9 Internal Relationships 52](#_Toc443482710)

[10 Package-Wide Variables 52](#_Toc443482711)

[10.1 Standards and Conventions (SAC) Exemptions 53](#_Toc443482712)

[10.1.1 STANDARD SECTION: 4B–Package-wide variables 53](#_Toc443482713)

[10.1.2 STANDARD SECTION: 6D–FM compatibility 53](#_Toc443482714)

[11 Globals 54](#_Toc443482715)

[11.1 Global Journaling, Translation, and Replication 56](#_Toc443482716)

[11.1.1 Journaling 56](#_Toc443482717)

[11.1.2 Translation 56](#_Toc443482718)

[11.1.3 Replication 57](#_Toc443482719)

[12 Security 58](#_Toc443482720)

[12.1 Security Management 58](#_Toc443482721)

[12.2 Mail Groups and Alerts 58](#_Toc443482722)

[12.3 Remote Systems 58](#_Toc443482723)

[12.4 Interfacing 59](#_Toc443482724)

[12.5 Electronic Signatures 59](#_Toc443482725)

[12.6 Security Keys 59](#_Toc443482726)

[12.7 File Security 59](#_Toc443482727)

[12.8 References 60](#_Toc443482728)

[12.9 Official Policies 60](#_Toc443482729)

[13 Troubleshooting 60](#_Toc443482730)

[13.1 How to Obtain Technical Information Online 60](#_Toc443482731)

[13.2 Help at Prompts 60](#_Toc443482732)

[Glossary 61](#_Toc443482733)

[Index 65](#_Toc443482734)

Figures and Tables

Figures

[Figure 1. Type of M system prompt ix](#_Toc443482735)

[Figure 2. VA FileMan Pointer Map 9](#_Toc443482736)

[Figure 3. VA FileMan exported options diagram 38](#_Toc443482737)

Tables

[Table 1. Document revision history ii](#_Toc443482738)

[Table 2. Documentation symbol descriptions viii](#_Toc443482739)

[Table 3. VA FileMan routine variables and default values x](#_Toc443482740)

[Table 4. VA FileMan routine global references xi](#_Toc443482741)

[Table 5. VA FileMan File List 3](#_Toc443482742)

[Table 6. VA FileMan Routine and Callable Entry Points 14](#_Toc443482743)

[Table 7. PRINT TEMPLATE File (#.4)—Cross-references 44](#_Toc443482744)

[Table 8. SORT TEMPLATE File (#.401)—Cross-references 44](#_Toc443482745)

[Table 9. NPUT TEMPLATE File (#.402)—Cross-references 45](#_Toc443482746)

[Table 10. FORM File (#.403)—Cross-references 45](#_Toc443482747)

[Table 11. BLOCK File (#.404)—Cross-references 46](#_Toc443482748)

[Table 12. IMPORT TEMPLATE File (#.46)—Cross-references 46](#_Toc443482749)

[Table 13. FILE of Files (#1)—Cross-references 46](#_Toc443482750)

[Table 14. SQLI\_TABLE\_ELEMENT File (#1.5216)—Cross-references 47](#_Toc443482751)

[Table 15. SQLI\_COLUMN File (#1.5217)—Cross-references 47](#_Toc443482752)

[Table 16. SQLI\_PRIMARY\_KEY File (#1.5218)—Cross-references 47](#_Toc443482753)

[Table 17. Package-wide Variables 52](#_Toc443482754)

[Table 18. Package-wide Variables—DISY (special Meaning) 52](#_Toc443482755)

[Table 19. List of variables VA FileMan is exempted from KILLing 53](#_Toc443482756)

[Table 20. VA FileMan Security Keys 59](#_Toc443482757)

[Table 21Table 22. Glossary 61](#_Toc443482758)

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, as a set of interactive or “silent” routines, or as a 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. 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 and Callable Routines/Entry Points/Application Programming Interfaces (APIs)
* Exported 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.

* Information Resource Management (IRM)—System administrators at Department of Veterans Affairs (VA) sites who are responsible for computer management and system security on the VistA M Servers.
* Product Development (PD)—VistA legacy development teams.
* Product Support (PS).

Disclaimers

Software Disclaimer

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.

 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 22.0; 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 (OI&T) 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 2 gives a description of each of these symbols:

Table . Documentation symbol descriptions

|  |  |
| --- | --- |
| Symbol | Description |
| Note | **NOTE / REF:** Used to inform the reader of general information including references to additional reading material. |
| Caution | **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 file (#.7). The *non*-standard code used (if any) depends on the answer to the prompt:

Figure . 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.

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 or DD. (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 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. All 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.

The following local variables are of special importance in the VA FileMan routines:

Table . 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 has to 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.  Note **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 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 following globals:

Table . VA FileMan routine global references

| Global | Description |
| --- | --- |
| ^DD | All attribute dictionaries. |
| ^DDA | Data dictionary audit trail. |
| ^DI | Data types. |
| ^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 and Alternate Editors. |
| ^DISV | Most recent lookup value in any file or subfile (by DUZ). |
| ^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.

# 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.

# Implementation and Maintenance

VA FileMan is initialized with the DINIT routine, followed by an install using the Kernel Distribution and Installation system (KIDS) as directed in the *VA FileMan Installation Guide*. DINIT only needs to be run once; it is nondestructive to a system if run again. Standalone VA FileMan sites do *not* perform the KIDS install.

VA FileMan routines and globals occupy approximately 3.5 MB of memory. The size of the globals, particularly those that store file 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 file (#3.5) entry that uses a sequential disk processor (SDP) device type.

 **REF:** The *Kernel 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 KILLed 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*.

# Files

This section lists all the VA FileMan files, file numbers, global locations, and a brief description of each. Data exported with standalone VA FileMan is also indicated.

VA FileMan uses files numbered between 0 and 2..

VA FileMan files should *not* be altered, per VHA Directive 10-93-142.

Table . VA FileMan File List

| 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 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.  Note **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 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 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. |
| 1 | File | ^DIC( | The FILE file stores the name, number, global name or location, package name, security access, and 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. |
| 1.521 | 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. |
| 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 file [#1.5216], 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 file [#1.5216]) 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 used to relate primary keys to foreign keys unambiguously.  Note **REF:** For information on table elements, see the SQLI\_TABLE\_ELEMENT file (#1.5216). |
| 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: includes name and file number (see the SQLI\_TABLE\_ELEMENT file [#1.5216]). 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. |
| 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.  Note **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.  Note **REF:** For information on table elements, see the SQLI\_TABLE\_ELEMENT file (#1.5216). |
| 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).  Note **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 file [#1.5216]), 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.  Note **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.  Note **REF:** For more information, see the SQLI\_ERROR\_TEXT and DIALOG files. |

The DINIT routines install the files listed in Table 5.

Another set of init routines (DIPKINIT), called by DINIT during installation, is sent with each release of VA FileMan. These routines install the PACKAGE file (#9.4), if you are running a version of Kernel prior to Version 8.0 or if you are running standalone VA FileMan without Kernel. The PACKAGE file (#9.4) is necessary to build inits using DIFROM. A single entry for DIPK is created in the PACKAGE file (#9.4) by the DIPKINITs.

 **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 22.0 does *not* transport the new Key and Index structures and should *not* be used to transport any file making use of these new features.

## 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 . VA FileMan Pointer Map

File/Package: Date: SEP 8,1994

FILE (#) POINTER (#) FILE

POINTER FIELD TYPE POINTER FIELD FILE POINTED TO

--------------------------------------------------------------------------------

L=Laygo S=File not in set N=Normal Ref. C=Xref.

\*=Truncated m=Multiple v=Variable Pointer

---------------------

KEY (#.31) | |

UNIQUENESS INDEX ..... (N )-> | .11 INDEX |

---------------------

---------------------

ARCHIVAL ACTIVITY (#1.11) | |

PRINT TEMPLATE ....... (N )-> | .4 PRINT TEMPLA\* |

FILEGRAM HISTORY (#1.12) | |

FILEGRAM ............. (N )-> | FILE |-> FILE

| DESTINATION FI\* |-> FILE

---------------------

---------------------

ARCHIVAL ACTIVITY (#1.11) | |

SEARCH TEMPLATE ...... (N L)-> | .401 SORT TEMPL\* |

| FILE |-> FILE

---------------------

---------------------

KERNEL SITE PARAMETE (#4.3) | |

USER CHARACTERISTICS T\* (N S )-> | .402 INPUT TEMP\* |

KERNEL SYSTEM PARAME (#8989.3) | |

USER CHARACTERISTICS T\* (N S )-> | FILE |-> FILE

---------------------

---------------------

FORM (#.4031) | |

PAGE:HEADER BLOCK .... (N L)-> | .404 BLOCK |

PAGE:BLOCK:BLOCK NAME (N C L)-> | |

---------------------

---------------------

PRINT TEMPLATE (#.4) | |

EXPORT FORMAT ........ (N )-> | .44 FOREIGN FOR\* |

---------------------

---------------------

| .46 IMPORT TEMP\* |

| PRIMARY FILE |-> FILE

| CREATOR |-> NEW PERSON

| IMPORT:FILE\* |-> FILE

---------------------

---------------------

| .6 DD AUDIT |

| USER |-> NEW PERSON

---------------------

---------------------

SORT TEMPLATE (#.4014) | |

SORT FIEL:DATA TYPE F\* (N )-> | .81 DATA TYPE |

PRINT TEMPLATE (#.42) | |

EXPORT FIELD:DATA TYPE (N )-> | |

---------------------

---------------------

SQLI\_ERROR\_LOG (#1.52192) | .84 DIALOG |

FILEMAN\_ERROR ……. (N C)-> | PACKAGE |-> PACKAGE

---------------------

---------------------

DIALOG (#.847) | |

TRANSLATION:LANGUAGE . (N C )-> | .85 LANGUAGE |

NEW PERSON (#200) | |

LANGUAGE ............. (N S )-> | |

KERNEL SITE PARAMETE (#8989.3) | |

DEFAULT LANGUAGE ..... (N S )-> | |

---------------------

File/Package(Cont): Date: SEP 8,1994

FILE (#) POINTER (#) FILE

POINTER FIELD TYPE POINTER FIELD FILE POINTED TO

--------------------------------------------------------------------------------

L=Laygo S=File not in set N=Normal Ref. C=Xref.

\*=Truncated m=Multiple v=Variable Pointer

---------------------

VARIABLE-POINTER (#.12) | |

. . . . . . . . . . . (N S)-> | |

PRINT TEMPLATE (#.4) | 1 FILE |

FILE ................. (N )-> | |

DESTINATION FILE ... . (N )-> | |

SORT TEMPLATE (#.401) | DEVELOPER |-> NEW PERSON

FILE ................. (N )-> | |

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 )-> | |

NEW PERSON (#200.032) | |

ACCESSIBLE FILE ...... (N S C )-> | |

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 )-> | |

---------------------

---------------------

| 1.1 AUDIT |

| USER |-> NEW PERSON

| MENU OPTION US\* |-> OPTION

| v PROTOCOL or OP\* |-> OPTION

| |-> PROTOCOL

---------------------

File/Package(Cont): Date: SEP 8,1994

FILE (#) POINTER (#) FILE

POINTER FIELD TYPE POINTER FIELD FILE POINTED TO

--------------------------------------------------------------------------------

L=Laygo S=File not in set N=Normal Ref. C=Xref.

\*=Truncated m=Multiple v=Variable Pointer

---------------------

| 1.11 ARCHIVAL A\* |

| FILE |-> FILE

| ARCHIVER |-> NEW PERSON

| SELECTOR |-> NEW PERSON

| PURGER |-> NEW PERSON

| USER PERFORMIN\* |-> NEW PERSON

| DESTINATION FI\* |-> FILE

---------------------

---------------------

| 1.12 FILEGRAM H\* |

| FILE |-> FILE

| MESSAGE |-> 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 )-> | 1.5211 SQLI\_DAT\* |

SQLI\_KEY\_FORMAT (#1.5213) | |

KF\_DATA\_TYPE ......... (N C )-> | D\_OUTPUT\_FORMAT |->SQLI\_OUTPUT\_FO\*

SQLI\_OUTPUT\_FORMAT (#1.5214) | |

OF\_DATA\_TYPE ......... (N )-> | |

---------------------

---------------------

SQLI\_TABLE\_ELEMENT (#1.5216) | |

E\_DOMAIN ............. (N C )-> | 1.5212 SQLI\_DOM\* |

| DM\_DATA\_TYPE |-> SQLI\_DATA\_TYPE

| DM\_TABLE |-> SQLI\_TABLE

| DM\_OUTPUT\_FORM\* |->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 )-> | 1.5214 SQLI\_OUT\* |

SQLI\_DOMAIN (#1.5212) | |

DM\_OUTPUT\_FORMAT ..... (N )-> | OF\_DATA\_TYPE |-> SQLI\_DATA\_TYPE

SQLI\_COLUMN (#1.5217) | |

C\_OUTPUT\_FORMAT ...... (N C )-> | |

---------------------

---------------------

SQLI\_DOMAIN (#1.5212) | |

DM\_TABLE ............. (N C )-> | 1.5215 SQLI\_TAB\* |

SQLI\_TABLE (#1.5215) | |

T\_MASTER\_TABLE ....... (N C )-> | T\_SCHEMA |-> SQLI\_SCHEMA

SQLI\_TABLE\_ELEMENT (#1.5216) | |

E\_TABLE .............. (N C )-> | T\_MASTER\_TABLE |-> SQLI\_TABLE

---------------------

---------------------

SQLI\_COLUMN (#1.5217) | |

C\_TABLE\_ELEMENT ...... (N C )-> | 1.5216 SQLI\_TAB\* |

SQLI\_PRIMARY\_KEY (#1.5218) | |

P\_TBL\_ELEMENT ........ (N C )-> | E\_DOMAIN |-> SQLI\_DOMAIN

SQLI\_FOREIGN\_KEY (#1.5219) | |

F\_TBL\_ELEMENT ........ (N C )-> | E\_TABLE |-> SQLI\_TABLE

---------------------

---------------------

SQLI\_COLUMN (#1.5217) | |

C\_PARENT ............. (N C )-> | 1.5217 SQLI\_COL\* |

SQLI\_PRIMARY\_KEY (#1.5218) | |

P\_COLUMN ............. (N C )-> | C\_TABLE\_ELEMENT |->SQLI\_TABLE\_ELE\*

SQLI\_FOREIGN\_KEY (#1.5219) | |

F\_CLM\_ELEMENT ........ (N )-> | 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\*

---------------------

# Routines and Callable Routines/Entry Points/Application Programming Interfaces (APIs)

This section lists and briefly describes the VA FileMan routines (see Table 6). VA FileMan routines should *not* be altered, per Veterans Health Administration (VHA) Directive 10-93-142.

The callable entry points/Application Programming Interfaces (APIs) for those VA FileMan routines that can be invoked from other applications are shown in the “Callable Entry Point” column in Table 6.

 **REF:** The callable routines/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*:

Callable routines/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 4.1and 4.2, and are also described in the *VA FileMan Developer’s Guide*.

 **REF:** Routine mapping is described in Section 4.3.

Table . VA FileMan Routine 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  DOCLIST^  DDBR | Routines responsible for displaying ASCII text on a terminal screen, for viewing only. |
| DDBR0  DDBR1  DDBR2  DDBR3  DDBR4  DDBRAHT  DDBRAHTE  DDBRAHTJ  DDBRAHTR  DDBRAP  DDBRGE  DDBRP  DDBRS |  |  |
| DDBRT | $$TEST^  DDBRT |  |
| DDBRU  DDBRU2  DDBRWB |  |  |
| DDBRZIS | CLOSE^  DDBRZIS  OPEN^  DDBRZIS  POST^  DDBRZIS |  |
| DDFIX |  | Routine that checks nodes in the data dictionary and the FILE file (#1). |
| DDGF  DDGF0  DDGF1  DDGF2  DDGF3  DDGF4  DDGFADL  DDGFAPC  DDGFASUB  DDGFBK  DDGFBSEL  DDGFEL  DDGFFLD  DDGFFLDA  DDGFFM  DDGFH  DDGFHBK  DDGFLOAD  DDGFORD  DDGFPG  DDGFSV  DDGFU  DDGFUPDB  DDGFUPDP |  | Routines used to create and edit ScreenMan forms. |
| DDGLIB0  DDGLIBH  DDGLIBW  DDGLIBW1 |  | Routines that manage the screen for VA FileMan’s screen-oriented utilities. |
| DDIOL | EN^DDIOL | Routine that any of the following:   * Writes text to the screen. * Writes text in ScreenMan’s Command Area. * Loads text into an array, depending on the environment in which it is called. |
| DDMAP  DDMAP1  DDMAP2 |  | Routines that generate a graphic display of the pointer relationships among a specified group of package files to an output device. |
| DDMP  DDMP1  DDMP2  DDMPSM  DDMPSM1  DDMPU | FILE^DDMP | Routines used by the Import Tool. |
| DDR  DDR0  DDR1  DDR2  DDR3  DDR4 |  | Routines that contain the RPCs for the VA FileMan Delphi components. |
| DDS  DDS0  DDS01  DDS02  DDS1  DDS10  DDS11  DDS2  DDS3  DDS4  DDS41  DDS5  DDS6  DDS7  DDSBOX  DDSCAP  DDSCLONE  DDSCLONF  DDSCOM  DDSCOMP  DDSDBLK  DDSDEL  DDSDFRM  DDSFO  DDSIT  DDSLIB  DDSM  DDSM1  DDSMSG  DDSOPT  DDSPRNT  DDSPRNT1  DDSPRNT2  DDSPTR  DDSR  DDSR1  DDSRSEL  DDSRUN  DDSSTK  DDSU | DDS | Routines used to compile and run forms for data viewing and editing—ScreenMan. |
| DDSUTL | MSG^  DDSUTL  REFRESH^  DDSUTL  REQ^  DDSUTL  UNED^  DDSUTL |  |
| DDSVAL | $$GET^  DDSVAL  PUT^  DDSVAL |  |
| DDSVALF | $$GET^  DDSVALF  PUT^  DDSVALF |  |
| DDSVALM  DDSWP  DDSZ  DDSZ1  DDSZ2  DDSZ3 |  |  |
| DDU  DDUCHK  DDUCHK1  DDUCHK2  DDUCHK3  DDUCHK4  DDUCHK5 |  | Routines responsible for running the data dictionary checking utility. |
| DDW  DDW1  DDW2  DDW3  DDW4  DDW5  DDW6  DDW7  DDW8  DDW9  DDWC  DDWC1  DDWF  DDWG  DDWH  DDWK  DDWT1 |  | Routines responsible for full screen text editing. |
| DDXP  DDXP1  DDXP2  DDXP3  DDXP31  DDXP32  DDXP33  DDXP4  DDXP41  DDXP5  DDXPLIB |  | Routines responsible for the data export to a Foreign Format tool. |
| DI |  | Routine for direct entry into VA FileMan. |
| DIA  DIA1  DIA2  DIA3 |  | Routines responsible for gathering fields to be edited. |
| DIAC | DIAC | Routine that determines file access. |
| DIALOG | BLD^  DIALOG  $$EZBLD^  DIALOG | Routines to build VA FileMan dialogues and their functions. |
| DIALOGU |  |  |
| DIAR  DIARA  DIARB  DIARCALC  DIARR  DIARR1  DIARR2  DIARR3  DIARR4  DIARR5  DIARR6  DIARU  DIARX |  | Routines responsible for VA FileMan archiving. |
| DIAU |  | Routine used for auditing. |
| DIAX  DIAXD  DIAXERR  DIAXF  DIAXM  DIAXM1  DIAXM2  DIAXM3  DIAXMS  DIAXP  DIAXT  DIAXU | EN^DIAXU | Routines responsible for extracting data to a VA FileMan file. |
| 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 |  |  |
| DICA  DICA1  DICA2  DICA3 |  | Routines responsible for DBS Updater functions. |
| DICATT  DICATT0  DICATT1  DICATT2  DICATT22  DICATT3  DICATT4  DICATT5  DICATT6 |  | Routines responsible for the Modify File Attributes option. |
| DICATTA |  | Routine responsible for data dictionary audits. |
| DICATTD  DICATTD0  DICATTD1  DICATTD2  DICATTD3  DICATTD4  DICATTD5  DICATTD6  DICATTD7  DICATTD9  DICATTDD  DICATTDE  DICATTDK  DICATTDM |  | 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. |
| DICL  DICL1  DICL10  DICL2  DICL3  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  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  DICU2  DICUF  DICUIX  DICUIX1  DICUIX2 |  | Routines containing utilities used during lookups. |
| 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. |
| DIDH |  | Headers for the data dictionary listings. |
| DIDH1 |  |  |
| DIDX |  | Brief data dictionary listing. |
| 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. |
| DIE | DIE  CHK^DIE  FILE^DIE  HELP^DIE  $$KEYVAL^  DIE  UPDATE^  DIE  VAL^DIE  VALS^DIE  WP^DIE | Routines responsible for the Enter or Edit File Entries option and for DBS filing and help retrieval functions. |
| 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. |
| DIEV  DIEV1  DIEVK  DIEVK1  DIEVS |  | Routines responsible for data validation functions. |
| DIEZ  DIEZ0  DIEZ1  DIEZ2  DIEZ3  DIEZ4 | DIEZ  EN^DIEZ | Routines that compile INPUT templates. |
| 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 |  |  |
| DIFROM  DIFROM0  DIFROM1  DIFROM11  DIFROM12  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 | DIFROM | Routines responsible for generating init packages for export and supporting Kernel’s KIDS functions. |
| DIG |  | Routine responsible for the Scattergram option on the Statistics submenu. |
| DIH |  | Routine responsible for the Histogram option on the Statistics submenu. |
| DII  DII1 |  | Routines responsible for the main menu in standalone VA FileMan and for the Inquire to File Entries option. |
| DIIS  DIISS |  | Routines responsible for device selection for standalone VA FileMan. Stored in the Manager account as %ZIS 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  DIKCDD  DIKCFORM  DIKCP  DIKCP1  DIKCP2  DIKCP3  DIKCU  DIKCU1  DIKCU2  DIKCUTL  DIKCUTL1  DIKCUTL2  DIKCUTL3 |  | Routines responsible for defining, printing, and executing the logic for New-Style indices. |
| DIKK  DIKK1  DIKK2  DIKKDD  DIKKFORM  DIKKP  DIKKUTL  DIKKUTL1  DIKKUTL2  DIKKUTL3  DIKKUTL4 |  | Routines responsible for defining, printing, and verifying the integrity of Keys. |
| 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  DIM3  DIM4 | DIM | Routines responsible for the M syntax checker. |
| DINIT |  | Routines that initialize VA FileMan. |
| DINIT\* |  | Numerous routines starting with “DINIT” are used in the initialization process. |
| DINTEG |  | Routines used for integrity checking of VA FileMan routines. |
| DINTEG\* |  | Several routines starting with “DINTEG” are used during integrity checking. |
| DINVDTM  DINV1DTM  DINVMSM  DINVONT  DINVVXD  DINV1VXD |  | Routines that DINZMGR saves as %ZOSV and, for some operating systems, %ZOSV1, during a standalone VA FileMan installation. |
| DINZMGR  DINZMGR1  DINZDTM  DINZMSM  DINZONT  DINZVXD |  | Routines that rename routines in the Manager account and set up ^%ZOSF nodes for standalone VA FileMan. |
| 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  DIP4  DIP5 | 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. |
| DIPKINIT |  | Routines that install a new data dictionary for the PACKAGE file (#9.4). The data dictionary brings no data. |
| DIPKI\* |  | There are numerous routines starting with “DIPKI” used in the PACKAGE file (#9.4) initialization. (During VA FileMan Installation, DINIT automatically installs the PACKAGE file at sites *not* running Kernel 8.0 or later.) |
| 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  DIR0W  DIR1  DIR2  DIR3  DIRQ | DIR | Routines responsible for the standard reader used in VA FileMan. |
| 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. |
| DITM  DITM1  DITM2  DITMGM1  DITMGM2  DITMGM2A  DITMGM2B  DITMGM2C  DITMGMRG  DITMGMRI  DITMU1  DITMU2  DITMU3  DITMU4 |  | Routines used to compare/merge two records located within a single file. |
| 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 |  |  |
| 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. |
| 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  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*.

## Direct Mode Utilities

In addition to the callable entry points shown in Table 6, 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*.

## 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 “Prog. Mode Utilities” section in the “ScreenMan Forms” section in the “ScreenMan” section in the *VA FileMan Developer’s Guide*.

## Mapping Routines

Routine mapping is at the discretion of the systems manager. The RTHIST routines provide a method for each site to determine the extent to which certain routines are used.

The following list is provided as a *recommendation only*:

* In the MGR account:
* %DT, %DTC, %RCR
* In the VAH account:
* DDBR\*
* DDIOL
* DDS, DDS0\*, DDS1\*, DDSCOM, DDSR\*, DDSU
* DDW4, DDW5, DDW6, DDWT1
* DIAC, DIALOG\*
* DIC, DIC0, DIC1\*, DIC2, DIC3
* DICA, DICA1, DICA2, DICA3
* DICD
* DICF\*
* DICL\*
* DICM\*
* DICN\*
* DICQ\*
* DICR, DICRW
* DICU\*
* DIE, DIE0, DIE1, DIE2, DIE3, DIE9, DIE17
* DIED
* DIEF\*
* DIEH\*
* DIEQ\*
* DIEV\*
* DIK, DIK1, DIKC, DIKC1, DIKC2, DIKCU, DIKCU1, DIKCU2
* DIL\*
* DIO\*
* DIP, DIP0, DIP1\*, DIP2, DIP21, DIP22, DIP3\*, DIP4, DIP5
* DIQG, DIQGQ, DIQGU\*
* DIW\*
* DIR, DIR0, DIR01, DIR1, DIR2, DIR3, DIRQ

 CAUTION: DIEZ\*, DIFRO\*, DIKZ\*, DIOZ\*, DIPZ\* *must not* be mapped.

# Exported Options

## Standalone VA FileMan

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

The following options are provided to standalone VA FileMan users:

* 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
* Data Dictionaries Being Audited
* 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
* 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 Dictionary Utilities
* List File Attributes
* Map Pointer Relations
* Check/Fix DD Structure
* Transfer Entries
* Transfer File Entries
* Compare/Merge File Entries

## VA FileMan with Kernel

VA FileMan exports the options listed in Figure 3. They are installed if Kernel exists on your system. The top-level VA FileMan menu option, DIUSER, 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 . VA FileMan exported options diagram

VA FileMan (DIUSER)

\*\*ENTRY ACTION:

W !!?10,”VA FileMan Version “\_^DD(“VERSION”)

|

|

--------------------------------------------------------- Enter or Edit

File Entries

[DIEDIT]

\*\*ENTRY ACTION:

D ^DIB

--------------------------------------------------------- Print File

Entries

[DIPRINT]

\*\*ENTRY ACTION:

D ^DIP

--------------------------------------------------------- Search File

Entries

[DISEARCH]

\*\*ENTRY ACTION:

D ^DIS

--------------------------------------------------------- Modify File

Attributes

[DIMODIFY]

\*\*ENTRY ACTION:

D ^DICATT

--------------------------------------------------------- Inquire to File

Entries

[DIINQUIRE]

\*\*ENTRY ACTION:

D INQ^DII

----- Utility ------------------------------------------- Verify Fields

Functions [DIVERIFY]

[DIUTILITY] \*\*ENTRY ACTION:

| S DI=1 G EN^DIU

|

|---------------------------------------------- Cross-Reference

| A Field [DIXREF]

| \*\*ENTRY ACTION:

| S DI=2 G EN^DIU

|

|---------------------------------------------- Identifier

| [DIIDENT]

| \*\*ENTRY ACTION:

| S DI=3 G EN^DIU

|

|---------------------------------------------- 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/Requir

| ed 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 ----------------------------------- List File

Utilities [DI Attributes

DDU] [DILIST]

| \*\*ENTRY ACTION:

| D ^DID

|

|---------------------------------------------- Map Pointer

| Relations [DI

| DDMAP]

|

|---------------------------------------------- Check/Fix DD

Structure [DI

DDUCHK]

--------------------------------------------------------- Transfer Entries

[DITRANSFER]

\*\*ENTRY ACTION:

D ^DIT

----- Other Options ----------- Filegrams [DIFG] -------- Create/Edit

[DIOTHER] \*\*LOCKED: Filegram

| XUFILEGRAM\*\* Template [DIFG

| | CREATE]

| | \*\*LOCKED:

| | XUFILEGRAM\*\*

| | \*\*ENTRY ACTION:

| | S DI=1 D

| | EN^DIFGO

| |

| |-------------------- Display Filegram

| | Template [DIFG

| | DISPLAY]

| | \*\*LOCKED:

| | XUFILEGRAM\*\*

| | \*\*ENTRY ACTION:

| | S DI=2 D

| | EN^DIFGO

| |

| |-------------------- Generate

| | Filegram [DIFG

| | GENERATE]

| | \*\*LOCKED:

| | XUFILEGRAM\*\*

| | \*\*ENTRY ACTION:

| | S DI=3 D

| | EN^DIFGO

| |

| |-------------------- View Filegram

| | [DIFG VIEW]

| | \*\*ENTRY ACTION:

| | S DI=4 D

| | EN^DIFGO

| |

| |-------------------- Specifiers [DIFG

| | SPECIFIERS]

| | \*\*LOCKED:

| | XUFILEGRAM\*\*

| | \*\*ENTRY ACTION:

| | S DI=5 D

| | EN^DIFGO

| |

| |-------------------- Install/Verify

| Filegram [DIFG

| INSTALL]

| \*\*LOCKED:

| XUFILEGRAM\*\*

| \*\*ENTRY ACTION:

| S DI=6 D

| EN^DIFGO

|

|

|-------------------- Audit Menu -------------- Fields Being

| [DIAUDIT] Audited

| \*\*LOCKED: [DIAUDITED

| XUAUDITING\*\* FIELDS]

| |

| |-------------------- Data

| | Dictionaries

| | Being Audited

| | [DIAUDIT DD]

| |

| |-------------------- Purge Data

| | Audits [DIAUDIT

| | PURGE DATA]

| |

| |-------------------- Purge DD Audits

| | [DIAUDIT PURGE

| | DD]

| |

| |-------------------- Turn Data Audit

| On/Off [DIAUDIT

| TURN ON/OFF]

|

|

|-------------------- ScreenMan [DDS ---------- Edit/Create a

| SCREEN MENU] Form [DDS

| \*\*LOCKED: EDIT/CREATE A

| XUSCREENMAN\*\* FORM]

| |

| |-------------------- Run a Form [DDS

| | RUN A FORM]

| | \*\*ENTRY ACTION:

| | D ^DDSRUN

| |

| |-------------------- Delete a Form

| | [DDS DELETE A

| | FORM]

| |

| |-------------------- Purge Unused

| Blocks [DDS

| PURGE UNUSED

| BLOCKS]

|

|

|---------------------------------------------- Statistics

| [DISTATISTICS]

| \*\*ENTRY ACTION:

| D ^DIX

|

|-------------------- VA FileMan -------------- Data Dictionary

| Management [DI Cross-reference

| MGMT MENU] Compile/Uncompil

| \*\*LOCKED: e [DI DD

| XUMGR\*\* COMPILE]

| |

| |-------------------- Input Template

| | Compile/Uncompil

| | e [DI INPUT

| | COMPILE]

| | \*\*ENTRY ACTION:

| | D EN1^DIEZ K DNM

| |

| |-------------------- Print Template

| | Compile/Uncompil

| | e [DI PRINT

| | COMPILE]

| |

| |-------------------- Sort Template

| | Compile/Uncompil

| | e [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 ---------- Define Foreign

| Foreign Format File Format

| [DDXP EXPORT [DDXP DEFINE

| MENU] FORMAT]

| | \*\*LOCKED:

| | DDXP-DEFINE\*\*

| | \*\*ENTRY ACTION:

| | D 1^DDXP

| |

| |-------------------- Select Fields

| | for Export [DDXP

| | SELECT EXPORT

| | FIELDS]

| | \*\*ENTRY ACTION:

| | D 2^DDXP

| |

| |-------------------- Create Export

| | Template [DDXP

| | CREATE EXPORT

| | TEMPLATE]

| | \*\*ENTRY ACTION:

| | D 3^DDXP

| |

| |-------------------- Export Data

| | [DDXP EXPORT

| | DATA]

| | \*\*ENTRY ACTION:

| | D 4^DDXP

| |

| |-------------------- Print Format

| Documentation

| [DDXP FORMAT

| DOCUMENTATION]

| \*\*ENTRY ACTION:

| D 5^DDXP

|

|

|-------------------- Extract Data To --------- Select Entries

| Fileman File to Extract [DIAX

| [DIAX EXTRACT SELECT]

| MENU] \*\*LOCKED:

| \*\*LOCKED: DIEXTRACT\*\*

| DIEXTRACT\*\* \*\*ENTRY ACTION:

| \*\*EXIT ACTION: S DI=1 D EN^DIAX

| K DIAX

| |

| |-------------------- Add/Delete

| | Selected Entries

| | [DIAX

| | ADD/DELETE]

| | \*\*LOCKED:

| | DIEXTRACT\*\*

| | \*\*ENTRY ACTION:

| | S DI=2 D EN^DIAX

| |

| |-------------------- Print Selected

| | Entries [DIAX

| | PRINT]

| | \*\*LOCKED:

| | DIEXTRACT\*\*

| | \*\*ENTRY ACTION:

| | S DI=3 D EN^DIAX

| |

| |-------------------- Modify

| | Destination File

| | [DIAX MODIFY]

| | \*\*LOCKED:

| | DIEXTRACT\*\*

| | \*\*ENTRY ACTION:

| | S DI=4 D EN^DIAX

| |

| |-------------------- Create Extract

| | Template [DIAX

| | CREATE]

| | \*\*LOCKED:

| | DIEXTRACT\*\*

| | \*\*ENTRY ACTION:

| | S DI=5 D EN^DIAX

| |

| |-------------------- Update

| | Destination File

| | [DIAX UPDATE]

| | \*\*LOCKED:

| | DIEXTRACT\*\*

| | \*\*ENTRY ACTION:

| | S DI=6 D EN^DIAX

| |

| |-------------------- Cancel Extract

| | Selection [DIAX

| | CANCEL]

| | \*\*LOCKED:

| | DIEXTRACT\*\*

| | \*\*ENTRY ACTION:

| | S DI=8 D EN^DIAX

| |

| |-------------------- Purge Extracted

| | Entries [DIAX

| | PURGE]

| | \*\*LOCKED:

| | DIEXTRACT\*\*

| | \*\*ENTRY ACTION:

| | S DI=7 D EN^DIAX

| |

| |-------------------- Validate Extract

| Template [DIAX

| VALIDATE]

| \*\*LOCKED:

| DIEXTRACT\*\*

| \*\*ENTRY ACTION:

| S DI=9 D EN^DIAX

|

|---------------------------------------------- Import Data

| [DDMP IMPORT]

|

|---------------------------------------------- Browser

[DDBROWSER]

# 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.

## PRINT TEMPLATE File (#.4)

Table . PRINT TEMPLATE File (#.4)—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. |

## SORT TEMPLATE File (#.401)

Table . SORT TEMPLATE File (#.401)—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. |

## INPUT TEMPLATE File (#.402)

Table . NPUT TEMPLATE File (#.402)—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. |

## FORM File (#.403)

Table . FORM File (#.403)—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. |
| IS THIS A POP UP PAGE? (subfield of PAGE Multiple) | AC | This cross-reference ensures that no header block, next page, or previous page is associated with a pop up page. |

## BLOCK File (#.404)

Table . BLOCK File (#.404)—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. |

## IMPORT TEMPLATE File (#.46)

Table . IMPORT TEMPLATE File (#.46)—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. |

## FILE of Files (#1)

Table . FILE of Files (#1)—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 |

## SQLI\_TABLE\_ELEMENT File (#1.5216)

Table . SQLI\_TABLE\_ELEMENT File (#1.5216)—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. |

## SQLI\_COLUMN File (#1.5217)

Table . SQLI\_COLUMN File (#1.5217)—Cross-references

| Field | X-Ref ID | Description |
| --- | --- | --- |
| C\_FIELD | D | Column by VA FileMan file number, by field number. |

## SQLI\_PRIMARY\_KEY File (#1.5218)

Table . SQLI\_PRIMARY\_KEY File (#1.5218)—Cross-references

| Field | X-Ref ID | Description |
| --- | --- | --- |
| P\_SEQUENCE | C | Primary key by table, by sequence. |

# Archiving and Purging

## 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*.

## 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. These audits are started and stopped by using the options on the Auditing submenu. 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 file (#.6) (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 file (#1.11) 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 file (#1.11) when the extract process is complete. This option should be run when using the Extract Tool for archiving purposes to remove extracted data.

# External Relationships

VA FileMan is designed to function as a standalone database management system. It can accomplish most of the activities described in the *VA FileMan Developer’s Guide* and user manuals without relying on other software (except an implementation of ANSI Standard M on which to run).

When installing VA FileMan, you identify the current M operating system. An entry in the MUMPS OPERATING SYSTEM file (#.7) is selected. The information in this file allows things to be accomplished that are done differently by various implementations of M. In addition, when the DINZMGR routine is run during a standalone installation, %ZOSF global nodes are set in the Manager account. These nodes contain additional implementation-specific information that allows for expanded functionality.

 **REF:** For details of installing standalone VA FileMan, see the “Standalone VA FileMan Installation” section in the *VA FileMan Installation Guide*.

By attending to at least minimal operating system concerns, VA FileMan can be installed *without* the remaining parts of Kernel *except* for the PACKAGE file (#9.4). The PACKAGE file (#9.4) is *not* a VA FileMan file; however, it is shipped with VA FileMan, and is installed automatically during the installation of VA FileMan, if either Kernel is *not* present or if a version prior to Kernel 8.0 is present. The PACKAGE file (#9.4) *must* be present on your system to use the DIFROM routines to export software packages.

 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 22.0 does *not* transport the new Key and Index structures and should *not* be used to transport any file making use of these new features.

VA FileMan’s capability is enhanced when it is installed with Kernel and MailMan. Specifically, VA FileMan 22.0 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 file (#200)
* 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 file (#.7). The global ^DD *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.

## 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.

### 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 A**CTIVE 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.

### 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.

### 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.

# Internal Relationships

All options can be independently invoked.

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

# Package-Wide Variables

VA FileMan package-wide or key variables that can be assumed to be defined at all times are the following:

Table . Package-wide Variables

| Variable | Description |
| --- | --- |
| DUZ | The internal entry number from the NEW PERSON file (#200). |
| 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 following variable has a special meaning for VA FileMan although it is *not* always defined:

Table . Package-wide Variables—DISY (special Meaning)

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

## Standards and Conventions (SAC) Exemptions

Beginning January 1, 1995, VA FileMan 22.0 has been granted exemptions from the following standards by the Programming Standards and Conventions Committee (SACC).

### STANDARD SECTION: 4B–Package-wide variables

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

Table . 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 |

### 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.

# Globals

VA FileMan’s globals are listed below:

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

 **REF:** For a description of these globals, see Table 4.

The ^UTILITY and ^TMP globals are temporary globals used and then KILLed by many VA FileMan options. The ^%ZOSF global is used by the Screen Editor and ScreenMan. If VA FileMan is used with Kernel, nodes in ^%ZOSF are set up during Kernel’s installation.

 **REF:** For details about setting up the ^%ZOSF nodes at standalone sites, see the *VA FileMan Installation Guide*.

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.

## Global Journaling, Translation, and Replication

This section contains recommendations for maintaining VA FileMan globals:

* Journaling
* Translation

 **NOTE:** Translation is called “impliciting” when running M/SQL.

* Replication

### Journaling

Journaling is recommended for the following globals:

* ^DD
* ^DI
* ^DIBT
* ^DIC
* ^DIE
* ^DIPT
* ^DIST
* ^DIZ

 **NOTE:** The ^DIZ global is the default location for locally-produced files. Management of this global should be decided locally.

 Do *not* journal (translation is optional) the ^DISV global.

### Translation

Translation is *recommended* for the following globals:

* ^DDA
* ^DIA
* ^DIBT
* ^DIC
* ^DIZ

 **NOTE:** The ^DIZ global is the default location for locally-produced files. Management of this global should be decided locally.

### Replication

Replication is recommended for the following globals:

* ^DD
* ^DI
* ^DIAR
* ^DIE
* ^DIPT
* ^DIST
* ^DMSQ
* ^DOPT

# Security

VA FileMan 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.

 **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. Kernel’s Sign-on/Security component is described in the *Kernel Systems Management Guide*. If Kernel’s File Access Security system has been implemented on your system, you can use it to control user access to files.

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.

The VA FileMan menus are found on the DIUSER menu. This menu is usually located on the EVE menu distributed with Kernel.

 **REF:** For a diagram of the complete menu tree for VA FileMan, see Figure 3 in the “Exported Options” section.

## 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.

## Mail Groups and Alerts

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

## Remote Systems

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

## Interfacing

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

## Electronic Signatures

Electronic signatures are *not* used within VA FileMan.

## Security Keys

VA FileMan options are locked with the security keys described in Table 20. 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 . 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. |
| **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. |

## 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.

## References

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

Veterans Health Administration (VHA) Directive 10-93-142

## Official Policies

Modification of any part of the VA FileMan software is *not permitted* as per VHA Directive 10-93-142.

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

# Troubleshooting

There are no known issues or anomalies with VA FileMan 22.0.

For product support, contact the National Help Desk.

## 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.

Glossary

Table Table . 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. 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 Programmer 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. |
| 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. |
| 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. Learn As You 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). |
| 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

%

%ZOSV, 28

%ZOSV1, 28

^

^%ZOSF, 50

^%ZOSF Global, xi, 49, 54

^DD Global, xi, 54, 56, 57

^DD(”OS”), 49

^DDA Global, xi, 54, 56

^DI Global, xi, 54, 56, 57

^DIA Global, xi, 54, 56

^DIAR Global, xi, 54, 57

^DIBT Global, xi, 54, 56

^DIC Global, xi, 54, 56

^DIE Global, xi, 54, 56, 57

^DIPT Global, xi, 54, 56, 57

^DIST Global, xi, 54, 56, 57

^DISV Global, xi, 1, 53, 54, 56

^DIZ, 56

^DIZ Global, xi, 54, 56

^DMSQ Global, 54, 57

^DOPT Global, xi, 54, 57

^DOSV, 1

^DOSV Global, xi, 53, 54

^TMP Global, 54

^UTILITY Global, 54

A

ACTIVE by Custodial Package Option, 50

Alerts, 58

ALTERNATE EDITOR File (#1.2), 5

Application Programming Interfaces (APIs), 14

ARCHIVAL ACTIVITY File (#1.11), 5, 48

Archiving, 48

Assumptions, xii

AUDIT File (#1.1), 5, 48

B

BLOCK File (#.404), 4, 46

BOOLEAN

SQLI DATA TYPE File (#1.5211), 5

Browser, 37

C

Callable Entry Points, 14

Callout Boxes, ix

CHARACTER

SQLI DATA TYPE File (#1.5211), 5

COMPILED ROUTINE File (#.83), 4

Conventions

Documentation, viii

Cross-references, 44

Custodial Package Menu, 50

D

Data Dictionary

Data Dictionary Utilities Menu, xii

Listings, xii

DATA TYPE File (#.81), 4

Database Server, 14

DATE

SQLI DATA TYPE File (#1.5211), 5

DBA Approvals, 50

DBA IA CUSTODIAL MENU, 50

DBA IA CUSTODIAL Option, 50

DBA IA INQUIRY Option, 50

DBA IA ISC Menu, 50, 51

DBA IA SUBSCRIBER MENU, 51

DBA IA SUBSCRIBER Option, 51

DBA Menu, 50, 51

DD AUDIT File (#.6), 4, 48

DDXP-DEFINE Security Key, 59

DESTINATION File (#.2), 3

DEVICE File (#3.5), 1

DI DDU Menu, xii

DIALOG File (#.84), 4, 8

DIEXTRACT Security Key, 59

DIFROM, 8

DILIST Option, xii

DINIT Routine, x, 1, 8, 27, 29, 59

DINZMGR Routine, 49

DIPKINIT Routine, 8

Direct Mode Utilities, 32

Directives

VHA Directive 10-93-142, 14, 60

Disclaimers, vii

Software, vii, 58

DISYS Variable, 50, 52

DIUSER Menu, 38, 58

DMSQ MENU, 38

Documentation

Conventions, viii

Navigation, ix

Symbols, viii

DT Variable, 52

DTIME Variable, 52

DUZ Variable, 52

DUZ(”LANG”) Variable, 52

DUZ(0) Variable, 52, 58

E

Electronic Signatures, 59

Enter or Edit File Entries Option, 23

Entry Points, 14

EVE Menu, 38, 58

Exemptions

Standards and Conventions (SAC), 53

Export Tool, 59

Exported Options, 35

Exported PRINT Templates, 3

External Relationships, 49

Extract Tool, 48

F

Figures, v

FILE File (#1), 5, 15, 46

File Security, 59

Filegram, 49

FILEGRAM ERROR LOG File (#1.13), 5

FILEGRAM HISTORY File (#1.12), 5

FileMan

What is it?, vi

Files, 3

ALTERNATE EDITOR (#1.2), 5

ARCHIVAL ACTIVITY (#1.11), 5, 48

AUDIT (#1.1), 5, 48

BLOCK (#.404), 4, 46

COMPILED ROUTINE (#.83), 4

DATA TYPE (#.81), 4

DD AUDIT (#.6), 4, 48

Description, 3

DESTINATION (#.2), 3

DEVICE (#3.5), 1

DIALOG (#.84), 4, 8

FILE (#1), 5, 15, 46

FILEGRAM ERROR LOG (#1.13), 5

FILEGRAM HISTORY (#1.12), 5

FOREIGN FORMAT (#.44, 4

FORM (#.403), 4, 45

FUNCTION (#.5), 4

IMPORT TEMPLATE (#.46), 4, 46

INDEX (#.11), 3

INPUT TEMPLATE (#.402), 3, 45

KEY (#.31), 3

LANGUAGE (#.85), 4

Location, 3

MUMPS OPERATING SYSTEM (#.7), ix, 4, 49, 50, 52

NEW PERSON (#200), 49, 52

PACKAGE (#9.4), 8, 29, 49

PRINT TEMPLATE (#.4), 3, 44

SORT TEMPLATE (#.401), 3, 44

SQLI\_COLUMN (#1.5217), 7, 47

SQLI\_DATA\_TYPE (#1.5211), 5

SQLI\_DOMAIN (#1.5212), 6

SQLI\_ERROR\_LOG (#1.52192), 8

SQLI\_ERROR\_TEXT (#1.52191), 8

SQLI\_FOREIGN\_KEY (#1.5219), 7

SQLI\_KEY\_FORMAT (#1.5213), 6

SQLI\_KEY\_WORD (#1.52101), 5

SQLI\_OUTPUT\_FORMAT (#1.5214), 6

SQLI\_PRIMARY\_KEY (#1.5218), 7, 47

SQLI\_SCHEMA (#1.521, 5

SQLI\_TABLE (#1.5215), 6

SQLI\_TABLE\_ELEMENT (#1.5216), 5, 6, 7, 47

FOREIGN FORMAT File (#.44, 4

FORM File (#.403), 4, 45

FUNCTION File (#.5), 4

G

Global Location, 3

Globals, 54

^%ZOSF, xi, 49, 54

^DD, xi, 54, 56, 57

^DDA, xi, 54, 56

^DI, xi, 54, 56, 57

^DIA, xi, 54, 56

^DIAR, xi, 54, 57

^DIBT, xi, 54, 56

^DIC, xi, 54, 56

^DIE, xi, 54, 56, 57

^DIPT, xi, 54, 56, 57

^DIST, xi, 54, 56, 57

^DISV, xi, 1, 53, 54, 56

^DIZ, xi, 54, 56

^DMSQ, 54, 57

^DOPT, xi, 54, 57

^DOSV, xi, 53, 54

^TMP, 54

^UTILITY, 54

Journaling, 56

Replication, 57

Translation, 56

H

Help

At Prompts, xii, 60

Online, xii, 60

Question Marks, xii, 60

Home Pages

Adobe Website, xiii

VA Software Document Library (VDL) Website, xiii

How to

Obtain Technical Information Online, xii, 60

Use this Manual, vi

I

ICRs, 50

Implementation, 1

Impliciting

Globals, 56

IMPORT TEMPLATE File (#.46), 4, 46

INDEX File (#.11), 3

Initialization, 1

INPUT TEMPLATE File (#.402), 3, 45

Inquire Option, 50

Inquire to File Entries Option, 25

Installing Standalone VA FileMan, 49

INTEGER

SQLI DATA TYPE File (#1.5211), 5

Integration Agreements Menu, 50, 51

Integration Control Registrations (ICRs), 50

Current List for VA FileMan

Custodian, 50

Subscriber, 51

Detailed Information, 50

Intended Audience, vii

Interfacing, 59

Internal Relationships, 52

Introduction, vi, 1

J

Journaling

Globals, 56

K

Kernel, x, xii, 1, 8, 9, 29, 38, 49, 50, 52, 54, 58, 59, 60, 62, 63

KIDS, 1, 8, 24, 49

VA FileMan, 38

KEY file (#.31), 3

L

LANGUAGE File (#.85), 4

List File Attributes Option, xii

M

Mail Groups, 58

MailMan, xii, 49, 50, 60, 62

Maintenance, 1

Management

Security, 58

Manuals

Reference, xiii

Mapping Routines, 33

MEMO

SQLI DATA TYPE File (#1.5211), 5

Menu Structure, 35

Menus

Custodial Package Menu, 50

Data Dictionary Utilities, xii

DBA, 50, 51

DBA IA CUSTODIAL MENU, 50

DBA IA ISC, 50, 51

DBA IA SUBSCRIBER MENU, 51

DI DDU, xii

DIUSER, 38, 58

DMSQ MENU, 38

EVE, 38, 58

Integration Agreements Menu, 50, 51

Subscriber Package Menu, 51

Modify File Attributes Option, 20

MOMENT

SQLI DATA TYPE File (#1.5211), 5

MUMPS OPERATING SYSTEM File (#.7), ix, 4, 49, 50, 52

MUMPS-type Cross-references, 44

N

NEW PERSON File (#200), 49, 52

New-Style Cross-references, 3

NUMERIC

SQLI DATA TYPE File (#1.5211), 5

O

Official Policies, 60

Online

Documentation, xii, 60

Technical Information, How to Obtain, xii, 60

Options

ACTIVE by Custodial Package, 50

Custodial Package Menu, 50

Data Dictionary Utilities, xii

DBA, 50, 51

DBA IA CUSTODIAL, 50

DBA IA CUSTODIAL MENU, 50

DBA IA INQUIRY, 50

DBA IA ISC, 50, 51

DBA IA SUBSCRIBER, 51

DBA IA SUBSCRIBER MENU, 51

DI DDU, xii

DILIST, xii

DIUSER, 38, 58

DMSQ MENU, 38

Enter or Edit File Entries, 23

EVE, 38, 58

Exported, 35

Inquire, 50

Inquire to File Entries, 25

Integration Agreements Menu, 50, 51

List File Attributes, xii

Modify File Attributes, 20

Print ACTIVE by Subscribing Package, 51

Search File Entries, 30

Standalone VA FileMan, 35

Subscriber Package Menu, 51

Orientation, vi

P

PACKAGE File (#9.4), 8, 29, 49

Package-wide Variables, 52

Pointer Map, 9

Pointer Relationships, 9

PRIMARY\_KEY

SQLI DATA TYPE File (#1.5211), 5, 6

Print ACTIVE by Subscribing Package Option, 51

PRINT TEMPLATE File (#.4), 3, 44

PS Anonymous Directories, xiii

Purging, 48

Q

Question Mark Help, xii, 60

R

Reference Materials, xiii

References, 60

Relationships

External, 49

Internal, 52

Remote Systems, 58

Replication

Globals, 57

Revision History, ii

Routines, 14

DINIT, x, 1, 8, 27, 29, 59

DINZMGR, 49

DIPKINIT, 8

Mapping, 33

RTHISt, 33

RTHIST Routine, 33

S

ScreenMan-Specific Utilities, 33

Search File Entries Option, 30

Security, 58

Security Keys, 59

DDXP-DEFINE, 59

DIEXTRACT, 59

XUAUDITING, 59

XUFILEGRAM, 59

XUMGR, 59

XUPROGMODE, 59

XUSCREENMAN, 59

Security Management, 58

Software Disclaimer, vii, 58

Software Product Security, 58

SORT TEMPLATE File (#.401), 3, 44

SQLI DATA TYPE File (#1.5211)

BOOLEAN, 5

CHARACTER, 5

DATE, 5

INTEGER, 5

MEMO, 5

MOMENT, 5

NUMERIC, 5

PRIMARY\_KEY, 5, 6

TIME, 5

SQLI\_COLUMN File (#1.5217), 7, 47

SQLI\_DATA\_TYPE File (#1.5211), 5

SQLI\_DOMAIN File (#1.5212), 6

SQLI\_ERROR\_LOG File (#1.52192), 8

SQLI\_ERROR\_TEXT File (#1.52191), 8

SQLI\_FOREIGN\_KEY File (#1.5219), 7

SQLI\_KEY\_FORMAT File (#1.5213), 6

SQLI\_KEY\_WORD File (#1.52101), 5

SQLI\_OUTPUT\_FORMAT File (#1.5214), 6

SQLI\_PRIMARY\_KEY File (#1.5218), 7, 47

SQLI\_SCHEMA File (#1.521, 5

SQLI\_TABLE File (#1.5215), 6

SQLI\_TABLE\_ELEMENT File (#1.5216), 5, 6, 7, 47

Standalone VA FileMan

Options, 35

Standards and Conventions (SAC)

Exemptions, 53

Subscriber Package Menu, 51

Symbols

Found in the Documentation, viii

T

Table of Contents, iii

Tables, v

Templates

Exported PRINT, 3

TIME

SQLI DATA TYPE File (#1.5211), 5

Translating, 56

Translation

Globals, 56

Troubleshooting, 60

U

U Variable, 52

URLs

Adobe Website, xiii

VA Software Document Library (VDL) Website, xiii

Utilities

Direct Mode, 32

ScreenMan-Specific, 33

V

VA FileMan

What is it?, vi

VA FileMan with Kernel, 38

VA Software Document Library (VDL)

Website, xiii

Variables

DISYS, 50, 52

DT, 52

DTIME, 52

DUZ, 52

DUZ(”LANG”), 52

DUZ(0), 52, 58

Key, 52

Package-wide, 52

U, 52

VHA Directive 10-93-142, 3, 14, 60

W

Websites

Adobe Website, xiii

VA Software Document Library (VDL), xiii

What is VA FileMan?, vi

X

XUAUDITING Security Key, 59

XUFILEGRAM Security Key, 59

XUMGR Security Key, 59

XUPROGMODE Security Key, 59

XUSCREENMAN Security Key, 59