System Reference Guide

IS 4.0 HP Integrity Edition
and IS 4.0 SP5

9844084-003

November 2006

Notices

This document contains information proprietary to FileNet Corporation (FileNet). Due to continuing product development, product specifications and capabilities are subject to change without notice. You may not disclose or use any proprietary information or reproduce or transmit any part of this document in any form or by any means, electronic or mechanical, for any purpose, without written permission from FileNet.

FileNet has made every effort to keep the information in this document current and accurate as of the date of publication or revision. However, FileNet does not guarantee or imply that this document is error free or accurate with regard to any particular specification. In no event will FileNet be liable for direct, indirect, special incidental, or consequential damages resulting from any defect in the documentation, even if advised of the possibility of such damages. No FileNet agent, dealer, or employee is authorized to make any modification, extension, or addition to the above statements.

FileNet may have patents, patent applications, trademarks, copyrights, or other intellectual property rights covering subject matter in this document. Furnishing this document does not provide any license to these patents, trademarks, copyrights, or other intellectual property.

Please take a few moments to read the ***Software License Notice*** on the Image Services documentation CD. By installing the Image Services software, the customer agrees to be bound by the terms of this agreement.

FileNet, ValueNet, Visual WorkFlo, and OSAR are registered trademarks of FileNet Corporation.

Document Warehouse and UserNet are trademarks of FileNet Corporation.

All other product and brand names are trademarks or registered trademarks of their respective companies.

FileNet Corporation
3565 Harbor Boulevard
Costa Mesa, California 92626
800.FILENET (345.3638)
Outside the U.S., call:
1.714.327.3400
www.filenet.com

Contents

3 Applications 61

1

System Design and Architecture

This System Reference Guide is intended for use by support personnel responsible for Image Services (IS) software.

Overview

This guide discusses the system architecture, directory structure, MKF database structure and contents, and descriptions of shared libraries.

This guide assumes that you have knowledge of the following subjects:

- An understanding of the operating system (AIX, HP-UX, Solaris, or WIndows) associated with the platform you are using
- An understanding of, or prior experience with, the system administration tools available on a specific platform (**smit** on AIX, **sam** on HP-UX, **admintool** and **swmtool** on Solaris 8, **Solaris Management Console** tools on Solaris 9, and **Control Panel** on Windows)
- An understanding of file system configuration methods and logical volume management concepts.

Note If you have questions about the information in this document, contact the FileNet Response Center (FRC) or the FileNet support person responsible for your site.

Scope of Guide

This guide contains a brief discussion of the following Image Services system items:

- Server configurations
- Directory structure (and content)
- Subsystem descriptions
- Configuration files (some supplied with the Image Services software others generated at the time of installation)
- Application programs
- shared libraries
- MKF database tables/contents (permanent, transient, and security)

The following table details where additional system support information is documented and the type of information you can expect to find in each document.

Document Name	Document Description
System Messages Handbook	A detailed discussion of system messages and an explanation of the message tools.
System Administrators Handbook	A detailed discussion of critical System Administrative tasks, including database management, security management, system management, storage library control, and more.
System Administrator's Companion for UNIX ***System Administrator's Companion for Windows***	A discussion on platform-specific operating system operation. Includes appendices detailing the types of tape drives and media supported. It also discusses backing up the RDBMS database files.

Document Name	Document Description
System Tools Reference Manual	A description of the system tools used to diagnose and manage a FileNet system. Some tools are used only by FileNet support personnel whereas many others are used by Image Services System Administrators to analyze, diagnose, update, and repair Image Services components. **Note:** This manual is available only to those who are authorized to provide support functions on FileNet servers.
Enterprise Backup/Restore User's Guide	A description of Enterprise Backup/Restore (EBR), EBR concepts and functions, and how to use EBR to backup and restore a FileNet system.
Third-Party Backup/Restore Guidelines	A listing of general requirements and recommendations when using non-FileNet backup and restore programs or utilities.
Restore Guide for Windows	A discussion of methods used for restoring the permanent, transient, security, and RDBMS databases.

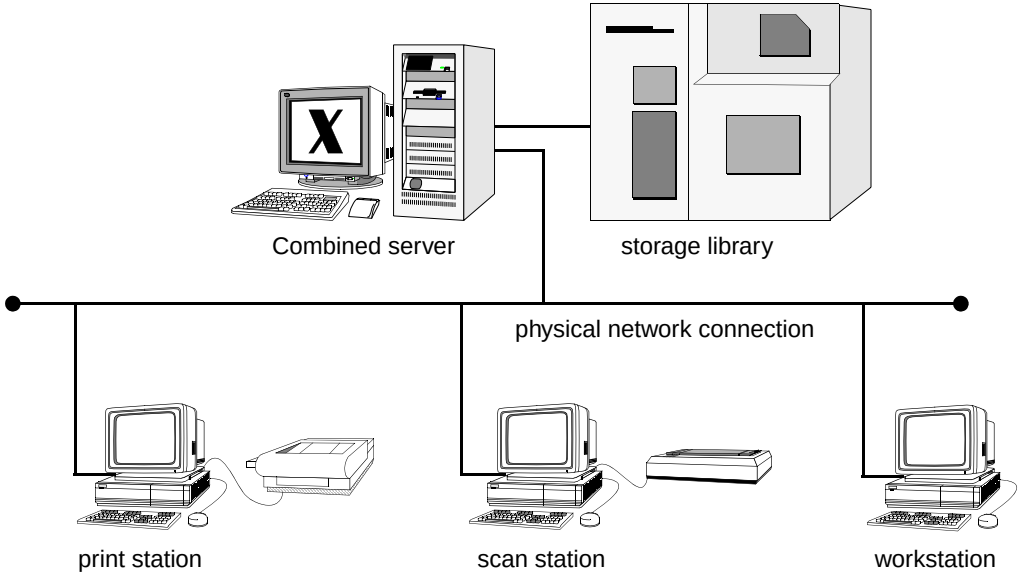
Server Configurations

The Image Services software is designed to operate in a distributed environment. The individual services run in the following configurations:

- Root/Index/Storage Library server: Combined server
- Root/Index and Storage Library servers: Dual server or Multiple Storage Library server system
- Application server
- Visual workflo server
- Entry server (Remote Entry)

Combined Server Configuration

Within the Combined server configuration, only one FileNet Image Services server exists and it hosts all IS services. The Combined server configuration is typically used in development or small production environments. The figure below shows the typical configuration for a Combined server system.

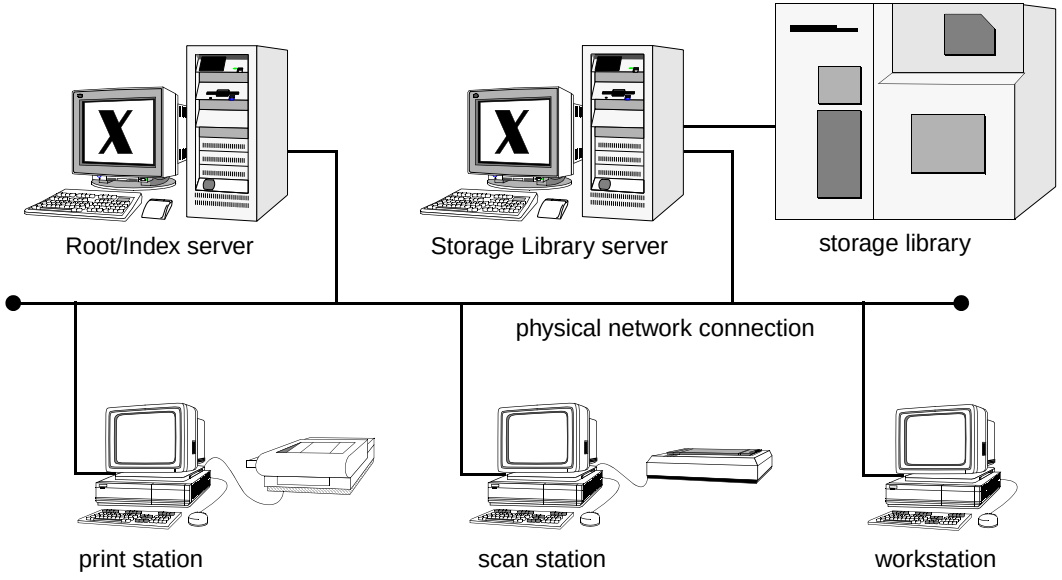


In this configuration the root/index, the Storage Library services, and all additional services reside on the same Combined server system.

Dual or Multiple Storage Library Server Configuration

In a Dual or Multiple Storage Library server configuration, at least two FileNet Image Services servers exist. The first server hosts the root/

Index services and the other server(s) hosts only the storage library service. The RDBMS software is installed only on the Root/Index server. There can be more than one Storage Library server installed on a Multiple Storage Library server system. The figure below shows the typical configuration for a Dual server system.

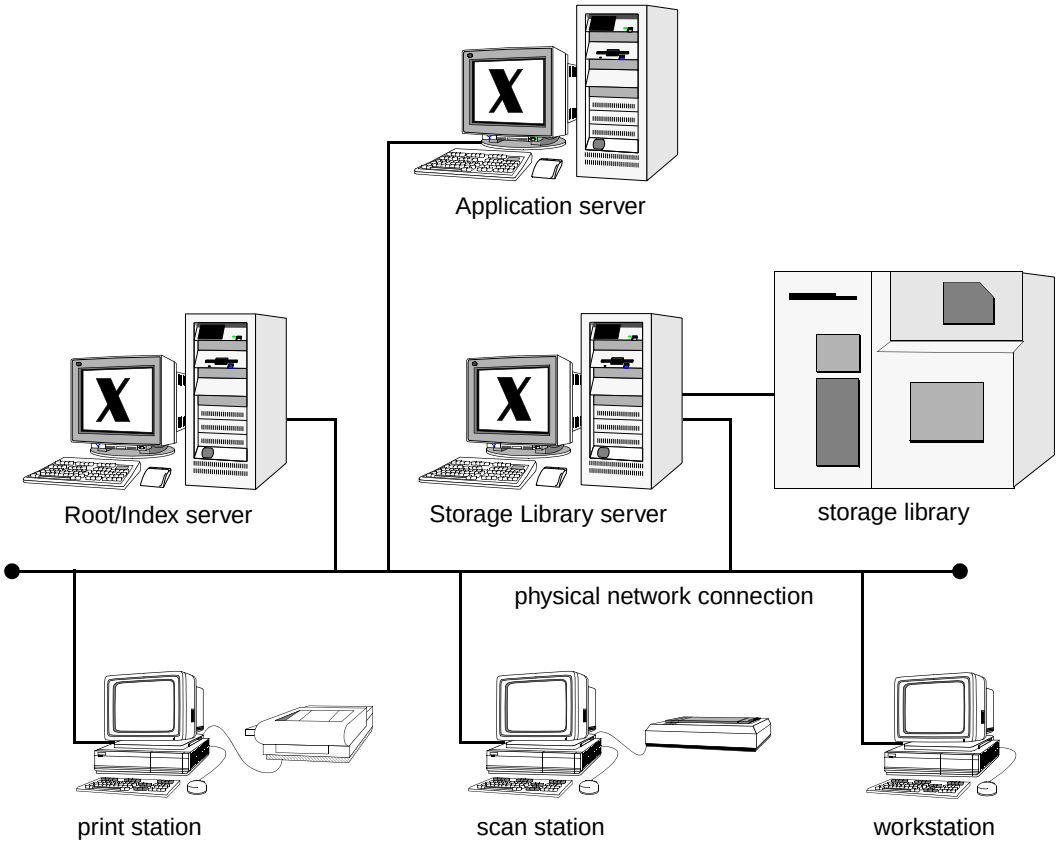


For information concerning the reasons why a Dual server configuration is desirable, consult your FileNet Account Executive.

Application Server Configuration

An Application server configuration consists of two or more Image Services servers. An Application server is a separate, dedicated server that hosts specific services (such as WorkFlo Queue or VW services). The Application server must be attached to a Combined or Dual server

system. The figure below shows an Application server attached to a Dual server system.



If the Application server is hosting either Process services, SQL services or VW services, the RDBMS software is installed on both the Root/Index server and the Application server. Otherwise the RDBMS software is installed only on the Root/Index server.

You can add more than one Application server to the system if the requested resources outstrip the ability of the first Application server.

The following services can exist on more than one Application server in a system:

- Batch services
- Cache services
- Print services
- Structured Query Language services
- Process services
- VW Services

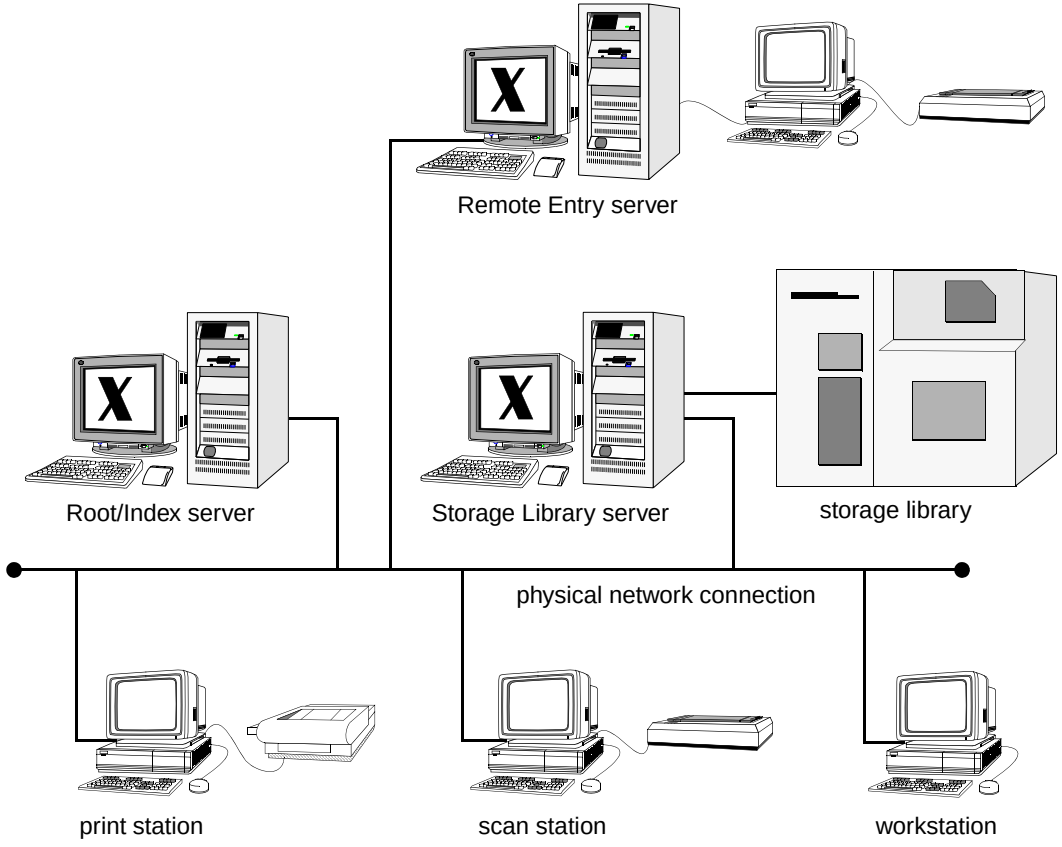
The Application server configuration provides maximum system performance since the Application server can be configured to assume responsibility for services that might otherwise slow the other servers in the system.

Entry Server Configuration

An Entry server configuration (a Remote Entry server system) consists of two or more servers.

The Remote Entry server (RES) is configured as a Combined server without any storage library devices attached. Documents are scanned in at the Remote Entry server and sent over the network to a system that has storage capabilities.

The figure below shows a Remote Entry server attached to a Dual server system.



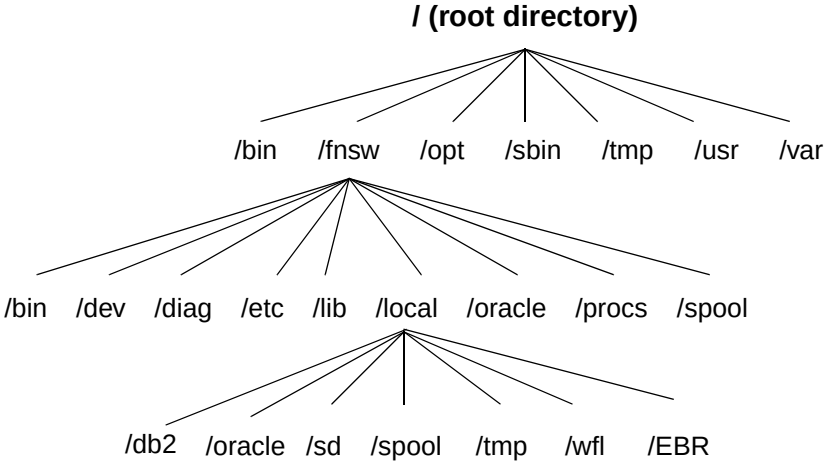
The Entry server's main function is to enter documents into the system. An entry system provides document entry on behalf of a target system.

It functions independently of the target system's status. (For example, the target system can be on-line or off-line.)

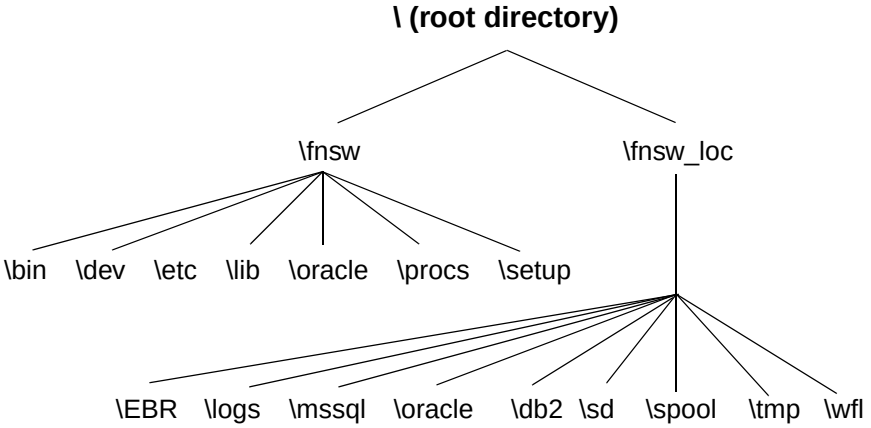
Directory Structure

The FileNet Image Services software resides in unique directories on both the UNIX and Windows platforms.

The illustration below shows the typical directory structure on UNIX platforms. The directory structure is an approximation for all UNIX platforms. The actual structure may vary depending on platform differences or changes to the intended directory structure introduced during installation.



The illustration below shows the typical directory structure on the Windows platform.



Directory Contents

While the directory structure may be different depending on server usage or platform differences, all Image Services servers have the programs, shared libraries, tools, etc., residing in the same relative directory locations. The following sections provide a general overview of the contents of FileNet Image Services directories. The actual path designation varies according to the platform.

/fnsw/bin

The /fnsw/bin directory contains FileNet-specific tools and application programs. For a complete discussion on the tools located in the /fnsw/bin directory, consult the *System Tools Reference Manual*. If you are not authorized to perform support functions on your FileNet system, you will not have access to this manual.

The default path is:

- UNIX: **/fnsw/bin**
- Windows: **\fnsw\bin**

/fnsw/dev/1

On the UNIX platform, the /fnsw/dev/1 directory contains links/device files to Image Services and FileNet-controlled RDBMS database files. This directory contains pointers to raw partitions containing data in MKF databases, the FileNet-controlled RDBMS database, and cache. The actual data is in different physical locations and their directory names depend on the server platform.

On the Windows platform, the data files physically reside in the \fnsw\dev\1 directory. This directory is the default location for MKF database, RDBMS, and cache files. The user determines on which disk drive to place MKF database and cache files. The user can use the FileNet utility, fn_edit, to identify file locations.

The default path is:

- UNIX: **/fnsw/dev/1**
- Windows: **\fnsw\dev\1**

/fnsw/diag (UNIX Only)

The /fnsw/diag directory contains hardware diagnostics programs and utilities specifically related to storage library devices.

The /fnsw/diag directory exist on the UNIX platforms only. The default path is:

- UNIX: **/fnsw/diag**

/fnsw/etc

The /fnsw/etc directory contains administration files, initialization applications, and scripts. Most of the files located in this directory are used either during the Image Services software installation process or by support personnel during diagnostic evaluation of the system.

The default path is:

- UNIX: **/fnsw/etc**
- Windows: **\fnsw\etc**

/fnsw/lib

The /fnsw/lib directory contains library and configuration files. This directory also contains EBR scripts and X11 uid files and help files for Motif.

The /fnsw/lib/conf_db subdirectory also contains the server templates for configuration. The /fnsw/lib/dev/msg subdirectory contains the .msg and .cat files. The /shobj subdirectory contains FileNet shared libraries (sometimes called shared objects, abstracts, or loadable libraries). Also in here are the /nls, /nltmaps, /perf, and /translate subdirectories.

The default path is:

- UNIX: **/fnsw/lib**
- Windows: **\fnsw\lib**

/fnsw/local (or \fnsw_loc)

The /fnsw/local (or on Windows: \fnsw_loc) directory contains subdirectories critical to the operation of the FileNet Image Services software. On most platforms, this directory is a file system that is automatically mounted during system startup.

The default path is:

- UNIX: **/fnsw/local**
- Windows: **\fnsw_loc**

/fnsw/local/sd

The /fnsw/local/sd (on Windows \fnsw_loc\sd) subdirectory contains the NCH (Network Clearinghouse) database. In addition, this subdirectory contains the server configuration files for the local server. On a Combined server system, the root_station file resides in this subdirectory. This subdirectory also contains service information and configuration data.

/fnsw/local/spool

The /fnsw/local/spool subdirectory contains only the network log files for the TM_daemon process.

- UNIX: **/fnsw/local/spool**
- Windows: **\fnsw_loc\spool**

/fnsw/local/tmp

The /fnsw/local/tmp subdirectory contains configuration log files. This directory also contains an EBR subdirectory (/fnsw/local/tmp/EBR) which contains sample EBR and shell scripts.

- UNIX: **/fnsw/local/tmp**
- Windows: **\fnsw_loc\tmp**

/fnsw/local/wfl

No information available. Information will be added as it becomes available.

- UNIX: **/fnsw/local/wfl**
- Windows: **\fnsw_loc\wfl**

/fnsw/local/<RDBMS>

The existence of this subdirectory depends upon the platform type and the type of RDBMS software installed on the system.

\fnsw_loc\oracle, \fnsw_loc\db2 or **\fnsw_loc\mssql** on Windows

- On a system with the Oracle RDBMS software installed, the Oracle subdirectory exists. The /fnsw/local/oracle subdirectory contains subdirectories, Oracle parameter files, and SQL scripts for starting and stopping the Oracle software. In addition, this subdirectory contains subdirectories in which the Oracle RDBMS control files reside.
- On a system with the MS SQL Server RDBMS software installed, the \fnsw\local\mssql subdirectory exists. The \fnsw\local\mssql directory contains parameter files for the RDBMS software.
- On a system with the DB2 RDBMS software installed, the \fnsw\local\db2 subdirectory exists.

RDBMS (/fnsw/oracle or \fnsw\mssql)

The actual path designation varies according to the RDBMS software installed on the system:

- UNIX (Oracle Only): **/fnsw/oracle**
- Windows (Oracle, DB2 and Microsoft SQL Server): **\fnsw\oracle, \fnsw\db2 or \fnsw\mssql**

The /fnsw/oracle directory contains SQL scripts used primarily for Oracle initialization. This directory contains scripts for installing patches.

On the Windows platform, if Microsoft SQL Server is installed on the system, the \fnsw\mssql directory exists. If Oracle is installed on the system, the /fnsw/oracle directory exists.

/fnsw/procs

The path, separated by platform, is shown below:

- UNIX: **/fnsw/procs**
- Windows : **\fnsw\procs**

The /fnsw/procs directory stores process files for all FileNet Image Services processes. If the directory does not exist the FileNet Image Services software will not start and will report serious errors.

/fnsw/spool (UNIX Only)

The /fnsw/spool directory contains a COR subdirectory (/fnsw/spool/COR) to which all Courier process log files are sent.

\fnsw\setup (Windows Only)

The Image Services software installation and setup software resides in this directory. The permission configuration files are also located here. Do not delete this directory or remove the contents after installing the Image Services software.

Subsystem Descriptions

Within the context of discussing FileNet Image Services, services are actually subsystems that have the ability to operate with the other subsystems in the Image Services software.

This section provides a summary discussion of each of the services within the FileNet Image Services software.

AT (Release Tools)

Tools used during the installation and updating of the Image Services software.

AR (Archiving)

FileNet Archiving services moves document index and locator information from magnetic disk to optical disk. This frees up space on magnetic disk, allowing more documents to be added to the system without increasing the magnetic disk database size.

B2 (Enterprise Backup/Restore)

FileNet Enterprise Backup and Restore (EBR) is a set of scripts and programs designed to automate the process for protecting data

through backups and restores. Restores are performed infrequently, usually to recover from a disk failure or other disaster.

BS (Batch Services)

The Batch services are responsible for creating, updating, and deleting batches; keeping track of image and index verification status; accumulating batch totals; creating documents from images; maintaining the batch queues; and maintaining the transient database of batches in progress. A batch service has a three-part name with the batch service name first, followed by the system name followed by the organization: BatchService:pubs:FileNet.

CS (Cache Services)

The Cache services are responsible for moving documents between different caches; the services are also responsible for overall cache management. Cache services provides clients a magnetic disk based storage and retrieval mechanism for objects. Routines are provided to log on to or log off from a cache service, to create, open, close, read, write, delete, move, rename, and copy objects, to get attributes of a cache, and to list objects in a cache.

DM (Database Maintenance)

The database maintenance subsystem is a collection of programs that give the user the ability to set up and maintain database information needed for the storage and retrieval of documents.

DS (Document Services)

The Document services are responsible for all optical disk activity related to prefetching, assigning document IDs, locating documents (by DOC ID), migrating documents (to and from disk), copying, and importing documents.

EV (SystemV Applications Executive)

The System V Applications Executive is a set of applications that manages the state of the Image Services software. The functionality of each application is as follows:

- **Xtaskman** is the graphical user interface application used to control the state of the Image Services software, list the current FileNet processes, view the error log, and view RPC activity.
- **initfnsw** is the console application used to control the state of the FileNet Image Services software.
- **whatsup** is the console application used to list the current FileNet processes.
- **vl** is the console application used to view the error log.
- **Xapex** is the graphical user interface application used to log into Image Services. The menu options assist users to initiate other server applications after login.

FL (FileNet LAN Analysis Tool)

The FileNet LAN Analysis Tool (FLAT) is a support tool that simplifies local area network (LAN) packet trace analysis. The tool can read and decode the output of network analyzer programs and network packet

traces on UNIX platforms. It provides some FileNet-specific analysis capabilities not found in other commercially-available analyzers and analyzes traces captured from a variety of other tools, such as: LANalyzer and Sniffer, LANwatch, iptrace (AIX/6000 platform only, and snoop (Solaris Operating Environment platform only).

GB (Generic Database)

The Generic Database subsystem is a set of libraries (GDB, GDBO, GDBM, ORA) that handle the interface between the FileNet software and the supported Relational Databases, Oracle, DB2 and Microsoft SQL Server. All subsystems that want information from the RDBMS will call the GDB libraries with their database queries and updates.

HE (Help Error)

The Help Error subsystem controls the areas of Help text and error messages.

IS (Index Services)

Index Services handles queries and updates to the index database. It stores the document index records, document class and folder information, and the data dictionary.

LM (Software License Management)

FileNet Software License Management (LM) is a set of programs and libraries designed to manage the use of various components of the Image Services system that are separately licensed. It introduces the concept of a Licensed User and Licensed Configurations.

MK (Multi-Keyed Files)

The multi-keyed files subsystem manages the transient database (Trans_DB*n*), the permanent database (Perm_DB*n*), the network clearinghouse database (NCH_db0), and the security database (Sec_DB*n*).

MV (SystemV Miscellaneous)

The MV subsystem controls programming utilities, shared memory and killfnsw.

NC (Network Clearing House)

The Network Clearinghouse (NCH) allows the distributed resources to be located. System elements can locate the services by appropriate calls to the Clearinghouse service. The name of a resource consists of three parts that identify a resource object, domain (system name), and the organization (FileNet): object:domain:organization

NL (National Language Services)

The NL subsystem controls the localization tools.

NV (System V Networking)

System V Networking deals with network protocols, Courier, and process management.

OS (OSAR Services)

The OSAR services are responsible for monitoring and controlling all traffic (input/output) between the server and the storage library devices. These services are supported by Document services.

PM (Performance Measurement Tools)

The Performance Measurement Tools (PM) subsystem contains programs designed to measure some aspect of performance on a system. For example, one such program, CPT_test, measures the Courier network throughput and latency using the CPT's (Courier Performance Testing Tool's) server.

PR (Print Services)

Print Services manages print requests for documents or text streams and displays information on printer attributes, printer status, and job queue status.

PR (Print Services II)

Print Services II is a newer version of Print services and performs the same functions as Print services.

PV (Server Print)

The Server Print services are responsible for print documents in cache or in memory. Controls print requests and monitors printing status.

SE (Security Services)

FileNet Security services (SE) is a set of programs and libraries that provide user authentication and access control for the Image Services system. Access control is based on hierarchical group membership and distinguishes Read - Write - Execute/Append privileges.

SF (System Configuration Tools)

The SF subsystem controls the tools used to initialize and update the Image Services configuration databases, configuration files, and devices.

SU (System Utilities)

System Utilities are miscellaneous utility programs like **ident**, **stamp**, **less**, and **uncompressdir**.

WQ (WorkFlo Queue Services)

The WorkFlo Queue services are responsible for monitoring and controlling WorkFlo queues.

XV (XVT Portable Toolkit)

XVT Portable Toolkit is a third party toolkit on which the graphical user interface server applications are built.

2
Configuration Files

Each FileNet system must be configured properly before it is used in a production environment. The configuration process includes configuring hardware and software parameters and device files. To manage all configuration settings and maintain consistent performance, the Image Services software uses configuration files to control how each component or device is set up.

A typical FileNet Image Services system contains numerous configuration files. Some files contain parameters for several Image Services components, procedures, and databases. Other files control settings for only a single component.

Configuration files contain numerous parameters determining the configuration for each component installed on the system. Default parameters are established during the Image Services software installation; however, you can change the parameters when necessary.

Configuration Database

The IMS_xxx.CDB file is the main configuration database file for a FileNet system. It is created and modified by the fn_edit program. For a complete description of the fn_edit utility, see ***System Configuration Overview***.

The IMS_xxx.CDB file is located in the following directory (where xxx is a sequentially increasing number assigned automatically by the system):

- UNIX: **/fnsw/local/sd/conf_db/IMS_xxx.cdb**
- Windows: **\fnsw_loc\sd\conf_db\IMS_xxx.cdb**

You can use the System Configuration Editor to view current configuration information.

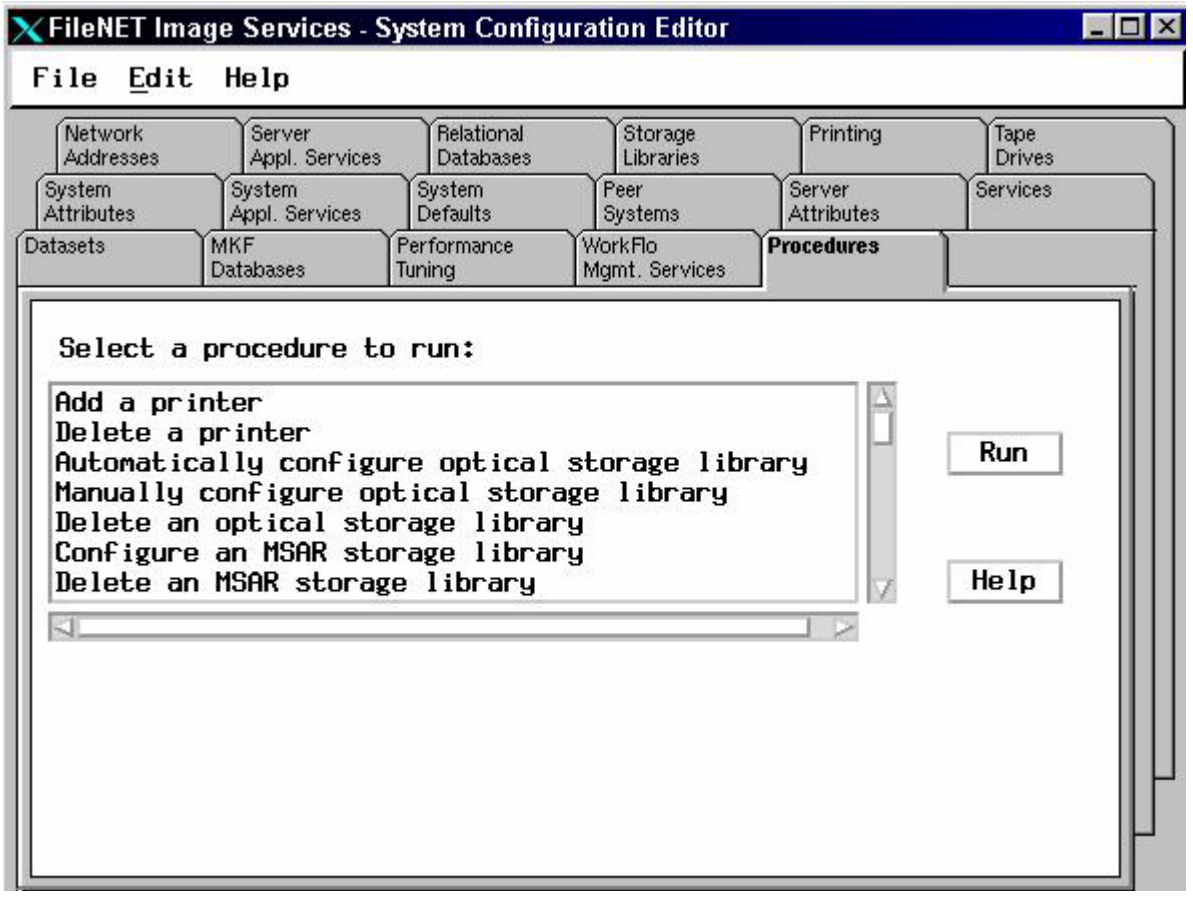
1 At the system prompt, enter the following command:

fn_edit &

Note On UNIX platforms you must be running an X Windows/Motif interface to use the System Configuration Editor.

Windows Only: double-click the Configuration Editor icon.

2 Click on any of the available tabs to view the configuration settings.



Note Starting with release 3.3.0, the **IMS_xxx.CDB** database file took the place of the **system** configuration file and the **initparms** file, which was produced by the **es** tool.

When the IS software is initially installed, the fn_edit program accesses read-only templates to create the first configuration database file: IMS_1.CDB.

However, as you make and save changes to the system configuration (through the fn_edit interface), the version number of the .CDB file increments accordingly. (The earlier .CDB files are left unaffected in the original directory.)

The IMS_xxx.CDB file contains information about all aspects of IS software operation including, but not limited to, the following items:

- Cache sizes and locations
- Service configurations and locations
- Dataset sizes and locations
- RDBMS type and version information
- Tablespace names and locations
- Network configuration information
- Storage library configuration information
- Peer system information
- Performance tuning parameters

Supplied Configuration Files

This section describes configuration files that are placed on the server at the time of the IS software installation.

Note You can view any of the files discussed in this chapter using a native text editor (Windows - Wordpad, Solaris - textedit, all UNIX platforms - vi, etc.)

erm.msg

The erm file is the main message file on the system containing most of the IS messages. The erm file is located in the following directory (separated according to platform):

- UNIX: **/fnsw/lib/nls/msg/erm.msg**

A partial listing of the contents of the erm.msg file is shown below.

```
00001 <15,255,32>SPP_ErrSuspCreated: Connection suspended
00002 <15,255,4096>ERP_ErrUnspec: Unspecified error was detected at
destination
00003 <15,255,4352>ERP_ErrUnspec2: Unspecified error before reaching
destination
00004 <15,255,4608>ERP_ErrBadConn: Packet send to suspended transport
connection
00005 <15,255,4864>ERP_ErrNoEpds: Unable to allocate epd to send error
pkt
00006 <15,255,8192>COR: Unknown remote service
00007 <15,255,12288>An attempt was made to access a file for a network
service that is either not installed or not configured on this host.
(CONFIGURATION ERROR)
...
```

UNIX Files

The .cat (catalog) and .msg (message) files are provided as part of the standard IS software. Both files contain the same information in different formats. The .msg files are in ASCII and are input files to the .cat files in binary, which are the output files. The UNIX tool, gencat, uses the .msg files to build the .cat files that contain language-specific information for the messages. The .cat files are data files that are accessed by the application programs and cannot be read with any of the conventional viewing commands.

The /fnsw/lib/msg.category file is referenced by the msg command to map a message tuple number with the name of a category, usually a shared library. The file is provided as part of the standard IS software. It cannot be read with conventional viewing commands.

Windows Files

The .dll files are provided as part of the standard Release software and are the equivalent to the .cat (catalog) files on the UNIX platforms.

The .dll files are data files that are accessed by the application programs and cannot be read with any of the conventional viewing commands. The equivalent to the UNIX .msg files are not included in the standard IS software.

Language Directory

For UNIX, this /fnsw/lib/nls/msg/<lang> directory contains the same three .cat files described above but with messages in a language other than English. To use this directory, you set a language environment variable. Otherwise, Image Services uses the English files in the default path. For example, the de_DE, for Deutsche, directory contains

the .cat files in German. The .msg files are created manually in German, and then the gencat tool is run to build the .cat files.

Refer to the ***Image Services System Messages Handbook*** for more information.

fninfo.msg

The different platforms use unique message resources to provide information on the IS status. The system information file is located in the following directory:

- UNIX: **/fnsw/lib/nls/msg/fninfo.msg**

A partial listing of the fninfo.msg (for the UNIX platforms only) is shown below.

```
...
05953 <170,12,1>For %s tape %d, please mount tape with serial number %.6s
05954 <170,12,2>For %s tape %d, please mount a labelled tape.
05955 <170,12,3>For %s tape %d, please mount an unlabelled tape.
05956 <170,12,4>Enter 1 (or <CR>) when tape ready, 2 if no tape available
05957 <170,12,5>For %s tape %d (labelled), overwrite unlabelled tape?
(1=yes,2=no)
05958 <170,12,6>For %s tape %d (unlabelled), overwrite labelled tape?
(1=yes,2=no)
...
```

Note **Windows:** The ERM_INFO file is used exclusively by the FileNet backup and restore programs.

The fninfo.msg also contains SNMP trap messages. SNMP traps are alerts that are pro-actively generated by the agent software and sent to

the third party network management software. (Refer to **“snmp.conf” on page 49** for more snmp information.)

tmmsg.msg

The tmmsg file is the task manager message file used by Xtaskman and initfnsw. The tmmsg file contains all of the dynamic task management messages that appear in the *Current Status* pop-up window when you recycle the FileNet software.

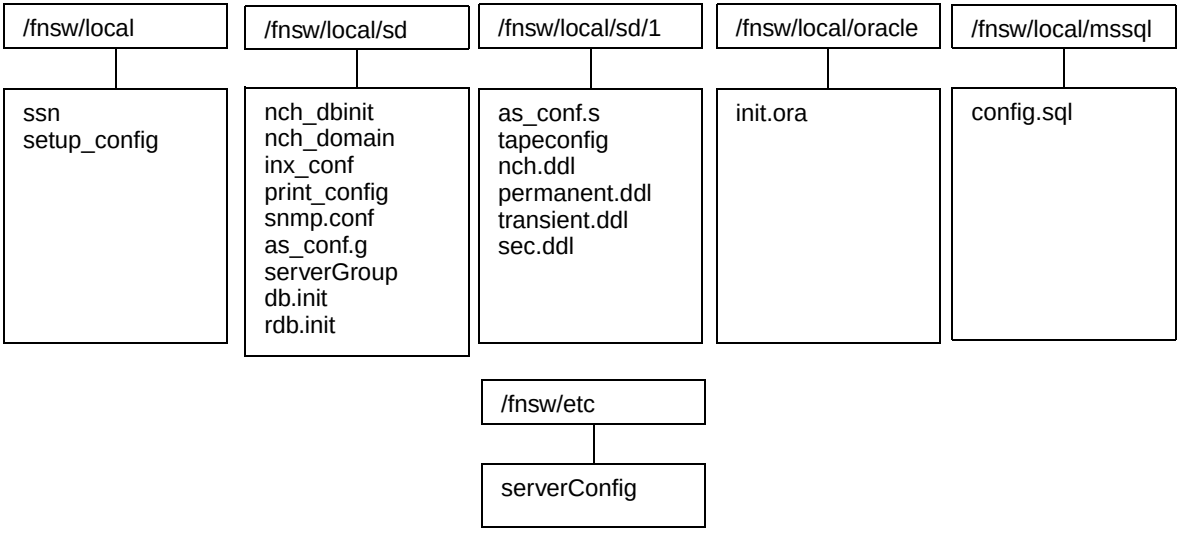
The tmmsg file is located in the following directory:

- UNIX: **/fnsw/lib/nls/msg/tmmsg.msg**

Generated Configuration Files

All of the files discussed in this section are generated by the fn_build program after Image Services software installation. The information contained in the generated files comes from a combination of system information and information entered in the system configuration database file.

The fn_build program automatically creates the new files in the directories shown below.



as_conf.g

The as_conf.g file contains all the system-specific Document services configuration information that applies to the entire domain. It is located in the following directory:

- UNIX: **/fnsw/local/sd/as_conf.g**
- Windows: **\fnsw_loc\sd\as_conf.g**

Note Do not manually edit the contents of the as_conf.g file. Fn_build runs every time you start/restart IS. You can also run fn_build after modifying the file. In both cases, modifications will be overwritten. **Manually editing the as_conf.g file is NOT recommended.**

The content of a typical as_conf.g file is shown below.

```
processes {
  notify    ds_notify 2
  scheduler     dsched
  dtp           dtp
  dtp_tran       dtp_tran 1
  rmt_commit    rmt_commit 1
  fbc_commit    fbc_commit
  del_commit    del_commit
  osi_migrate   osi_migrate
}

compatible_systems {
}

delete_on_write f
fbc_breakup f
fbc_breakup_count 400
conversion_system f
pages_per_retrieval 8
page_cache_doc_duration 300 300 120
docnum_range 100000 3999999999
surfaceid_range 3000 4294967290
doc_batch_bufsize 80
ims_name DefaultIMS:earth:FileNet
```

as_conf.s

The as_conf.s file is located in the following directory:

- UNIX: **/fnsw/local/sd/1/as_conf.s**
- Windows: **\fnsw_loc\sd\1\as_conf.s**

The as_conf.s file handles all the server-specific Document services configuration information. The content of a typical as_conf.s file is shown below.

```
cache_partitions {
  /fnsw/dev/1/cache0 0 102400
}
special_caches {
  FN_print_cache print_cache1:earth:FileNet
}
scheduling_options {
  minimum_transfers 50
  critical_read_time 60
  write_delays 50 60 600
  io_active_time 1800
  spindown_delay 6
  prefetch_delays 0 0
}
data_base_names {
  permanent /fnsw/dev/1/permanent_db0 DocServer
  transient /fnsw/dev/1/transient_db0 DocServer
}
document_bufcnt 16
document_bufsize 64
csm_io_process_cnt 2
read_lookahead_cnt 10
dir_bufsize 16
dir_bufcnt 4
cache_threshold 85 80 90
page_cache_doc_duration 300 300 120...
```

init.ora

The init.ora is a text file containing a list of instance configuration parameters. This file contains Oracle database parameters and is built by fn_build. In FileNet-controlled installations, Oracle software reads the init.ora file each time the RDBM software starts. The init.ora file is located in the following directory:

- UNIX: **/fnsw/local/oracle/init.ora**
- Windows: **\fnsw_loc\oracle\init.ora**

The Oracle software must be able to read the init.ora file to start an instance. If the init.ora file is not present on the server or is corrupted, no Oracle instance will start. The content of a typical init.ora file is shown below.

```
control_files = /fnsw/local/oracle/control0/ctl.ora,/fnsw/local/oracle/control1/
ctl.ora
db_name = indexdb
nls_territory = AMERICA
db_block_buffers = 1000
db_block_size = 2048
db_files = 200
db_file_multiblock_read_count = 8
dml_locks = 100
log_archive_start = FALSE
log_checkpoint_interval = 50000
log_buffer = 163840
max_dump_file_size = 10240
processes = 64
shared_pool_size = 64M
compatible = 9.2.0.1.0
sort_area_size = 131072
rollback_segments = rs0,rs1,rs2,rs3
# WARNING: lines above this line will be replaced by FileNet at startup.
```

You can edit parameter values in a parameter file with a basic text editor; however, editing methods are operating system-specific. Normally, this file is rebuilt each time the software is started (by fn_build). Changes should be made using fn_edit, except for entries below the **#WARNING** line. Oracle treats string literals defined for National Language Support (NLS) parameters in the file as if they are in the database character set.

Note Additions to the init.ora file below the # WARNING ... line will **NOT** be destroyed by **fn_build**.

inx_conf

The inx_conf file identifies if documents will be catalogued to the FileNet-defined oracle database, and if user indexes will be used in distributing queues. The inx_conf file is located in the following directory:

- UNIX: **/fnsw/local/sd/inx_conf**
- Windows: **\fnsw_loc\sd\inx_conf**

The content of a sample inx_conf file is shown below.

```
no_catalog = 0
ixwfq_ctl_file = 1
```

snmp.conf

The snmp.conf file is located in the following directory:

- UNIX: **/fnsw/local/sd/snmp.conf**
- Windows: **\fnsw_loc\sd\snmp.conf**

The content of a sample snmp.conf file is shown below.

```
OkTrapFlag = 1
WarningTrapFlag = 1
OperatorTrapFlag = 1
SevereTrapFlag = 1
```

MKF.ddl (s)

The Data Definition Language (ddl) files contain the parameters used when an MKF database is created. The fn_build program runs a different script to create each MKF database. The data definition files are located in the following directory:

- UNIX: **/fnsw/local/sd/1/*.ddl**
- Windows: **\fnsw_loc\sd\1*.ddl**

Once an MKF database is created, support personnel may need to update it due to a software upgrade or changes to performance tuning parameters.

The table below outlines the scripts used with the ddl files. The fn_build program uses a different script for the Permanent and Transient data-

bases following some upgrades. The table below associates each MKF database with a ddl file and the script that creates it or updates the file.

*.ddl (MKF) File	Database	Created by	Updated by
nch.ddl	NCH	fn_util initnch	fn_util initnch
permanent.ddl	Permanent	fn_util initperm	fn_util updateperm
sec.ddl	Security	fn_util initsec	fn_util updatesec
transient.ddl	Transient	fn_util inittrans	fn_util updatetrans

The contents of the ddl files vary slightly from one file to another. However, they all contain default parameters for each MKF database including the size of the database and recovery log partitions. The content of a typical permanent.ddl file is shown below:

```
--permanent Document Server Database DDL
PARAMETERS
(
number_of_buffers = 256,
max_concurrent_transactions = 3,
max_concurrent_long_transactions = 1,
read_after_write = set,
max_record_types = 32,
max_items_per_record = 32,
overwrite_rl_action = warning_message,
rl_update_frequency = 230
);
FILES
(
target_station "DocServer";
base data partition "/fnsw/dev/1/permanent_db0" (blocks =
102400);
recovery_log partition "/fnsw/dev/1/permanent_rl0" (start
= 0, blocks = 40960);
);
#include "/fnsw/lib/perm_db.desc"
```

nch_dbinit

This configuration file is the Network Clearinghouse database initialization file. The fn_util initnch program builds the NCH database using parameters stored in this file. The fn_build program is responsible for transferring modified information from IMS_xxx.CDB to nch_dbinit.

The nch_dbinit file is located in the following directory:

- UNIX: **/fnsw/local/sd/nch_dbinit**
- Windows: **\fnsw_loc\sd\nch_dbinit**

The content of a typical nch_dbinit file is shown below.

```
# nch_header
init hint:defaults:FileNet
#
primarydomain glacrpt:FileNet
set domain glacrpt
set organization FileNet
removeremotes
...
# nch_ims
createobj DefaultIMS
additem DefaultIMS ims "System IMS Defaults"
additem DefaultIMS imsDesc 256 IndexServer DocServer 10337
FormsServer

# nch_systemDefault

createobj fn_system
additem fn_system system "System Defaults"
additem fn_system defServiceDesc 2 Bes1 \ DefaultIMS
PrintServer SKFserver WflS
erver \ SQLServer
additem fn_system defService1Desc 1 \ RFSServer
SecurityService \
additem fn_system defDeviceDesc 0 \ \
additem fn_system defCacheDesc 2 \ batch_cache1 revise_cache1
\ \ \
additem fn_system attributesDesc 1 1 11 11
...
```

nch_domain

This file is created by the fn_build utility. It contains the domain:organization names of the system. The fn_build utility reads the system file and creates nch_domain. The fn_build utility then calls the program nch_tool to update the NCH_db0 with the contents of the nch_domain file. The nch_domain file is located in the following directory (separated according to platform):

- UNIX: **/fnsw/local/sd/nch_domain**
- Windows: **\fnsw_loc\sd\nch_domain**

The content of a sample nch_domain file is shown below.

```
earth:FileNet
```

print_config

The print_config file contains both global and specific Print services parameters. This configuration file contains a description of each printer for all Print services on the system.

Note If a print_config file exists, it is renamed print_config.old when the fn_build program rewrites it each time fn_build is run.

The print_config file is located in the following directory (separated according to platform):

- UNIX: **/fnsw/local/sd/print_config**
- Windows: **\fnsw_loc\sd\print_config**

The partial content of a sample print_config file is shown below.

```
print_servicePrintServer:earth:FileNet  {
print_cachesys_print_cache1:earth:FileNet
cache_fullness_threshold 80
print_page_bytes50000
...
```

rdb.init

The rdb.init file contains a start_stop parameter at a value of 1 or 0. This file is located in the following directory (separated according to platform):

- UNIX: **/fnsw/local/sd/rdb.init**
- Windows: **\fnsw_loc\sd\rdb.init**

The content of a typical rdb.init file is shown below.

```
start_stop 1
```

serverGroup

The serverGroup file is used by the PPM shared library to enforce the number of licensed users accessing the Image Services software created by **fn_build**.

The serverGroup file is located in the following directory (separated according to platform):

- UNIX: **/fnsw/local/sd/serverGroup**

- Windows: **\fnsw_loc\sd\serverGroup**

The content of a typical serverGroup file is shown below.

```
#CULS, RDB_TYPE, INX, SQI, WQS
63 1 134231042 1 134231063 1 134231066
```

Once the maximum number of users is reached, RPCs for any program in that group will be rejected, and a reject count displayed incrementally. You can track the reject count using PPMOI. The serverGroup file contains at least one line, and each line is structured similar to the example shown below:

max RDB_Type program1 version1 program2 version2 ...

where **max** is the CUL limit, **RDB_Type** is an integer, **program** is the Courier remote program number, and **version** is the Courier program version number.

The example shown specifies a maximum of 50 users for INX, SQI and WQS.

50 1 0x8003402 1 0x8003417 1 0x800341a 1

All numbers are separated by spaces and specified according to C conventions, i.e., 0x# for hex, 0# for octal and # for decimal representations. Comment lines start with either a # or / character.

serverConfig

Each time the Image Services software starts, the serverConfig file is loaded into active memory on the server. It is actually part of the Process Manager Software and is static. The file contains the maximum number of request handlers that can be loaded into memory for each service or procedure. The PPM shared library checks the in-memory serverConfig file to determine the maximum number of request handlers that can be started for each request type.

The serverConfig file is located in the following directory (separated according to platform):

- UNIX: **/fnsw/etc/serverConfig**
- Windows: **\fnsw\etc\serverConfig**

The table below lists the maximum number of request handlers stored in the serverConfig file illustrated later in this section.

System Component	Request Handler	Maximum
Batch services	BESs	8
Cache services	CSMs	12
Index services	INXs	12
Document services	DOCs RMKs DLIs	12 1 5
NCH services	NCHs	12
OSAR services	OSIs OSSs OSCs	3 3 3
Print services	PRIs	12

System Component	Request Handler	Maximum
Backup	BRBs	1
Security services	SECs	12
Restore	BRRs	1

If you have been trained to support your FileNet system, use the PPMOI tool to change the maximum number of request handlers for the in-memory version of the serverConfig file. (See the *System Tools Reference Manual* for more information on using the PPMOI tool.)

The serverConfig file has a unique structure that allows you to easily interpret the current request handler status for each service type. Each request handler entry appears on its own line and consists of seven parameter fields separated by spaces. The example request handler entry below illustrates the typical structure of the serverConfig file entries.

```
BESs      0134231044   1   8   1   0   64
 |            |       |   |   |   |    |
 1            2       3   4   5   6    7
```

- The first (1) parameter field is the name of the request handler process.
- The second (2) parameter field is the remote program number.
- The third (3) parameter field is the remote program version number.
- The fourth (4) parameter field is the maximum number of request handler processes of this type.
- The fifth (5) parameter field is a flag. If the flag is set to 1, the process will be monitored by the FileNet SNMP proxy agent.
- The sixth (6) parameter field is an inactivity interval. If the value is other than zero, the number is the number of seconds after which

the request handler process will free certain resources (for example, log off of Oracle).

- The seventh (7) parameter field is the queue connection size for request handler processes.

The content of a typical serverConfig file is shown below.

```
...
NCHs     02         2 12 1 0 64
CSMs     0134231040 1 12 1 0 64
DOCs     0134231041 1 12 1 0 64
INXs     0134231042 1 12 1 0 64
PRIs     0134231043 1 12 1 0 64
BESs     0134231044 1 8  1 0 64
PSMs     0134231050 2 4  0 0 64 # Do not increase number of PSMs
RMKs     0134231051 1 1  0 0 64 # Do not increase number of request handlers
OSIs     0134231052 1 3  0 0 64
DLIs     0134231053 1 5  0 0 64
OSSs     0134231054 1 3  1 0 64
DLSs     0134231055 1 8  0 0 64
BRBs     0134231056 1 1  0 0 64
BRRs     0134231057 1 1  0 0 64
TAPs     0134231058 1 1  0 0 64
dbintf   0134231059 1 12 0 0 64
SECs     0134231060 1 12 1 0 64
SKFs     0134231061 1 12 0 0 64
SQIs     0134231063 1 10 1 0 64
FILs     0134231064 1 6  1 0 64
WQSs     0134231066 1 12 1 0 64
ARIs     0134231068 1 1  0 0 64
CPTs     0134231069 1 32 0 0 64
RJEs     0134231070 1 3  0 0 64
OSCs     0134231072 1 3  0 0 64
FSMs     0134231073 1 6  0 0 64
CDBs     0134231074 1 6  0 0 64
BRDs     0134231075 1 12 0 0 64
BRTs     0134231076 1 12 0 0 64
BRRMKs   0134231077 1 12 0 0 64
VWSs     0134231078 1 64 0 0 64
...
```

ssn

The ssn file contains the current system serial number. On UNIX systems, the serial number is based on the main processor identification number. However, on Windows systems, the serial number is based on the primary network interface card (NIC).

CAUTION Never modify the contents of the ssn file. The Image Services software will not operate properly if the incorrect system serial number is entered into the ssn file.

The ssn file is located in the following directory (separated according to platform):

- UNIX: **/fnsw/local/ssn**
- Windows: **\fnsw_loc\ssn**

Enter **ssn** at the system/command prompt to view the system serial number output. The content of a typical ssn file is shown below.

```
2620417
```

tapeconfig

The tapeconfig file is read by tape services software. The tapeconfig file is located in the following directory (separated according to platform):

- UNIX: **/fnsw/local/sd/1/tapeconfig**
- Windows: **\fnsw_loc\sd\1\tapeconfig**

This file contains the tape device configuration, specifying the default backup tape device. A separate tapeconfig file exists on each server that has a tape drive. The content of a typical tapeconfig file is shown below:

```
TAPE_CONFIG 10
backup tape1:earth:FileNet
tape1:earth:FileNet,5,0,(1,/dev/rmt/0m,/dev/rmt/0mn),\
(2,/dev/rmt/0m,/dev/rmt/0mn)
```

The file has the following structure for the parameters of the defined tape drive:

- One line containing the format level
- One line containing the name of the default backup tape drive
- One or more lines describing the name, type, capacity, densities, and device names of the various drives. (Type 1 is reel-to-reel, type 2 is cartridge. Low and high densities for cartridge tapes are specified with the values 1 [low] and 2 [high].)

Note Backslashes (\) indicate continued lines.

setup_config

Created by the fn_setup program, the setup_config file is used to determine the name of the DBA group.

The setup_config file is located in the following directory (separated according to platform):

- UNIX: **/fnsw/local/setup_config**
- Windows: **\fnsw_loc\setup_config**

The content of a typical setup_config is shown below.

```
RDBMS_TYPE=1
RDBMS_UID=oracle
RDBMS_GID=dba
RDBMS_HOME=/usr/oracle
```

3
Applications

This chapter lists a number of FileNet application programs used by various components of the FileNet software. All programs that belong to the FileNet software are stored in the bin directory under the fnsw file system/directory for both the UNIX and Windows platforms:

- UNIX: **/fnsw/bin**
- Windows: **\fnsw\bin**

CAUTION Do not install non-FileNet application programs in the /fnsw/bin subdirectory.

A FileNet server receives many different types of requests from its clients. A client may request to scan a batch of documents or store index values in a database. In order to efficiently manage the task of satisfying client requests, the IS software is broken down into separate tasks or services. Each IS service has a corresponding request handler. When a request arrives at a server, COR_Listen deciphers part of the message to find out which service will handle the request. It passes the request to the appropriate request handler (stub).

Application Program Descriptions

An application program with a name including a small “s” is a stub or request handler. For example BESs stands for Batch Entry services stub. A stub and a request handler are the same thing.

Backup/Restore

backup_daemon

The backup daemon program monitors backup partitions and makes periodic copies to avoid an overflow of data (Backup).

BRBs

The BRBs program handles I/O to a remote tape drive when backup is performed (Backup). The number of BRBs request handlers that can be loaded into memory is dictated by the BRBs entry in the server-Config file.

BRRs

The BRRs program handles the I/O to a remote tape drive when a restore is being performed (Restore). The number of BRRs request handlers that can be loaded into memory is dictated by the BRRs entry in the serverConfig file.

Batch Entry Services

BESs

The BESs program is the request handler that commences I/O with the batch tables of the Transient database on behalf of a request from a remote server. It is started up by the COR_Listen program when a request for batch table information is received over the network from a client. The number of BESs request handlers that can be loaded into memory is dictated by the BESs entry in the serverConfig file.

Cache Services

CSMs

The CSMs program is the request handler that commences I/O with the cache tables of the Transient database on behalf of a request from a remote server. It is started as a result of client requests for I/O with the csm tables. The number of CSMs request handlers that can be loaded into memory is dictated by the CSMs entry in the serverConfig file.

CSM_daemon

The CSM_daemon program is started by ds_init at FileNet Image Services startup. This program also runs on Application servers with Cache services. Objects (documents) remain in the page (retrieval) cache for a configurable period of time. They are then eligible for deletion if the space is needed for another object. The daemon manages the aging process of objects. It also monitors the free space and performs the deletions of eligible objects.

Document Services

bes_commit

The bes_commit program is the batch committal daemon. It is started up when the FileNet Image Services software is started. It checks the queue field of the batch_stat_hdr table every sixty seconds to see if a batch is ready for committal.

Having found a batch that is ready for committal, the daemon uses BES and BESl shared libraries to open the batch and perform an asynchronous committal.

If the cataloging is successful, the batch records are then deleted from the Transient database. If not, an error is posted, the next_phase field of the batch_dyn_hdr table is set to “recommit”, and the phase_status field is set to “has errors”.

DLIs

The DLIs program begins I/O to the docs, scalar_numbers, and cluster_map tables of a DocLocator database on behalf of a request from a remote server. The number of DLIs request handlers that can be loaded into memory is dictated by the DLIs entry in the serverConfig file.

DLSs

The DLSs program begins I/O to the docs and cluster_map tables of a DocLocator database on behalf of a background job request from a remote server (optical disk to disk copy, disk import, etc.).

DOCs

The DOCs program is the request handler that runs on the DocLocator server and implements entry points to Document services on behalf of a request from a remote server. The number of DOCs request handlers that can be loaded into memory is dictated by the DOCs entry in the serverConfig file.

ds_init

The ds_init (Document services initialization process) program links to all the abstracts in Document services, builds the surface records, rebuilds all the demand queues and the committal queue, then starts all the processes listed in the as_conf.g file.

ds_notify

The ds_notify program notifies the client when a retrieval has been completed.

fbc_commit

The fbc_commit program processes fast batch committals. This program doesn't need to run if the server has a storage library.

OSIs

The OSIs program is the server process for OSI (receives requests from OSIr). It performs Permanent database I/O on an Storage Library server on behalf of a request from a DocLocator server. The number of OSIs request handlers that can be loaded into memory is dictated by the OSIs entry in the serverConfig file.

OSSs

The OSSs program performs permanent database I/O on a Storage Library server for background jobs (disk to disk copy, disk import, etc.) on behalf of a request from a DocLocator server. The number of OSSs request handlers that can be loaded into memory is dictated by the OSSs entry in the serverConfig file.

RMKs

The RMKs program is the server process for Remote MKF interfaces or RMK (receives requests from RMKr). It updates the Permanent database on behalf of a remote server when very low volume transactions are being performed. The number of RMKs request handlers that can be loaded into memory is dictated by the RMKs entry in the serverConfig file.

rmt_commit

The rmt_commit program is responsible for committing documents. The program is started by the ds_init process. Multiple instances of this program may run as determined by the number of remote domains that are configured for remote committal.

sas

The sas program is used to determine if session time-outs are working properly.

Fax Services

FSMs

Handles requests for Fax servers

File Services

FILs

FILs is the File services server process.

Index Services

INXbg

The INXbg program is a background process that keeps INXl and INXD in memory and handles data dictionary updates. This program normally runs only on the server(s) where index services and bes_ commit are running and where dictionary builds and abstract linking are particularly time-consuming.

INXs

The INXs program is a server stub program. This program receives remote procedure calls, does server-side session handling, and calls INXl on behalf of a remote client. The number of INXs request handlers that can be loaded into memory is dictated by the INXs entry in the serverConfig file.

INXu

The INXu program runs the server update process. To avoid interference with on-going queries, a separate process is sometimes needed to perform updates and commit transactions.

MKF Database Services

MKF_clean

The MKF_clean program preserves MKF database integrity in case of abnormal termination.

Network Clearinghouse Services

NCHs

The NCHs program handles I/O to the NCH database on behalf of requests from remote servers. The number of NCHs request handlers that can be loaded into memory is dictated by the NCHs entry in the serverConfig file.

NCH_daemon

The NCH_daemon is a program that only runs on the Root server and that transmits the Root server's protocol ID as each remote server is booted. There will be an NCH_daemon running on the Root server for each type of network protocol in use (for example, TCP/IP).

The NCH_daemon process listens for requests for NCH services and responds to these requests with a packet containing the network address at which the NCH database server can be reached.

Print Services

PRI_check

The PRI_check program checks to see if any printers that were marked as down are now responding. This function is done separately from the PRI_daemon because it takes about a minute for the network software to time out.

PRI_daemon

The PRI_daemon program handles time-related tasks. This program activates print requests which have been set to print at a certain time and deletes old print requests from the Transient database when these requests have been completed.

PRI_daemon also de-queues records from the status queue and updates Print services data structures as appropriate. The status queue contains responses from the dtp process indicating that a document retrieval is complete, and responses from the print server indicating that a page has been printed.

PRI_notify

The PRI_notify program sends the print complete status from Print services back to client.

PRIs

The PRIs program receives the client request for Print services and puts requests into the Transient database on behalf of a request from a remote server. This program modifies and cancels print requests, and returns status on the activities of Print services. The number of PRIs request handlers that can be loaded into memory is dictated by the PRIs entry in the serverConfig file.

PRI_worker

The PRI_worker program sends the pages to be printed into the appropriate print cache and then sends a request to the printer imaging software. Up to 12 PRI_worker programs can be started to handle system printing needs. PRI_worker is not dedicated to any one printer; each PRI_worker can handle requests for any printer.

PSMs

The PSMs program runs on the Print server and handles incoming documents to be printed

Security Services

SEC_daemon

The SEC_daemon program removes security logging information from the QMA queue and writes that information to the security log file. This program is started by the FileNet software initialization script and runs only on the Root server.

SECs

The SECs program services remote procedure calls (RPCs) to Security services, formats a response network message, sends a message back to the requesting client, and passes the request to the Security services target abstract, SECI (which provides the actual functionality. This program decrypts information received during user logon. The number of SECs request handlers that can be loaded into memory is dictated by the SECs entry in the serverConfig file.

Storage Library Services

antcopy

The antcopy program performs the document copy background process. This program copies a document's annotations from one optical disk to another.

del_commit

The del_commit program migrates documents to optical disk on a delayed basis. This program only runs on the document locator server.

doccopy

The doccopy program copies a document from one optical disk to another.

docimport

The docimport program imports document information and indexes from optical disk to MKF/RDBMS databases.

dsched

the dsched program schedules/controls the optical disk drives and the movement of the optical disks within each storage library.

dtp

The dtp program performs the data transfer process. This program handles the transfer of documents to and from optical disk.

dtp_tran

The dtp_tran program calls Cache services to update the Transient database when the writing of a document completes.

opendocs

The opendocs program determines statistics about optical disks. This program reports on the type, surface, open, closed, deleted, total, and percent of open statistics for an optical disk's documents.

OSCs

The OSCs program handles requests for Storage Library services from Document services. The number of OSCs request handlers that can be loaded into memory is dictated by the OSCs entry in the serverConfig file.

osi_migrate

The osi_migrate program handles interaction with the Permanent database in regard to remote committal.

sortod

The sortod program is used to sort binary files. This program is used in such functions as an optical disk to disk copy.

WorkFlo Queue Services

WQSs

WQSs is the server process for WorkFlo Queue services. It receives requests form WorkFlo Queue services.

Others

CDBs

The CDB program handles requests to the configuration database files.

CORs

The CORs program handles requests to transmit/receive database information and images between servers.

COR_Listen

COR_Listen is the daemon that must be running at the server to accept an incoming request from the client. COR_Listen transfers the connection that it opened when it received the request from the network to the available server stub. COR_Listen does not start the request handler itself but rather calls PPM to do the job.

For more information on PPM, go to **page 77**.

dbintf

The dbintf program performs I/O with the index database on behalf of a remote server when creating an index field or document class.

SKFs

The SKFs program performs I/O with the doctaba table of the Index database when the dbverify utility is run from a server other than the Index server.

SQIs

The SQIs program performs SKF database I/O for requests from remote servers.

TAPs

The TAPs program performs I/O on a server that is interfacing with a remote tape drive.

4

Shared Libraries

This chapter describes a number of different shared libraries used by various components of the FileNet software. The entries in this chapter include shared library paths and naming conventions, followed by shared library definitions.

Note Support personnel use the terms **abstract** or **shared object** when referring to a shared library. These are older terms but still appear in many tools.

A shared library is a collection of functions (for example, open file, close file, read form file, etc.). A running process calls the shared library when it needs to perform a function stored in the shared library. The system loads the functions into memory and the process accesses the code from there.

Note The Windows platform uses shared libraries called DLLs (dynamically linked library).

The programs used by the Image Services software rely on shared libraries to perform their functions. If an error occurs, the shared library usually reports the problem. The shared libraries detailed in this chapter are separated according to the services.

Shared Library Locations

All shared libraries that belong to the FileNet software are stored in the shobj directory under the fnsw filesystem. All shared libraries used by the Image Services software are located in the following path:

- UNIX Platforms: **/fnsw/lib/shobj <shared library name>**
- Windows: **\fnsw\lib\shobj <shared library name>**

  (where **<shared library name>** is the name of the shared library being called.)

Each platform uses a unique convention for shared library names. The table illustrates an example of the shared library naming conventions, according to platform, for the BES shared library.

Platform	Name
AIX/6000	BES
HP-UX	libBES.sl
Solaris	libBES.so
Windows	BES.dll

Shared Library Descriptions

Shared libraries are used by distributed applications and are often separated into remote (client) and local (server) objects.

Note Some shared library names include r or l (for example, BESr or BESl). The r indicates that the shared library is used remotely, and the l indicates the shared library is used locally. (Other letters appended to the shared library names are more or less arbitrary and do not indicate the location of the shared library execution.)

All Services

COR

COR stands for Courier. The shared library is executed by a program to transport an image or database information across the network from the server to the client. A request handler calls COR and hands off the server response. COR is involved in getting the server response packed up to travel over the network back to the client.

PPM

PPM is the Protocol Process Manager which reads the serverConfig file to find out the maximum number of request handlers that can be loaded into memory. PPM is the shared library that reports an error if a request handler cannot be started. PPM starts a request handler if a server stub is not available in memory. Once PPM either locates an available request handler or starts a new one, it advises COR_Listen that the correct stub is now available. PPM is only a shared library or set of functions. It requires a program to call it. For more information on COR_Listen, go to **page 73**.

Batch Entry Services

BES

BES is used by the bes_commit program to interface with the batch tables. For instance, the BES code executes the functions needed to create, open, close, enqueue, dequeue, commit, delete, update, and find a batch.

BESI

BESI is used by the BES shared library to perform I/O on the fields of the batch tables. Most batch service errors are reported by BESI.

Cache Services

CSM

CSM performs I/O with the csm tables of the Transient database.

CSMI

CSMI assists CSM in handling I/O with the csm tables of the Transient database and all caches.

Document Services

ASH

ASH assists in the handle of caching which is used to associate an Image Services handle with a service handle and vice versa.

BKG

BKG manages the background requests. It handles the MKF background requests table (bkg_requests), which is in the Transient database, and also starts and stops processes which are performing background requests (e.g., an optical disk to disk copy).

CKS

CKS computes a checksum on the data indicated.

CMT

CMT is the Document services shared library to commit documents on the document server.

CNF

CNF is the configuration shared library for Document services. It accesses the as_conf.g and as_conf.s files. Its entry points provide things like the number and names of the retrieval cache partitions, optical drives, storage libraries, etc.

CNT

CNT compiles performance statistics for Document Services. It includes counts for surface records, migrates from optical disk, pages committed, documents committed, fast batches committed, etc.

DBL

DBL is the Document services database logon shared library for MKF. It contains entry points for Document services to log on or off Permanent and Transient databases.

DBP

DBP controls memory buffers which are used for reading document headers and document pages. The size and number of buffers is controlled by the configuration file (see CNF). Manages images in memory being transferred to/from cache.

DIA

DIA contains routines which import document headers from optical disk into the MKF and RDBMS databases.

DIG

DIG is a diagnostic interface. It reserves a storage library for use by an instance of Document services.

DLI

DLI is the document locator interface shared library. It is used by the Storage Library servers to request actions by the document locator.

DLIl

DLII performs the actual Permanent database I/O on the Storage Library server.

DLIo

DLIo is the locking shared library for DLI. It requests information for a Storage Library server from a DocLocator server.

DLS

DLS is the document locator service shared library. It provides document locator functions for the background tasks: document copy, document import, and annotation copy.

DLSI

DLSI performs the actual Permanent database I/O on the DocLocator server for the document locator service.

DLSr

DLSr requests a background job from a DocLocator server.

DOC

DOC is the shared library interface component for Document services. It processes all calls to the rest of the service. It is used to initiate I/O with the docs table of a Permanent database.

DOCa

DOCa implements Document services on an archive Image Services system.

DOCb

DOCb is Document services fast batch commit.

DOCI

DOCI is the target shared library. It implements the actual service—making DBMS calls. DOCI is the heart of Document services. It performs the actual I/O with the docs table of a Permanent database with the help of the DT shared library.

DOCp

The DOCp module implements Document services on a portable database cache (PDB).

DSA

DSA is the Document services shared library to interface with the annotations table.

DT

DT manages insertions, deletions, and updates to the docs table on the Storage Library server.

FDT

FDT accesses the family_disk table of the Permanent database.

FLT

FLT manages the family_locator table which is used to find the proper Storage Library server ID given a family ID. Callers of this shared library must log on to the Document Locator Permanent database using RMK, and use RMK for all database transactions using this shared library.

HLT

HLT handles high level tasks. It reassigns read/write requests, enables/disables disks, and updates optical disk families and SRF and ODT records.

IS

The IS module contains routines to facilitate access to an Image Service. It contains handle management routines, and also some miscellaneous functions.

MKF

MKF establishes contact for any program needing access to an MKF database.

OSI

OSI is the Storage Library server interface. It is used by the DocLocator and the Storage Library servers to request actions of a specified Storage Library server.

OSII

OSII is called by the OSIs request handler to perform Permanent database I/O on the Storage Library server.

OSIr

OSIr is called by the OSI shared library to request information from a Storage Library server.

OSS

OSS, the Storage Library server service shared library, provides Storage Library server functions for the background tasks: document copy, document import, and annotation copy.

OSSI

OSSI is called by the OSSs request handler to perform Permanent database I/O on the Storage Library server.

QMA

QMA, the Queue Manager shared library, manages memory queues for requesting Document services shared libraries.

RFT

RFT is the Document services shared library to interface to the remote_family table.

RMK

RMK is the remote MKF interface. It is used by the DocLocator and the Storage Library servers to request low volume MKF operations remotely.

RMKr

RMKr is used by remote servers to request a low-volume MKF database exchange.

SAS

SAS, the Session Application Service shared library, assists the various services (DOC, INX, CSM, PRI, etc.) in manipulating sessions. A "session" is called active when a process has logged on to a service, and SAS has stored information about that logon on a remote system. SAS opens, closes, and deletes sessions for all of the services.

SDC

SDC is the interface for Single Document Committal (SDC) which provides for the committal, deletion, and security update of documents. All these functions would normally require calls to multiple services (e.g. INX and DOC) and/or calls to BES (which would require a series of calls to manage a batch). These calls are intended to simplify the interface to application services for clients which require simple functionality.

SRF

SRF, the Surface Record Format shared library, handles manipulation of surface records. There is one surface record for each optical disk surface either in the storage library, or which has pending demands. The surface records are memory structures which parallel the Optical Disk Table (ODT) records.

Index Services

INX

INX decides whether to make local or remote calls to the rest of the service.

INXa

INXa implements the actual index service in the archive database.

INXD

INXD reports if the RDD is empty or corrupted.

INXl

INXI implements the actual service—making RDBMS calls.

INXr

INXr performs RDD caching, Courier RPCs, and client-side session handling.

Network ClearingHouse Services

NCH

NCH initiates contact with the NCH database.

NCHI

NCHI performs I/O with the NCH database.

NCHr

NCHr requests access from the NCH database for non-root servers.

SLMI

SLMI directly accesses encrypted license key information in the NCH database.

Print Services

PRI

PRI initiates a validation of print options and the processing of fax requests.

PRII

PRII performs a print option validation and processes fax requests.

Security Services

SEC

SEC accesses information in the security database.

SECI

SECI performs security DB searches, updates, deletes, adds, licensing checks.

SECr

SECr processes requests for Security services for a client (non-root) server.

SMM

SMM allocates/de-allocates chunks of shared memory on a per table basis.

SSD

SSD provides interface between SECl and the security database.

SSU

SSU encrypts/decrypts data, performs object name conversion.

Storage Library Services

ARM

ARM contains routines that control mechanical functions of a storage library.

CLT

CLT is the shared library for the DS III Cluster Map Table.

DIS

DIS assigns requests to optical disk surfaces.

ODF

ODF inspects/modifies optical disk descriptor without using record structure.

ODS

ODS provides functions that access the family_disk table on the Storage Library server.

ODT

ODT manages optical disk table surf_dyn_info and surf_stat_info.

ODU

ODU contains routines to access the optical disk in a formatted manner.

ODX

ODX contains routines to access the optical disk in an unformatted manner.

OFA

OFA writes COLD documents to optical disk families.

OPM

OPM transmits and receives messages for the Storage Library Control program.

OSA

OSA allows program to get/set status about storage libraries' slots, drives, etc.

SLT

SLT manages the surf_locator table.

SNT

SNT maintains the scalar_numbers table of the Permanent database.

WRT

WRT manages the write requests table.

System Configuration

CDB

CDB accesses IMS_xxx.cdb file. It creates the initial configuration database file.

Miscellaneous System V

ORA

ORA performs I\O with Oracle database used by GDB Oracle compatibility layer.

System Messages

ERM

ERM is used by any program that requires access to the error message file.

5

Permanent Database Tables and Contents

Document and Storage Library services access the information stored in the tables of the Permanent database. The Permanent database contains information about each document that is written to optical disk as well as about the optical disks themselves. The Permanent database acts as a directory for the storage library. Many tables in this database are referenced during the document retrieval process.

This chapter discusses the following Permanent database tables:

- annotations
- cluster_map
- docs
- family_disk
- family_locator
- od_stats
- remote_family
- scalar_numbers
- surf_dyn_info
- surf_info
- surf_locator
- surf_stat_info

Tables and Contents

Each table (and its contents) in the Permanent database are discussed in this section; some of the tables contain examples of the contents. You can use the MKF_tool to view these tables and contents directly.

annotations

This table maintains a record of each annotation and its associated document and page. The actual annotation text is stored in a hexadecimal format.

Field Name	Field Description
annot_key	Unique search key comprised of doc_id, page, ssn, and annot_id keys - (group).
doc_id	The id number of the document containing the annotation.
page	The page number of the document.
ssn	The system serial number that the document was originated on.
annot_id	System assigned id number for the annotation relative the id, page number and ssn of the document. The first annotation of a page always has a value of 1 in this column.
create_date	The creation date of the annotation.
modify_date	The last time the annotation was modified.
security_len	The number of bytes in the security_info column.
security_info	Encoded access restrictions - read, write, execute/append.
capability	Capability of user updating the annotation. Null if no update is in progress.
data_len	The number of bytes in the data column.
data	The annotation (in hexadecimal). May be up to 800 bytes large.

The following example illustrates a typical record in the **annotations** table:

```
doc_id............8639401  page................   1
ssn............2620417
annot_id..............1  data_len..............27
create_date.....752962296 => 99/11/10 12:11:36
modify_date.....752962296 => 99/11/10 12:11:36
data          [0]....0x040002000101000400000000020004000000000c0001020300
```

cluster_map

This table maintains a record of each cluster id and the optical disk surface id associated with it. The table is referenced during the act of committal. Data I/O with this table is primarily accomplished through the use of the CLT shared library.

Field Name	Field Description
cluster_key	Unique search key comprised of cluster_space and cluster_id keys - (group).
cluster_space	0,1,.... Identifies the cluster number of the surface.
cluster_id	The id number of the cluster. It is a 6 byte id number derived from the index_cluster table of the index database.
surface_id	The current optical disk surface that is involved in the cluster operation.

The following example shows a typical record from the **cluster_map** table:

```
cluster_space  cluster_id      surface_id
-------------  ----------      ----------
            1 0xe0cae8cae400         3008
```

docs

This table maintains a record for each document id number and its associated optical disk surface. The table is updated during committal and also when writing to optical disk. The table is referenced whenever a document is retrieved.

Field Name	Field Description
doc_id	Unique search key - document id number.
status_1	null or 0 = normal 1 = bad (unreadable) 2 = deleted 3 = not migrated to optical disk
status_2	null or 0 = normal 1 = bad (unreadable)
pages	The number of pages in the document.
contig_sectors	The number of contiguous optical disk sectors in the last page of the document.
surface_id_1	The surface number of the “family” disk for this document.
offset_1	The sector of the “family” disk containing this documents long descriptor file.
surface_id_2	The surface number of the “tranlog” disk for this document.
offset_2	The sector of the “tranlog” disk containing this documents long descriptor file.
security_len	The number of bytes in the security_info column.
security_info	Encoded access restrictions - read, write, execute/append.
orig_doc	Unique search key comprised of orig_ssn and orig_doc_id keys - (group). This number is NULL if the documents are local.
orig_ssn	System serial number of the system the document was originally from. This number is NULL if the documents are local.
orig_doc_id	Document id number on original system. This number is NULL if the documents are local.

Field Name	Field Description, Continued
cache_id	Cache id when not on optical disk.
back_contig	Number of contiguous sectors (page 0 last).
duration	Time at which object is to be written to optical disk. This field was added to accommodate delayed migration.

The following example shows a typical record from the **docs** table.

```
doc_id.............101200  pages................1  surface_id_1......3004
offset_1...........101706  surface_id_2......3006  offset_2........234756
security_len............3  orig_ssn.........13476  orig_doc_id.....114537
security_info [0]....0x8181d8
```

family_disk

This table keeps a record of each optical disk family on the system. It is referenced during committal. Data I/O with this table is primarily accomplished through the use of the FDT shared library.

Field Name	Field Description
current_surfs	Current optical disk surface number available for a write operation.
des_cur_surfs	Desired number of current optical disk surfaces that are available for a write operation.

Field Name	Field Description, Continued
disk_type	0 = unspecified disk type 1 = Maxell 12” 2.6 GB WORM 2 = Maxell 5” 600 MB WORM 3 = Maxell 5” 600 MB Erasable 4 = Philips 12” 2 GB WORM 5 = Maxell 12” 7 GB WORM 6 = Philips 12” 5.5 GB WORM 7 = no longer in 3.6 Database Maintenance 8 = no longer in 3.6 Database Maintenance 9 = Standard 5” 1.3 GB Erasable 10 = Standard 5” 1.3 GB WORM 11 = Philips 12” 12 GB WORM 12 = Standard 5” 2.6 GB Erasable 13 = Standard 5” 2.6 GB WORM 14 = IBM 5” 2.6 GB Ablative WORM 15 = Standard 5” 5.2 GB Erasable 16 = Standard 5” 5.2 GB WORM 17 = IBM 5” 5.2 GB Ablative WORM 18 = Plasmon 12” 30 GB WORM 19 = Standard 5” 9.1 GB Erasable 20 = Standard 5” 9.1 GB WORM 21 = MSAR 1 GB 22 = MSAR 2 GB 23 = MSAR 4 GB 24 = MSAR 8 GB 25 = MSAR 16 GB 26 = MSAR 32 GB
family_id	System-assigned optical disk family identification number.
family_name	User-assigned optical disk family name.
future_surfs	Future optical disk surface number reserved for this family.
interleaved	Write this number of side A’s before side B’s.

Field Name	Field Description, Continued
is_primary	Is this the primary family? 0 = False (transaction) 1 = True (primary)
next_ordinal	Not used.
no_migrate_fb	This value is FALSE when at least one optical library is present on the server, and the write requests will be executed to migrate the documents to the optical library. This value is TRUE when the system has no optical storage and the write requests will not be completed.
num_cur_surfs	Actual number of current optical disk surfaces that are available for a write operation.
num_futr_surfs	The number of future surfaces reserved for this optical disk family.
preferred_osar	ID number (a through h), of the storage library that contains the optical disk called out in the “current_surfs” column.
prev_0_surfs	Previous surfaces for cur_surf (current surfaces).
prev_surfs	The optical disk surface number previously written to. (Used when interleaving).
sides	1 = single sided disk 2 = double sided disk
tran_families	Family ids of the tranlogs for the family. If all zero, no tran logging.

The following example shows a typical record from the **family_disk** table.

```
family_id.............41  interleaved.........1  des_cur_surfs........0
num_cur_surfs...........0  current_surfs........0  prev_surfs...........0
preferred_osar........""  num_futr_surfs.......0  future_surfs.........0
disk_type..............2  is_primary...........1
family_name..."CloseDocs"
tran_families [0]....0,0,0,0,0,0,0,0
```

family_locator

This table is used in a multiple Storage Library server environment. The records within the table associate optical disk family id numbers with a Storage Library server. It is referenced during the act of committal. Data I/O with this table is primarily accomplished through the use of the FLT shared library.

Field Name	Field Description
family_name	Unique search key - user-defined name for an optical disk family.
family_id	System assigned number for an optical disk family.
server_ids	System assigned Storage Library server id number. Used to designate which Storage Library server relates to a given optical disk family.
num_ids	The number of Storage Library servers that use the disks of a given optical disk family.

od_stats

This is a statistical record table that maintains a log of optical disk write errors. Data I/O with this table is primarily accomplished through the use of the ODS shared library.

Field Name	Field Description
surface_id	Optical disk surface number that incurred an error.
osardrive	Unique search key comprised of osar_num and drive_num keys - (group).
osar_num	The number of the storage library that incurred the error.
drive_num	The optical drive which incurred the error.
num_rewrites	The number of 1k sectors that incurred an error during a write operation.
last_day_sec	The date and time of the last I/O error that occurred.

Field Name	Field Description, Continued
num_elt	The number of elements expressed in the days_errors and total_errors columns.
days_errors	Each number (also known as an element), in this column correlates to a specific type of optical I/O error. For each instance that a particular type of error occurs, the element increments. The value for each element is equal to the total number of errors that occurred since midnight of the day expressed in the last_day_sec column.
total_errors	This column is similar to days_errors except that the error counts shown are for the lifetime of a particular drive or surface, not just since midnight of a given day.

remote_family

The presence of records in this table indicates that committal operations must be done on the remote domain and family name indicated in addition to the local family.

Field Name	Field Description
fam_key	Unique search key comprised of family_id and rec_num keys - (group).
family_id	ID of local family.
rec_num	Number of this entry in the table.
rmt_domain	Name of remote domain.
rmt_org	Name of remote organization.
rmt_fam_name	Name of remote family (may be different than local family, but must be a primary family ID).

The following example shows a typical record from the **remote_family** table:

```
family_id.................52  rec_num.............1
rmt_domain..........."costa6"
rmt_fam_name....."Clustering"
```

scalar_numbers

This table maintains a record of the next available image id number (for scanning), optical disk surface number (for writing to optical), and background job request number (for optical disk copying).

Field Name	Field Description
rec_num	Unique search key. The value is always 0.
next_doc	The next available image id number to be used for scanning.
next_surface	The next available optical disk surface number that can be assigned.
next_bkg_job	The next available job id number that can be assigned to a background job (optical disk copy, annotation copy, optical disk import).
db_version	Version of the MKF database.
boot_num	Number of times the system has been booted.
char_set	The character set of this database.

The following example shows a typical record from the **scalar_numbers** table.

```
rec_num.............0  next_doc.......8752760  next_surface......3338
next_bkg_job.........132  db_version...........4  boot_num.........1477
char_set.............11
```

surf_dyn_info

This table keeps a record of available space on each optical disk surface. It is referenced and updated when writing to optical disk. Data I/O with this table is primarily accomplished through the use of the ODT shared library.

Field Name	Field Description
surface_id	Unique search key - optical disk surface id number.
next_sector	Next available sector on the optical disk surface that can be written to. This is also known as the high-water mark.
num_act_docs	Number of retrievable documents already on optical disk surface. See note below.
num_del_docs	Number of non-retrievable (deleted) documents on optical disk surface.
num_clusters	Number of clusters involved with this optical disk surface.
last_desc_sect	This is the last sector of the data area, of the last extent, of the short descriptor file on the optical disk surface. The value of this column will be 0 if no short descriptor file has been created yet.
nxt_short_desc	Next available sector in the short descriptor file. The value of this column will be 0 if no short descriptor file has been created yet.
num_unwrt_desc	Number of attempted but unwritten sectors in the short descriptor file. The value of this column should always be 0. Any other number indicates that some type of interruption or crash occurred while a short descriptor file was being written.
short_doc_ids	The document id numbers of the documents that have been written to disk, but have not yet had a short descriptor file written.
short_sects	The sector numbers of the long descriptor files that have not yet had a short descriptor file written.
num_used_sects	The number of 1024 byte sectors occupied on optical disk surface.
num_pages	Total number of document pages on the surface.

Field Name	Field Description, Continued
next_fid_sect	The next available sector that can be written to in the optical disk directory file.
last_fid_sect	The last available sector that can be written to in the optical disk directory file.
cur_fid_off	Offset in the directory of the current entry.
inbox_priority	Priority which keeps the surface in the storage library. This is only set for side 1 surfaces.

The following example shows a typical record from the **surf_dyn_info** table.

```
surface_id..........3001  next_sector.........0  num_act_docs........0
num_del_docs............0  num_clusters.........0  last_desc_sect.......0
nxt_short_desc..........0  num_unwrt_desc.......0  num_used_sects.......0
num_pages...............0  next_fid_sect........0  last_fid_sect........0
cur_fid_off.............0  inbox_priority.....128
short_doc_ids
[0].0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
              [29]...0,0,0
short_sects
[0].0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
              [29]...0,0,0
```

Note The num_act_docs value refers to the number of active documents on a given surface. An active surface is the primary surface and the active tranlog. Images on either of these active surfaces can be directly accessed by IS. The num_act_docs value for the primary surface and the active tranlog will be incremented after a successful image committal.

Alternate (or secondary) tranlogs are not considered active. Images stored in these alternate tranlogs must be imported into the system in

order to access these images. If your system is set up to use multiple tranlogs, only one of them contains active documents and the last tranlog selected is considered to be the active tranlog. The num_act_docs value for any alternate tranlog(s) is not incremented after a successful image committal.

surf_info

This is not a database table. It is an alias (view) used by MKF_tool to retrieve information from the **surf_dyn_info** and **surf_stat_info** tables. For example, the command below retrieves all of the information from both the surf_dyn_info and surf_stat_info tables concerning optical disk surface number 3005.

select surf_info * where surface_id = 3005

surf_locator

This table is used in a multiple Storage Library server environment. The records within the table associate optical disk surface id numbers with a Storage Library server. It is referenced during committal and document retrieval. Data I/O with this table is primarily accomplished through the use of the SLT shared library.

Field Name	Field Description
surface_id	Optical disk surface id number.
server_id	System assigned number of the Storage Library server controlling the library that contains this optical disk.

Field Name	Field Description, Continued
orig_surf	Unique search key comprised of orig_ssn and orig_surf_id keys - (group).
orig_ssn	If imported, the original system serial number that this disk came from.
orig_surf_id	ID of the original surface. This column is used in conjunction with “orig_ssn” for certain search operations.
fam_id	Family ID of the surface.
disk_type	0 = unspecified disk type 1 = Maxell 12” 2.6 GB WORM 2 = Maxell 5” 600 MB WORM 3 = Maxell 5” 600 MB Erasable 4 = Philips 12” 2 GB WORM 5 = Maxell 12” 7 GB WORM 6 = Philips 12” 5.5 GB WORM 7 = no longer in 3.6 Database Maintenance 8 = no longer in 3.6 Database Maintenance 9 = Standard 5” 1.3 GB Erasable 10 = Standard 5” 1.3 GB WORM 11 = Philips 12” 12 GB WORM 12 = Standard 5” 2.6 GB Erasable 13 = Standard 5” 2.6 GB WORM 14 = IBM 5” 2.6 GB Ablative WORM 15 = Standard 5” 5.2 GB Erasable 16 = Standard 5” 5.2 GB WORM 17 = IBM 5” 5.2 GB Ablative WORM 18 = Plasmon 12” 30 GB WORM
num_act_docs	Number of documents for each side.

surf_stat_info

This table keeps a general record about each optical disk surface. Each record contains information like family id, surface id, disk status, the date it was labeled, etc. The table is referenced when an optical disk retrieval is needed. Data I/O with this table is accomplished primarily through the use of the ODT shared library.

Field Name	Field Description
surface_id	Optical disk surface id number.
family_id	System assigned optical disk family id number for the surface.
series_ordinal	Used to preformat disks.
disk_status	0 = normal 1 = surface terminated
write_protect	0 = write enabled 1 = write protected
sides	1 = single sided disk 2 = double sided disk
label_date	When the volume label was written to the optical disk.
full_date	The date and time the surface became full.
archive_date	The date and time the last active document was deleted.
last_disk_sect	The last usable optical disk sector on the surface.
orig_surf	Unique search key comprised of orig_ssn and orig_surfid keys - (group).
orig_ssn	If imported, the original system serial number that this disk came from.
orig_surfid	ID of the original surface. This column is used in conjunction with “orig_ssn” for certain search operations.
old_hw	Value of “next_sector” when the optical disk is converted form 1.8 to 2.2, or null.

Field Name	Field Description, Continued
need_verify	Whether the imported disk was verified.
disk_type	0 = unspecified disk type 1 = Maxell 12” 2.6 GB WORM 2 = Maxell 5” 600 MB WORM 3 = Maxell 5” 600 MB Erasable 4 = Philips 12” 2 GB WORM 5 = Maxell 12” 7 GB WORM 6 = Philips 12” 5.5 GB WORM 7 = no longer in 3.6 Database Maintenance 8 = no longer in 3.6 Database Maintenance 9 = Standard 5” 1.3 GB Erasable 10 = Standard 5” 1.3 GB WORM 11 = Philips 12” 12 GB WORM 12 = Standard 5” 2.6 GB Erasable 13 = Standard 5” 2.6 GB WORM 14 = IBM 5” 2.6 GB Ablative WORM 15 = Standard 5” 5.2 GB Erasable 16 = Standard 5” 5.2 GB WORM 17 = IBM 5” 5.2 GB Ablative WORM 18 = Plasmon 12” 30 GB WORM

The following example shows a typical record from the **surf_stat_info** table.

```
surface_id...........3004  family_id............3  disk_status..........0
write_protect..........0  sides................2  last_disk_sect..287983
orig_ssn............13476  orig_surfid.......3004  disk_type............2
label_date......638480831 => 99/3/26 11:47:11
```

surface_activity

This table stores a surface activity record for each surface. It also records the date when activity occurred on the system. Surface activity logging must be enabled for this information to be recorded.

This table includes special records that track surface activity logging enabling and disabling. These records have a surface id of ASE_MIN_SURF_ID - 1 along with the date on which the record was created. Other fields are null for the logging enable records. When the system comes up, if the logging state is not the same as the last one set in the file, a new record is added.

Field Name	Field Description
surface	The surface id on the system that the disk is from.
act_date	The date activity occurred on the system.
mounts	Mounts of this surface, on this day. 0 = disabled, 1 = enabled
reads	Reads for surface-day.
writes	Writes for surface-day.

lib_surfaces

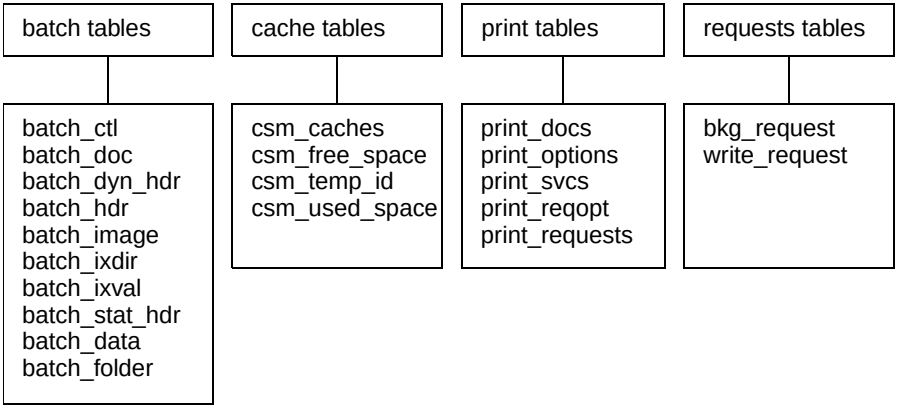
Field Name	Field Description
surface_id	Surface ID
loc_key	Will be set to NULL if out_of_box
library	library Identifier
unit_indicator	Unit indicator: 1= in_slot 2 = in_drive 3 = in_gripper
unit_number	slot, drive, gripper
out_of_box	NULL/0 => in box 1 = Out of box because oddump is converting 2 = Out of box because BKG processing 3 = Out of box because it was ejected
ejected	
name_length	surf_file name length
surf_file	Surface filename -- Must include the absolute path -- Has the .dat extension -- Will be NIL for optical surfaces
disk_type	Disk type

6

Transient Database Tables and Contents

Batch, Cache, and Print services use the tables in the Transient database. The Transient database contains information about documents that are in transition during document entry, retrieval cache, and print cache. Due to its temporary nature, this database is generally assigned a small amount of magnetic disk space.

This chapter discusses the following Transient database tables:



The tables are detailed in sections separated according to the Image Services functions they support.

Batch Tables and Contents

Each table (and its contents) in the Transient database used by the batch service is discussed in this section; some of the tables contain

examples of the contents. You can use the MKF_tool to view these tables and contents directly.

batch_ctl

This table contains the next available batch_id number that can be used. Although it is primarily updated during the batch creation phase, anytime a batch is accessed, the batch_open_id column of this table is updated.

Field Name	Field Description
ctl_index	Unique search key for the table.
batch_id	Batch identification number used in all of the batch tables of the Transient database.
batch_name_id	Next available batch name that will automatically be assigned to a batch if an operator does not choose one.
batch_open_id	The last time that the “batch_id” column was accessed.
folder_id	Next available folder id.

The content of a typical **batch_ctl** table record is shown below:

```
ctl_index    batch_id  batch_name_id  batch_open_id
 ---------    --------  -------------  -------------
         1          11              6      757791177
```

batch_doc

Keeps a record of multiple documents that have been scanned under a single batch id number. Information such as document type, number of pages, phase status, optical disk family number, etc., is recorded. The

record is cleared from this table upon the successful committal and cataloging of the batch.

Field Name	Field Description
doc_index	Unique search key comprised of batch_id and doc_id keys - (group).
batch_id	Identification number of the batch.
doc_id	The document id number.
doc_type	The type of document. 0 = image 1 = text 2 = fill in form 3 = mixed 9 = separator sheet
num_pages	Number of pages in the document.
num_indices	Number of indexing fields associated with this document.
num_reqd_indcs	Number of mandatory indexing fields associated with this document.
phase_num	The phase in which the “phase_error” occurred. Will display 65535 if no error has occurred. See the table **“Document Phases” on page 115**.
phase_error	0 = no error 1 = error occurred while performing this “phase” of the document.
fam_id	Optical disk family id number to which this document belongs.
cluster_space	Identification number of the cluster operation that is involved. This feature is currently not used.
ocr_schema	Reserved for use by OCR.
text_len	The number of bytes in the text_data column.
phase_status	The current status for each of the eleven possible phases that the document could experience. See the table **“Phase Status” on page 115**.
cluster_id	Identification number of the cluster that’s involved. Value is 0 if clustering is not involved.
text_data	Reserved. If the value of text_len = 0, the field is not displayed.

The content of a typical **batch_doc** table record is shown below

```
batch_id..............8  doc_id..............5  doc_type.............0
num_pages.............2  num_indices.........1  num_reqd_indcs.......1
phase_num.............0  phase_error.........0  fam_id...............0
cluster_space.........0  ocr_schema..........0  text_len.............0
phase_status  [0]....0,0,0,0,0,3,0,0,0,0,0
cluster_id    [0]....0x000000000000
```

batch_dyn_hdr

A dynamic record is created within this table for each batch that is created. This record is updated as the status of the batch changes, for example, when it is scanned, indexed, verified, or committed. The record is cleared from this table upon the successful committal and cataloging of the batch.

Field Name	Field Description
batch_id	Unique search key for the table - batch identification number.
exp_pages	Expected number of pages to be scanned in for the batch.
exp_docs	Expected number of documents in the batch.
pages_per_doc	Number of pages/document in batch. 0 = varies per document.
double_sided	0 = single sided pages 1 = double sided pages
verify_images	0 = no image verify necessary 1 = image verification needed
verify_indices	0 = no index verify necessary 1 = index verification needed
committal_type	0 = do not migrate 1 = migrate to optical disk
batch_total	0 = batch totaling unnecessary 1 = batch totaling needed

Field Name	Field Description, Continued
this_phase	Current phase that the document is in. See the **"Document Phases" on page 115** table for values.
next_phase	Next phase for that document. See the table **"Document Phases" on page 115** for possible values.
sort_order	Sort order for the pages in cache. 0 = Sequential from Page 0 1 = Reverse order
ocr_enable	1 = OCR enabled
auto_index	Used by OCR
text_len	The number of characters in the text_data field.
migrate_delay	Number of seconds to delay before migrating an object from the cache to the optical disk. Only meaningful if committal type is COMMIT_NORMAL. A value of 4294967295= do not migrate.
batch_name	Name of the batch entered by the user at the start of document entry.
phase_status	The current status for each of the eleven possible phases that the document could experience. See the table **"Phase Status" on page 115** for possible values.
phase_1st_time	The last time each of the eleven possible phases for this document was attempted. See the table **"Phase Status" on page 115** for possible values.
phase_user_id	ID number of operator who last worked on given phase.
text_data	Reserved
migr_dest_name	Name of the cache that an object will be migrated to (other than page cache).

The content of a typical **batch_dyn_hdr** table record is shown below:

```
batch_id..............9  exp_pages.............0  exp_
docs............0
pages_per_doc..........5  double_sided..........0  verify_
images........0
verify_indices.........1  committal_type.........1  batch_
total..........0
this_phase.............5  next_phase.............5  sort_
order...........0
ocr_enable.............0  auto_index.............0  text_
len............54
migrate_delay..........0
batch_name.........."pmc2"
phase_status  [0]....3,3,3,3,3,2,0,4,0,4,4
phase_lst_time[0]......757619781 => 99/1/3 09:56:21
              [1]......757619947 => 99/1/3 09:59:07
              [2]......757787161 => 99/1/5 08:26:01
              [3]....-2000000000 =>
              [4]......757790895 => 99/1/5 09:28:15
              [5]......757791193 => 99/1/5 09:33:13
              [6]......757789039 => 99/1/5 08:57:19
              [7]....-2000000000 =>
              [8]....-2000000000 =>
              [9]....-2000000000 =>
              [10]...-2000000000 =>
phase_user_id [0]....21,11,11,0,11,11,11,0,0,0,0
text_data............""
migr_dest_name......."::"
```

Document Phases

The following table shows the document phase for a batch in the **batch_dyn_hdr** table.

Number	Phase
0	Batch Creation
1	Scanning
2	Image Verification
3	Rescan
4	Assembly
5	Indexing
6	Index Verification
7	Batch Totaling
8	Committal
9	Cataloging
10	Recommit

Phase Status

The following table shows the phase status associated with a batch in the **batch_dyn_hdr** table.

Number	Phase
0	Not started
1	In progress
2	Interrupted
3	Completed
4	Unnecessary
5	Error occurred
6	Optional

batch_hdr

This is not a database table. It is an alias (view) used by MKF_tool. For example, the following command retrieves all of the information from both the **batch_dyn_hdr** and **batch_stat_hdr** tables concerning this batch.

select batch_hdr * where batch_id = b7

batch_image

This table keeps track of each image id number associated with a batch. It is updated during the scanning of a document and also during the act of scan verification. The record is cleared from this table upon the successful committal and cataloging of the batch.

Field Name	Field Description
image_index	Unique search key comprised of batch_id and image_id keys - (group).
batch_id	Identification number of the batch. Searches on this column can be done in conjunction with the “image id” column.
image_id	Image id number.
doc_index	Unique search key comprised of batch_id2, doc_id, and page_id keys - (group).
batch_id2	Same value as that of the “batch_id” column. However searches on this column can be done in conjunction with the columns “doc_id” and “page_id”.
doc_id	This is the document id number of a document relative to a specific batch. For example, if a batch has multiple documents in it, the first document will be document id number 1.
page_id	Page number of a specific document.
image_length	The size of the image expressed in bytes.

Field Name	Field Description, Continued
image_type	Type of image. 0 = scanned image 1 = text 2 = fill in form 3 = mixed 9 = separator sheet 196 = pc form (In the case of a pc form, the columns batch_id2, doc_id, and page_id are not used.)
image_ver_stat	Verification status. 0 = unverified 1 = verified good 2 = failed verification
end_of_doc	A value of 1 indicates that this is the last page of the document. The value is 0 for all other pages.
index_len	The number of bytes in the “index_value” column.
text_len	The number of bytes in the text_data column.
index_value	This column is used in conjunction with images that contain text data. The indexing information is displayed in hexadecimal form.
text_data	Reserved

The content of a typical **batch_image** table record is shown below:

```
batch_id.............8  image_id........100171  batch_id2............8
doc_id...............12  page_id..............1  image_length.....39640
image_type.............2  image_ver_stat.......0  end_of_doc...........0
index_len..............2  text_len.............8
index_value   [0]....0x3b00
```

batch_ixdir

Keeps a general record for each indexing field that is associated with a specific batch (type of index, name of the index, is verification required, etc.). The record is cleared from this table upon the successful committal and cataloging of the batch.

Field Name	Field Description
batch_id	Identification number of the batch.
index_id	Index id number for the field. This number can be cross referenced to the Index database table doctaba. (For example, a value of 36 in this column equals column name A36 of the doctaba table.)
index_type	The type of index field. 49 = numeric 50 = string 52 = menu 56 = date
max_string_len	This field only applies if the index_type is 50 which is the maximum number of characters for a string index.
verify_flag	0 = no index verify necessary 1 = index verification needed
required_flag	0 = optional 1 = index info must be entered
batch_tot_flag	0 = batch totaling unnecessary 1 = batch totaling needed
up_case_flag	0 = no auto uppercase conversion of string data 1 = Convert string data automatically to uppercase when cataloging.
source	How the fields will be indexed. 0 = unknown 1 = manually 2 = ocr 3 = aperture card

Field Name	Field Description, Continued
index_name	The name of the indexing field.
menu_name	The name of an indexing menu field.
exp_total	Expected batch total number expressed in a 16 byte format.
act_total	Actual batch total number expressed in a 16 byte format.
mask	Template being used for a numeric or date index field.

The content of a typical **batch_ixdir** table record is shown below:

```
batch_id..............9  index_id...........31  index_type.........50
max_string_len.........6  verify_flag.........1  required_flag.......0
batch_tot_flag.........0  up_case_flag.........1  source..............1
index_name............"I1"
menu_name............""
exp_total     [0]....0x00000000000000000000000000000000
act_total     [0]....0x00000000000000000000000000000000
```

batch_ixval

This table contains the actual indexing information that a given index field contains. Multiple records can exist for any given batch (one record per index field). The indexing information however is stored in a hexadecimal format. The record is cleared from this table upon the successful committal and cataloging of the batch.

Field Name	Field Description
ixval_index	Unique search key comprised of batch_id, doc_id, and index_id keys - (group).
batch_id	Identification number of the batch.

Field Name	Field Description, Continued
doc_id	The document id number.
index_id	Index id number from the Index database table doctaba.
index_type	The type of index field. 49 = numeric 50 = string 52 = menu 56 = date
index_len	The number of bytes comprising “index_value” column.
data_status	Reserved for use by OCR.
text_len	The number of bytes in the text_data column.
index_value	The index information (in hex).
text_data	Reserved. If the value of text_len = 0, the field is not displayed.

The content of a typical **batch_ixval** table record is shown below:

```
batch_id.............8  doc_id.................5  index_
id...........31
index_type...........50  index_len.............4  data_
status.........0
text_len..............0
index_value   [0]....0x646f7567
```

batch_stat_hdr

A general record is created within this table each time a batch is created. The record contains information such as batch id number, name of the document class associated with the batch, and the number of indexing fields to be completed. The record is cleared from the table upon successful committal and cataloging of the batch.

Field Name	Field Description
batch_id	Batch identification number.
queue_index	Non-unique search key comprised of queue and qtime keys - (group).
service_name	Organization/domain/object
queue	0 = not queued 1=Uncommit queue 2= queued for committal 3 = queue in progress
qtime	When the batch was queued for committal.
open_user_id	Identification number of the operator that opened the batch.
open_flag	0 = batch not currently open 1 = batch open
create_user_id	Identification number of the user that created the batch.
tab_out_flag	Equals 1 if the operator can exit the window using the ‘TAB” key.
num_indices	Number of index fields for the document.
num_reqd_indcs	Number of mandatory index fields for the document.
num_menus	Number of menu indexing fields for the document.
cluster_space	Identification number of cluster operation that is involved. This feature is currently not used as denoted by the number 65535.
cluster_index	Cluster column number.
fam_id	Optical disk family number for this document. 0=clustering

Field Name	Field Description, Continued
wfl_queue_name	WorkFlo queue name for documents entering the system.
wfl_sys_name	WorkFlo system name for documents entering the system.
retent_disp	How to handle document once retention offset is reached. 0 = delete 1 = archive
retent_base	When to start the retention offset “clock” running. 0 = date of closing 1 = date of entry
retent_offset	Number of months until the document will be deleted or archived.
act_pages	Total number of pages in the batch.
act_docs	Total number of documents in the batch.
next_doc_id	Next doc id to assign (0 relative).
csum_state	States if checksumming is being used on this batch. 0 = unknown 3 = checksumming 1 = undetermined 4= resetting checksum 2 = no checksumming
open_time	When the batch was opened.
create_time	When the batch was created.
doc_class_name	Document class name under which this batch was created.
indexing_form	Name of the indexing form used with the document class.
access_restrct	Security
delete	0 = not marked for deletion 1 = delete fast batch 2 = delete normal batch

The content of a typical **batch_stat_hdr** table record is shown below:

```
batch_id.............8  queue................0
qtime........757619523
open_user_id..........11  open_flag.............0  create_user_
id......21
tab_out_flag...........0  num_indices...........1  num_reqd_
indcs.......1
num_menus..............0  cluster_space..........0  cluster_
index........0
fam_id.................2  retent_disp...........49  retent_
base.........49
retent_offset.........12  act_pages.............11  act_
docs.............9
next_doc_id...........13  csum_state.............2
open_time.......757786838 => 99/1/5 08:20:38
create_time.....757619523 => 99/1/3 09:52:03
doc_class_name......."test"
indexing_form........"F_IXFORM1"
access_restrct[0]....0x00000001000000100000001
```

batch_data

This is the Batch services batch object data table. There is one entry in this table per object data.

Field Name	Field Description
data_index	Unique search key comprised of batch_id, object_type, object_id, and sequence_id keys - (group).
batch_id	Batch identifier.
object_type	Object type.
object_id	Object identifier.
sequence_id	ID of this data row in a sequence.
data_length	Number of bytes in the buffer.
data_value	Value of data.

batch_folder

This is the Batch services folder node table. There is one entry in this table for each folder node.

Field Name	Field Description
node_id	Folder node id.
parent_id	Parent Identifier.
base_node_name	Base node name.
batch_id	Batch Identifier. 0= non-batch node. Batch id does not need to be unique.
last_update	Time stamp of creation or update of the node.
info_len	Length associated with the information.
info_data	Actual information.

Cache Tables and Contents

Each table (and its contents) in the Transient database used by Cache services is discussed in this section; some of the tables contain examples of the contents. You can use the MKF_tool to view these tables and contents directly.

csm_caches

This table contains a list of all of the logical cache id numbers. It is referenced before any interaction with a cache is performed.

Field Name	Field Description
cache_cache_id	System-assigned identification number for a specific type of cache.
cache_name	Name of the cache.
cache_name_len	Number of characters in the cache's name.
inuse_objects	Total number of objects in use.
inuse_sectors	Total number of sectors in use.
locked_objects	Total number of locked objects currently in use.
locked_sectors	Total number of locked sectors currently in use.

The content of a typical **csm_caches** table is shown below:

```
cache_cache_id  cache_name_len  cache_name
--------------  --------------  ----------
             5              11  "page_cache1"
             6              16  "sys_print_cache1"
             7              13  "fillin_cache1"
             8              10  "bes_cache1"
             9              16  "app_print_cache1"
```

csm_free_space

This table keeps track of all of the empty sectors in the caches. Whenever the contents of the csm_used_space table are modified, this table is updated.

Field Name	Field Description
free_block	Unique search key comprised of the size and group_start keys - (group).
size	The number of available contiguous sectors from a given start sector.
group_start	Same as “start”.
start	The last three bytes indicate the starting sector number (relative to the partition), of available free space.

The content of a typical **csm_free_space** table is shown below:

```
 size  group_start   start
 ----  -----------   -----
    1   0x010001cf   0x010001cf
    1   0x0100052f   0x0100052f
    9   0x0100033c   0x0100033c
   37   0x01000277   0x01000277
   45   0x01000711   0x01000711
38626   0x0100091e   0x0100091e
```

csm_temp_id

This table contains a list of temporary id numbers. A temporary id number is assigned to any object, except an image, that is being transferred into the cache.

Field Name	Field Description
temp_id	Unique search key. Temporary system assigned identification number for each object or page being transferred into the cache. Once the transfer is complete, the temporary id number is cleared.

The content of a typical **csm_temp_id** table record is shown below:

```
temp_id
-------
4160000005
4160000006
4160000007
4160000008
4160000009
4160000010
```

csm_used_space

This table keeps track of every image, document, or object in the cache partitions. The contents of this table are referenced or updated during scanning, committal, retrieval, and printing.

Field Name	Field Description
cache_id	Number indicating the type of cache. The number can be cross referenced in the table “csm_caches”.
client_attr	Type of document, Image Form, Mixed etc.

Field Name	Field Description, Continued
created	Date and time when the image (page), was scanned.
csum	Check on the image. Value is no if there is no checksumming.
curlen	Current length. The size of this particular page (in bytes).
duration	Defined by the time in the ‘read’ field below added to the number specified in fn_edit, under Server Application services, Cache Duration, Refresh Cache Duration.
lenary	Number of sectors used to hold the object
maxlen	Amount of space reserved for this image or page in the cache.
numchunk	Number of contiguous chunks used to hold the object
obj_id	Image or document id number.
object	Unique search key comprised of the cache_id, ssn, obj_id, and page keys - (group).
page	Page number of this object. Uncommitted images have page number 65335. Documents that are committed but not yet migrated to optical disk start with page number 0. Migrated documents start with page 1.
read	Date and time when page was last read. Only applies to cache_id 1 (page-cache). All other caches display a constant date of 1969/12/31.
refcnt	Reference count. The number of times this page has been accessed while in cache.
sec_info	There are three security field for csm objects: read permission, write permission, and append/execute permission. The values in the sec_info identify the permission levels of each of the user groups. If there is no security for the object, these fields are set to ‘ANYONE'.
sec_len	Length (in bytes) of the “sec_info”
seqnum	The seqnum field keeps track of how often an object has been opened. It is for internal IS use only.
ssn	System serial number where document (or image) was created.
start	Location of an object in cache or first chunk. High byte is equal to the partition numbers and low 3 bytes is equal to the start sector number.

Field Name	Field Description, Continued
startary	The last three bytes of information indicate the starting sector number of the page relative to the partition.
temp_flag	Not used.

The content a typical **csm_used_space** table record is shown below:

```
cache_id.............5  ssn............100000  obj_id.........100090
page.................0  maxlen...........128  curlen...........128
temp_flag............0  sec_len...........0  refcnt............1
seqnum...............1  numchunk..........1  startary....0x01000043
lenary...............1
created.........756518115 => 93/12/21 15:55:15
read.................0 => 69/12/31 16:00:00
sec_info      [0]........
client_attr   [0]....0,2,0,0,0,0,0,0
```

Print Tables and Contents

print_docs

When multiple documents are to be printed from a single print request, a record is created in this table. One record exists for each document to be printed. Data I/O with this table is primarily accomplished through the PRII shared library.

Field Name	Field Description
print_doc	Unique search key comprised of the request_id and doc_order keys - (group).
request_id	System-assigned request identification number.
doc_order	The order in which the selected documents will be printed (starting from 0 for the first document).
doc_id	The document id number of the selected document.
first_page	Beginning page number of a document.
last_page	Ending page number of a document.
ssn	System serial number if the column “svc_type” has a value of 2.
svc_type	The service that requested the print job. 1 = Document services 2 = Cache services
delete_after	This value is set by the client software. 0 = do not delete from cache 1 = delete after printing
svc_id	Service id of service which has data

print_options

This table keeps a record of the options chosen for each entry in the print_requests table. Options include paper size, printer, header pages, etc. Data I/O with this table is primarily accomplished through the PRII shared library.

Field Name	Field Description
request_id	Unique search key - system-assigned request identification number.
print_time	Non-unique search key - time at which page fetching will begin.
priority	Priority level of the print job. If no value is present, the default priority level of 4 is being used. The valid range is 0 through 9, with 9 having the highest priority.
paper_size	Paper size to be used. 0 = unknown 1 = letter 2 = legal 3 = B-size 4 = C-size 5 = D-size 6 = E-size 7 = A0 8 = A1 9 = A2 10 = A3 11 = A4 12 = A5 13 = B4 14 = B5 15 = 18x24 16 = top tray 17 = bottom tray 18 = third tray 19 = default 20 = half letter 21 = best available 22 = 10x14 24 = executive (For FAX servers the value is always 11.)
collate	A value will only exist in this column if the printer has been configured to collate documents (through the client software).
staple	Not used.
two_sided	If no value is present, the print job is single-sided.

Field Name	Field Description, Continued
scaling	0 = normal (exact vert, clip horiz) 1 = clip both (Approx size, clipping) 2 = exact (exact fit) 3 = scale approx (Approx size, no clip) 4 = original size 5 = center (Not specified by user, automatic for screen prints) 6 = enhanced exact (not currently used)
request_header	0 = no header page 1 = print header page
doc_headers	If no value is present, document header pages will not be printed. If a value of 1 exists, a header page containing the document id number will be printed.
form_name_len	The number of bytes in the form_name column.
form_name	Name of the form to be printed.
orientation	NULL = default (does not appear) 1 = landscape 2 = portrait 3 = no rotation
annotations	0 = no annotations 1 = print annotations
note_len	The number of bytes in the note column (max 42).
note	Message up to 42 characters long that will appear on the header page of the print job.
overlay	Not used.
phone_num_len	The number of bytes in the phone_num column.
phone_num	Fax machine phone number.
phone_ext_len	Length of the phone_ext field.
phone_ext	TBD
mem_len	TBD

Field Name	Field Description, Continued
memo	Contains a memo entered through the client software.
headline_len	The number of bytes in the headline column.
headline	Headline message to be sent to the fax as part of the print job. The message can be up to 98 characters long.
fax_mode	null = coarse, 1 = fine
page_footnote	1 = page footnotes used
time_footnote	1 = time footnotes used
eject_tray	Paper output tray number
cover_doc	TBD
cover_ssn	TBD

print_svcs

This table is used to map Document and Cache services names to a 2 byte id. Print services uses IDs instead of names in the print_requests and print_docs tables, and the print_svcs table is used to save the id to mapping. There is one record in the print_svcs table for each document or cache service that Print services has communicated with.

Field Name	Field Description
svc_id	Unique search key - TBD
svc_type	1 = Document services 2 = Cache services
object_name	Object name of the service.
domain_name	Domain name of the service.
org_name	Organization name of the service.

The content of a typical **csm_temp_id** table record is shown below

```
svc_id.................0  svc_type..............1
object_name..."DocServer"
domain_name......"corona"
org_name........"FileNet"
-------------------------------------------------------------------------
---
svc_id.................1  svc_type..............2
object_name.........."sys_print_cache1"[for printing documents]
domain_name........."corona"
org_name............."FileNet"
```

print_reqopt

This is not actually a database table. It is an alias (view) used by MKF_ tool. For example, the following command retrieves all of the information from both the **print_requests** and **print_options** tables concerning this document.

select print_reqopt * where doc_id = 100127

print_requests

This table keeps a record of each outstanding print request. Data I/O with this table is primarily accomplished through the use of the PRIl shared library.

Upon the completion of a print job, the print request record is cleared within one minute from this table by the program PRI_daemon. If a print request is cancelled by a user, PRI_daemon waits ten minutes before actually clearing the record from this table.

Field Name	Field Description
request_id	Unique search key - system-assigned request identification number.
done_time	Time at which the request was completed, cancelled, or failed.
request_time	When the print request was placed into the queue.
request_length	If printing Text=size of print job (in bytes); Image=# of pages.
print_err	Reason for print job failure. fn_msg command decodes hex number.
pages_printed	Number of pages successfully printed.
notify_addr	Ethernet address of server to notify when print job completes. 0 = do not notify
fax_request	NULL = print request 1 = fax request
security	Document or text file security information.
doc_id	Document id number of document being printed. Number 4160000001, and up indicates text, not a document, is being printed.
first_page	Number of the first page that will be printed.
last_page	Number of the last page that will be printed.
ssn	System serial number if the svc_type column has a value of 2.
svc_type	The service that requested the print job. 1 = Document services 2 = Cache services
delete_after	0 = do not delete from cache 1 = delete after printing
svc_id	ID of service holding data to print
user_id	ID of user issuing print request
terminal_id	ID of terminal print request issued from
print_order	Unique search key comprised of the printer_num, inverse_prio, and sequence_num keys - (group).

Field Name	Field Description, Continued
printer_num	Printer number assigned this print request
inverse_prio	Inverse priority (10-priority #) if job ready to print. Else, 10 if print job is suspended or is scheduled for printing at later time.
sequence_num	Sequence number of print request. ex. 5010
request_stat	0= unknown 1 = queued 2 = printing 3 = completed 4 = failed 5 = cancelled 6 = waiting 7 = fetching 8 = suspended
copies	Number of copies to print. null for default (1 copy)

Requests Tables and Contents

bkg_request

This table keeps track of all background job requests. Optical disk to disk copies, optical disk imports, and optical disk annotation copies are considered background jobs .

Field Name	Field Description
job_number	Job number.
job_type	1 = Copy Documents 2 = Import Documents 3 = Copy Annotations 4 = Create Archive 5 = Find Open Docs 6 = Import Archive 7 = Migrate Documents 8 = Scan Surfaces 9 = Update Surfaces
jobqueuenum	Job number to use for messages.
long_wd_ary	Data relative to task type.
num_long_wds	Number of words in the array.
pid	Process ID of background job.
prevjob	Previous job in queue.
start_time	The time that the background job was started.

Field Name	Field Description, Continued
surfid	ID of the current optical disk surface. 0 = multiple surfaces are required.
suspended	0 = running 1 = suspended 2 = queued

write_request

This table keeps track of each document in the page cache that needs to be written to optical disk. This table is updated during committal and then cleared after the successful transfer of a document to optical disk.

Field Name	Field Description
doc_id	Unique search key - document ID.
cluster_space	Which clustering application.
cluster_id	Cluster ID.
family_id	Primary family to write.
new_cluster	Whether this is the first document in cluster. 00 = not a new cluster 01 = new cluster
num_pages	Number of pages in document.
req_type	0 or NULL = document 1 = batch
commit_error	Error during committal. 0 = none
batch_name	Batch name. Prefix of “F_” indicates that the system software created the batch.

7

Security Database Tables and Contents

The Security database contains information about all users, groups, and devices on the system. The tables are used exclusively by the Security services.

This chapter discusses the following Security database tables:

- sec_object
- sec_system
- sec_deleted
- sec_groups
- sec_functions
- sec_funcmbr
- sec_namemap
- sec_dbinfo
- sec_rm_config
- sec_map_prin_to_dn
- sec_ce_dom_to_id

Tables and Contents

Each table (and its contents) in the Transient database used by the Security service is discussed in this section; some of the tables contain examples of the contents. Some of the information in these tables is encrypted. You can use MKF_tool to view these tables and contents directly.

sec_object

This table contains one record for each object on the system (i.e., user, group, and device).

Field Name	Description
obj_class	The type of object: 1 = system 3 = user 2 = group 4 = device
obj_name	ASCII string containing object name
obj_id	Unique object ID
admin_class	A bitfield representation of the administrative class for the given object. Identifies the object as belonging to one or more of the following: supervisor, principal, account, group, audit, or "not administrator." If someone is "not administrator" then he/she is not any of the other administrator types. This field is not generally human-readable due to the way it is stored.
dev_class	The type of device: 1= terminal 2 = printer 5= fax null=not a device
epasswd	Encrypted password
prim_group	Primary group ID
admin_group	Administrative group ID

Field Name	Description, Continued
language	Native user language
dev_security	If the object is a user, this indicates if the user is subject to device ("terminal") security. If this is set, then a user may be restricted from accessing certain documents depending on what terminal he/she is using to do so. If the object is a device, this indicates whether the device is subject to device security, which means that it has different security than other terminals from which users can access documents.
creation time	Database creation date
from_min	Login restriction time - start time minutes (0-59)
to_min	Login restriction time - end time minutes (0-59)
from_hour	Login restriction time - start time hours (0-23 military)
to_hour	Login restriction time - end time hours (0-23 military)
from_dweek	Day of week (0-6: 0= Sunday)
exp_date	UNIX time for account termination
succ_log_time	Last successful login time
success_where	Last successful login location
fail_log_time	Last failed login time
failed_where	Last failed login location
failed_err	Last failed login error cause
nbr_logons	Total number of time nbr logged in to system
log_bits	Bits indicating logging options: log successful logons 1 (bit value 1) log failed logons 01 (bit value 2) log security changes 001 (bit value 3) log all 111
pwd_exp_time	Calculated UNIX expiration time
pwd_fail_time	Time of password failure

Field Name	Description, Continued
pws_fail_atts	Number of failed nbr attempts
sys_override	0=use system, 1=use users
max_session	Number of allowed nbr sessions
comments	General comments
sess_group	Session group
dn_len	Not used.
dn	Not used.
dn_hash	Not used.
concurrent_hwm	High-water mark for concurrent logons
pwd_never_expire	Password never expire flag

sec_system

This table contains the system record for the Security service.

Field Name	Description
system_id	Record which equals unique key
ceiling_id	Number to track assigned user ids
ceiling_snbr	Value of last session number assigned
ceiling_fid	Value of next function number
ceiling_lan	Value of current language ID
dev_security	0=off, 1=on
no_func_def_ok	0=not ok, 1=ok
update_time	Stamp of time db last updated
from_min	Start time in minutes (0-59)
to_min	End time in minutes (0-59)

Field Name	Description, Continued
from_hour	Start time in hours (0-23: military)
to_hour	End time in hours (0-23: military)
from_dweek	Day of week (0-6: 0=Sunday)
to_dweek	Day of week (0-6: 0=Sunday)
max_sessions	The maximum number of current logons per user. This ensures that one user cannot use up all the SLU's for a given server.
highwater	Highest number ever logged on. This field is currently not used.
pwd_spec_char	Special character required (0=no, 1=yes)
pwd_min_len	Maximum password length (0=minimum, 8=maximum)
pwd_rnwl_days	Password renewal from last change (0=not set, 1=minimum, 365=maximum)
pwd_grace	Time (in days) until lack of password change causes account suspension (0=not set, 1=minimum, 90=maximum)
pwd_attempts	Used to identify nbr of password attempts until account is disabled (0=not set, 1=minimum, 100=maximum)
pwd_fail_mins	0=not set, 1=minimum, 100=maximum
log_bits	Bits indicating logging options: log successful logons 1 (bit value 1) log failed logons 01 (bit value 2) log security changes 001 (bit value 3) log all 111
language	ASCII value indicating language
db_level	DB level of Security database
allow_over	Override system definitions (0=no, 1=yes)
ceiling_db_id	Pseudo_ID for DB logon
annot_security	Annotation security (0=no, 1=yes)
pwd_chg_upon_reset	Password change upon reset (0=no, 1=yes)
ext_pwd_validation	External third party password validation (0=no, 1=yes)

sec_deleted

This table is not currently used. It was designed to contain one record for each deleted object on the system.

Field Name	Description
obj_id	Deleted object ID
obj_name	ASCII string containing object, domain, and organization
obj_class	Object class
dev_class	Deleted device class
del_time	Deletion time

sec_groups

This table contains one record for each direct membership occurrence.

Field Name	Description
grp_group	The identifier for this group, containing group_id and member_id.
group_id	The numeric identifier (Object ID) for the group.
member_id	The numeric identifier (Object ID) for the member of the group.
grp_member	A member of the group specified in grp_group, containing group_id2. Does not display.
member_id2	Same as member_id (MKF requirement for multiple key setup).
group_id2	Same as group_id (MKF requirement for multiple key setup). Does not display.
member_class	Displays what kind of member it is: 1=admin 2=the group entry (specifying the same group) 3=user of group

sec_functions

This table contains the name and ID of each function that is known to Security services. Certain functions, such as “read from COLD,” write to COLD,” and I think some user defined functions, can have function security. This means access to these functions is restricted.

Field Name	Description
fun_name	ASCII string containing object, domain, and organization.
fun_id	A field associating a unique numeric ID to the function name specified in the fun_name field.

sec_funcmbr

This table is very similar to the sec_groups table. It contains one record for each “function_member.” Each function in the table can have members, just like a group.

Field Name	Description
fun_owner	This is a field grouping in MKF that indicates a membership. Should contain the same value as the fun_member field. Does not display.
fun_id	Indicates the MKF ID of the function from the functions table. Should contain the same value as the fun_id2 field.
member_id	Indicates the MKF object ID of the member of the function. Should contain the same value as the member_id2 field.
fun_member	This is a field grouping in MKF that indicates a membership. Should contain the same value as the fun_owner field. Does not display
member_id2	Indicates the MKF object ID of the member of the function. Should contain the same value as the member_id field.

Field Name	Description, Continued
fun_id2	Indicates the MKF ID of the function from the functions table. Should contain the same value as the fun_id field.
member_class	Indicates the object class of the member: 0=anyone 1=admin 2=group 3=user

sec_namemap

This table and the sec_dbinfo table are used to map FileNet users onto native database logons (for Oracle, DB2 and SQL Server). One of these is created for each mapping made to the DB logon. For example, FileNet user <*name1*>*FN* is the same as Oracle user <*name1*>*ORA*.

Field Name	Description
filenet_map	Field grouping in MKF that indicates a mapping. Should contain the same value as the db_map field. Does not display.
obj_id	Indicates the object id of the FileNet user that is to be mapped to a database user. Should contain the same value as the obj_id2 field.
db_pseudo_id	An index of a row in the sec_dbinfo table indicating the database user that this FileNet user is mapped to.
db_map	Field grouping in MKF that indicates a mapping. Should contain the same value as the filenet_map field. Does not display.
db_pseudo_id2	An index of a row in the sec_dbinfo table indicating the database user that this FileNet user is mapped to.
obj_id2	Indicates the object id of the FileNet user that is to be mapped to a database user. Should contain the same value as the obj_id field.

sec_dbinfo

This table contains a record for each db logon added by the Administrator.

Field Name	Description
db_name	Native DB user name
db_pseudo_id	Native DB pseudo ID
db_pwd	Native DB encrypted password
db_creator_id	ID of creator

sec_rm_config

This table contains a single record that defines the system-level Record Management configuration.

Field Name	Description
key_id	A numeric identifier that defines the security level. The current support is for system-level security only.
declare_level	A numeric identifier that sets the declaration level (i.e., lockdown mode). Level 1 = READ-ONLY (default) Level 2 = No Write (only new annotations can be created) Level 3 = No Change to document security (No op)
rm_group_id	The object ID of the configured “Record Management” group
purge_level	Not used.
scrub_level	Not used.
log_level	A numeric identifier that defines the “log level” mode. Level 0 = Minimal (errors only) Level 1 = Verbose
lock_sysadming	Not used.

sec_map_prin_to_dn

This table holds mappings of IS objects to their LDAP distinguished names (dn).

Field Name	Description
obj_id	Object id from sec_object table
ce_dom_id	Links to ce_dom_to_id table
dn_hash	Distinguished name hash key
ce_dom_id2	Links to ce_dom_to_id table
dn_len	Distinguished name string length
dn	Actual distinguished name

sec_ce_dom_to_id

This internal table contains all unique Content Engine (CE) GUIDs.

Field Name	Description
ce_dom_GUID	Unique GUID key for a CE domain
ce_dom_id	Independent internal key number

Index