IBM FileNet Image Services
Version 4.2

System Reference Guide

IBM
Note
Before using this information and the product it supports, read the information in “Notices” on page 167.

This edition applies to version 4.2 of IBM FileNet Image Services (product number 5724-R95) and to all subsequent releases and modifications until otherwise indicated in new editions.

© Copyright IBM Corporation 1992, 2011.
US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.
Contents

About this manual 17

Overview 17
Document revision history 17
Scope of this guide 18
Accessing IBM FileNet Image Services documentation 20
Software education 20
Comments and suggestions 20

1 System design and architecture 21

Server configurations 21
Combined server configuration 21
Dual or multiple storage library server configuration 22
Application server configuration 23
Entry server configuration 25

Directory structure 27

Directory contents 28
/fnsw/bin 28
/fnsw/dev/1 29
/fnsw/diag (UNIX only) 30
/fnsw/etc 30
/fnsw/lib 30
Contents

/fnsw/local (or \fnsw_loc) 31
 /fnsw/local/sd 31
 /fnsw/local/spool 32
 /fnsw/local/tmp 32
 /fnsw/local/wfl 32
 /fnsw/local/<RDBMS> 32
RDBMS (/fnsw/oracle, /fnsw/db2, or \fnsw\mssql) 33
/fnsw/procs 34
/fnsw/spool (UNIX only) 34
\fnsw\setup (Windows only) 34

Subsystem descriptions 35
AT (Release Tools) 35
AR (Archiving) 35
B2 (Enterprise Backup and Restore) 35
BS (Batch Services) 35
CO (Courier Networking) 36
CS (Cache Services) 36
DM (Database Maintenance) 36
DS (Document Services) 37
EV (SystemV Applications Executive) 37
FL (FileNet LAN Analysis Tool) 37
GB (Generic Database) 38
HE (Help Error) 38
IS (Index Services) 38
LM (Software License Management) 38
MK (Multi-Keyed Files) 39
MV (SystemV Miscellaneous) 39
NC (SystemV Miscellaneous) 39
NL (National Language Services) 39
2 Configuration Files 42

Configuration Database 43

Supplied Configuration Files 46
erm.msg 46
  UNIX Files 47
  Windows Files 47
  Language Directory 47
fninfo.msg 48
tmmsg.msg 49

Generated Configuration Files 50
as_conf.g 50
as_conf.s 52
init.ora 53
inx_conf 54
snmp.conf 55
MKF.ddl (s) 55
3 Applications 72

Application Program Descriptions 73
Backup/Restore 73
  backup_daemon 73
BRBs 73
BRRs 73
Batch Entry Services 74
  BESs 74
Cache Services 74
  CSMs 74
    CSM_daemon 74
Document Services 75
  bes_commit 75

nch_dbinit 57
nch_domain 58
print_config 58
rdb.init 59
serverGroup 59
serverConfig 61
  Tuning Courier timeout durations 64
  Add optional recv() timeout value for worker threads 65
  Add new utility "COR_set_recv_timeout" 66
  Add optional timeout value for prefetch threads 67
  Add new utility "COR_set_prefetch_timeout" 68
  Add optional maximum prefetch thread count 69
ssn 69
tapeconfig 69
setup_config 71
Contents

DLIs 75
DLSs 75
DOCs 76
ds_init 76
ds_notify 76
fbc_commit 76
OSIs 76
OSSs 77
RMKs 77
rmt_commit 77
sas 77
Fax Services 78
FSMs 78
File Services 78
FILs 78
Index Services 78
INXbg 78
INXs 79
INXu 79
MKF Database Services 79
MKF_clean 79
Network Clearinghouse Services 79
NCHs 79
NCH_daemon 80
Print Services 80
PRI_check 80
PRI_daemon 80
PRI_notify 81
PRIs 81
PRI_worker 81
4 Shared libraries  86
## Contents

- **Shared library locations** 87
- **Shared library descriptions** 88

### All services 88
- COR 88
- PPM 88

### Batch Entry Services 89
- BES 89
- BESI 89

### Cache Services 89
- CSM 89
- CSMI 89

### Document Services 89
- ASH 89
- BKG 90
- CKS 90
- CMT 90
- CNF 90
- CNT 90
- DBL 90
- DBP 91
- DIA 91
- DIG 91
- DLI 91
- DLIi 91
- DLIo 91
- DLS 92
- DLSI 92
- DLSr 92
- DOC 92
Contents

DOCa 92
DOCb 92
DOCl 92
DOCp 93
DSA 93
DT 93
FDT 93
FLT 93
HLT 93
IS 94
MKF 94
OSI 94
OSII 94
OSIr 94
OSS 94
OSSI 95
QMA 95
RFT 95
RMK 95
RMKr 95
SAS 95
SDC 96
SRF 96
Index Services 96
INX 96
INXa 96
INXD 96
INXI 97
INXr 97
Network ClearingHouse Services 97
Contents

NCH 97
NCHI 97
NCHr 97
SLMI 97
Print Services 98
PRI 98
PRII 98
Security Services 98
SEC 98
SECI 98
SECr 98
SMM 98
SSD 99
SSU 99
Storage Library Services 99
ARM 99
CLT 99
DIS 99
ODF 99
ODS 99
ODT 100
ODU 100
ODX 100
OFA 100
OPM 100
OSA 100
SLT 100
SNT 101
WRT 101
System Configuration 101
Contents

CDB 101
Miscellaneous System V 101
ORA 101
System Messages 101
ERM 101

5 Permanent database tables and contents 102

Tables and contents 103
annot_log 103
annotations 104
cluster_map 105
docs 106
family_disk 110
family_locator 112
lib_surfaces 113
od_stats 114
remote_family 115
scalar_numbers 116
surf_dyn_info 116
surf_info 119
surf_locator 119
surf_stat_info 121
surface_activity 123

6 Transient database tables and contents 125

Batch tables and contents 126
batch_ctl 126
### Contents

- **batch_doc** 127
- **batch_dyn_hdr** 128
  - Document Phases 131
  - Phase Status 131
- **batch_hdr** 132
- **batch_image** 132
- **batch_ixdir** 134
- **batch_ixval** 135
- **batch_stat_hdr** 137
- **batch_data** 139
- **batch_folder** 140

**Cache Tables and Contents** 141
- **csm_caches** 141
- **csm_free_space** 142
- **csm_temp_id** 143
- **csm_used_space** 144

**Print tables and contents** 147
- **print_docs** 147
- **print_options** 148
- **print_svcs** 150
- **print_reqopt** 151
- **print_requests** 151

**Requests tables and contents** 154
- **bkg_request** 154
- **write_request** 154

### 7 Security database tables and contents 156

**Tables and contents** 157
About this manual

This System Reference Guide is intended for use by support personnel responsible for IBM® FileNet® FileNet Image Services 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®) used on your FileNet Image Services servers.
- An understanding of, or prior experience with, the system administration tools available on your operating system (smit on AIX, sam on HP-UX, admintool and swmtool on Solaris 8, Solaris Management Console tools on Solaris 9 and Solaris 10, and Control Panel on Windows).
- An understanding of file system configuration methods and logical volume management concepts.

Document revision history

<table>
<thead>
<tr>
<th>Image Services version</th>
<th>Date</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.2</td>
<td>May 2011</td>
<td>Initial release.</td>
</tr>
</tbody>
</table>
Scope of this guide

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

- Server configurations
- Directory structure (and content)
- Subsystem descriptions
- Configuration files (some supplied with the FileNet Image Services software others generated at the time of installation)
- Application programs
- Shared libraries
- MKF database tables and their 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.

To download IBM FileNet documentation from the IBM support page, see “Accessing IBM FileNet Image Services documentation” on page 20.
### Document Name

<table>
<thead>
<tr>
<th>Document Name</th>
<th>Document Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>System Adminstrator’s Handbook</td>
<td>A detailed discussion of critical System Administrative tasks, including database management, security management, system management, storage library control, and more.</td>
</tr>
<tr>
<td>System Administrator’s Companion for UNIX</td>
<td>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.</td>
</tr>
<tr>
<td>System Administrator’s Companion for Windows</td>
<td>A detailed discussion of system messages and an explanation of the message tools.</td>
</tr>
<tr>
<td>System Tools Reference Manual</td>
<td>A description of the system tools used to diagnose and manage a FileNet system. Some tools are used only by support personnel, whereas many others are used by FileNet Image Services System Administrators to analyze, diagnose, update, and repair FileNet Image Services components.</td>
</tr>
<tr>
<td>Enterprise Backup/Restore User’s Guide</td>
<td>A description of Enterprise Backup/Restore (EBR), EBR concepts and functions, and how to use EBR to backup and restore a FileNet Image Services system.</td>
</tr>
<tr>
<td>Third-Party Backup/Restore Guidelines</td>
<td>A listing of general requirements and recommendations when using non-FileNet backup and restore programs or utilities.</td>
</tr>
<tr>
<td>Restore Guide for Windows</td>
<td>A discussion of methods used for restoring the permanent, transient, security, and RDBMS databases.</td>
</tr>
</tbody>
</table>
Accessing IBM FileNet Image Services documentation

To access documentation for IBM FileNet Image Services products:


2. Select IBM - Product Documentation for FileNet Image Services from the list of search results.

Software education

IBM provides various forms of education. Please visit the IBM Training and Certification for IBM software page at: www.ibm.com/software/sw-training.

Comments and suggestions

If you have comments or suggestions regarding the product documentation, please send your comments by e-mail to:

comments@us.ibm.com.

Be sure to include the name of the product, the version number of the product, and the name and part number of the book (if applicable). If you are commenting on specific text, include the location of the text (for example, a chapter and section title, a table number, a page number, or a help topic title).
System design and architecture

The FileNet 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)

Server configurations

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 envi-
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 service representative.

**Application server configuration**

An Application server configuration consists of two or more FileNet 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.

\( (\text{root directory}) \)

\texttt{\fnsw} \quad \texttt{\fnsw\_loc}

\texttt{\bin} \quad \texttt{\dev} \quad \texttt{\etc} \quad \texttt{\lib} \quad \texttt{\oracle} \quad \texttt{\procs} \quad \texttt{\setup} \\
\texttt{\EBR} \quad \texttt{\logs} \quad \texttt{\mssql} \quad \texttt{\oracle} \quad \texttt{\db2} \quad \texttt{\sd} \quad \texttt{\spool} \quad \texttt{\tmp} \quad \texttt{\wfl}

**Directory contents**

While the directory structure may be different depending on server usage or platform differences, all FileNet 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.

\texttt{/fnsw/bin}

The \texttt{/fnsw/bin} directory contains FileNet-specific tools and application programs. For a complete discussion on the tools located in the \texttt{/fnsw/bin} directory, see the *System Tools Reference Manual*. 
To download IBM FileNet documentation from the IBM support page, see “Accessing IBM FileNet Image Services documentation” on page 20.

The default path is:

**UNIX**

/fnsw/bin

**WIN**

\fnsw\bin

/fnsw/dev/1

On the UNIX platform, the /fnsw/dev/1 directory contains links/device files to FileNet 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

**WIN**

\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 exists on the UNIX servers 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 FileNet Image Services software installation process or by your service representative during diagnostic evaluation of the system.

The default path is:

```unix
/fnsw/etc
```

WIN

```
default etc directory```

/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

**WIN**

\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

**WIN**

\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

WIN \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

WIN \fnsw_loc\tmp

/fnsw/local/wfl

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

UNIX /fnsw/local/wfl

WIN \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
\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, /fnsw/db2, or \fnsw\mssql)

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

**UNIX**

(Oracle and DB2): /fnsw/oracle, /fnsw/db2

**WIN**

(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.

- If DB2 is installed on the system, the \fnsw\db2 directory exists.

/fnsw/procs

The path, separated by platform, is shown below:

**UNIX**

/fnsw/procs

**WIN**

\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 FileNet 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 FileNet Image Services software.
Subsystem descriptions

Each FileNet Image Services subsystem is a specialized portion of the IS software than interacts with other IS subsystems to provide the various system services.

This section provides a summary of each of the subsystems that comprise the FileNet Image Services software.

AT (Release Tools)

Tools used during the installation and updating of the FileNet 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 and 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; accumu-
lating 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:ExampleCo.

**CO (Courier Networking)**

The COR_Listen process listens for incoming RPC requests. When an RPC (Remote Procedure Call) arrives, COR_Listen immediately finds a suitable service process (stub) and hands over the RPC, then resumes listening for the next RPC. Only one COR_Listen process is running at any given time.

**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 FileNet 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 FileNet 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 event log.

- **Xapex** is the graphical user interface application used to log into FileNet 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 operating systems. 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 only), and snoop (Solaris 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 FileNet 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_DBn), the permanent database (Perm_DBn), the network clearinghouse database (NCH_db0), and the security database (Sec_DBn).

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 (your company name): 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 Tools) 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) are a set of programs and libraries that provide user authentication and access control for the FileNet 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 FileNet 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 vendor-acquired toolkit on which the graphical user interface server applications are built.
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 an Image Services system. It is created and modified by the fn_edit program. For a complete description of the fn_edit utility, see the System Configuration Overview. To download IBM FileNet documentation from the IBM support page, see “Accessing IBM FileNet Image Services documentation” on page 20.

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
```

**WIN**

```
\fnsw_loc\sd\conf_db\IMS_xxx.cdb
```

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

3 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.

4 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.

See the Image Services System Messages Handbook for more infor-
mation. To download IBM FileNet documentation from the IBM support
page, see “Accessing IBM FileNet Image Services documenta-
tion” on page 20.

fninfo.msg

The different platforms use unique message resources to provide infor-
mation 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. (See “snmp.conf” on page 55 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.

```
/fnsw/local
    ssn
    setup_config

/fnsw/local/sd
    nch_dbinit
    nch_domain
    inx_conf
    print_config
    snmp.conf
    as_conf.g
    serverGroup
    db.init
    rdb.init

/fnsw/local/oracle
    as_conf.s
    tapeconfig
    nch.ddl
    permanent.ddl
    transient.ddl
    sec.ddl
    init.ora

/fnsw/local/mssql
    config.sql

/fnsw/etc
    serverConfig
```

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:
2 Configuration Files
Generated Configuration Files

- 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.

```plaintext
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.

```plaintext
(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...
```
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 Oracle installations, the 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.

```
ctrl.ora
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.

```plaintext
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, your service representative 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.

<table>
<thead>
<tr>
<th>*.ddl (MKF) File</th>
<th>Database</th>
<th>Created by</th>
<th>Updated by</th>
</tr>
</thead>
<tbody>
<tr>
<td>nch.ddl</td>
<td>NCH</td>
<td>fn_util initnch</td>
<td>fn_util initnch</td>
</tr>
<tr>
<td>permanent.ddl</td>
<td>Permanent</td>
<td>fn_util initperm</td>
<td>fn_util updateperm</td>
</tr>
<tr>
<td>sec.ddl</td>
<td>Security</td>
<td>fn_util initsec</td>
<td>fn_util updatesec</td>
</tr>
<tr>
<td>transient.ddl</td>
<td>Transient</td>
<td>fn_util inittrans</td>
<td>fn_util updatetrans</td>
</tr>
</tbody>
</table>

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:

```plaintext
--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 Wf1S
server \ SQLServer
additem fn_system defService1Desc 1 \ RFSServer
SecurityService \ 
additem fn_system defDeviceDesc 0 \ 
additem fn_system defCacheDesc 2 \ batch_cache1 revise_cachel
\ \ \ 
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:YourCorp
```

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  {
p     print_cachesys_print_cache1:earth:FileNet 
   cache_fullness_threshold 80
   print_page_bytes 50000
...
```

**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
```

When 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 one of these directories (separated according to platform):

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

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

<table>
<thead>
<tr>
<th>System Component</th>
<th>Request Handler</th>
<th>Maximum</th>
</tr>
</thead>
<tbody>
<tr>
<td>Batch services</td>
<td>BESs</td>
<td>8</td>
</tr>
<tr>
<td>Cache services</td>
<td>CSMs</td>
<td>12</td>
</tr>
<tr>
<td>Index services</td>
<td>INXs</td>
<td>12</td>
</tr>
<tr>
<td>Document services</td>
<td>DOCs</td>
<td>12</td>
</tr>
<tr>
<td></td>
<td>RMKs</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>DLIs</td>
<td>5</td>
</tr>
<tr>
<td>NCH services</td>
<td>NCHs</td>
<td>12</td>
</tr>
<tr>
<td>OSAR services</td>
<td>OSI\s</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>OSS\s</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>OSC\s</td>
<td>3</td>
</tr>
<tr>
<td>Print services</td>
<td>PR\s</td>
<td>12</td>
</tr>
</tbody>
</table>
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.

An optional eighth column is available for the purpose of tuning Courier timeout durations. See page 64 for details.

<table>
<thead>
<tr>
<th>System Component</th>
<th>Request Handler</th>
<th>Maximum</th>
</tr>
</thead>
<tbody>
<tr>
<td>Backup</td>
<td>BRBs</td>
<td>1</td>
</tr>
<tr>
<td>Security services</td>
<td>SECs</td>
<td>12</td>
</tr>
<tr>
<td>Restore</td>
<td>BRRs</td>
<td>1</td>
</tr>
</tbody>
</table>

- 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 eighth (8) parameter is optional. See Tuning Courier timeout durations on page 64 for details.
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  
d intf  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  
RJEIs  0134231070 1 3 0 0 64  
OSCIs  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  
...  
```

**Tuning Courier timeout durations**

As of Image Services 4.2.0, an optional parameter, Tuning courier timeout durations, is available.

The values for this parameter are:
• Add optional recv() timeout value for worker threads.
• Add new utility "COR_set_recv_timeout"
• Add optional timeout value for prefetch threads.
• Add new utility "COR_set prefetch_timeout"
• Add optional maximum prefetch thread count to be between 75 and 200.

Add optional recv() timeout value for worker threads

An optional new column appendeds to /fnsw/etc/serverConfig. This new column can contain nothing, a zero, or a positive timeout value in seconds which is used as the timeout value for the particular request handler type.

For example:

<p>| | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>NCHs</td>
<td>02</td>
<td>2</td>
<td>24</td>
<td>1</td>
<td>0</td>
<td>30</td>
<td></td>
</tr>
<tr>
<td>CSMs</td>
<td>0134231040</td>
<td>1</td>
<td>24</td>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DOCs</td>
<td>0134231041</td>
<td>1</td>
<td>24</td>
<td>1</td>
<td>0</td>
<td>100</td>
<td></td>
</tr>
<tr>
<td>INXs</td>
<td>0134231042</td>
<td>1</td>
<td>24</td>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PRIs</td>
<td>0134231043</td>
<td>1</td>
<td>24</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>BESs</td>
<td>0134231044</td>
<td>1</td>
<td>16</td>
<td>1</td>
<td>0</td>
<td>555</td>
<td></td>
</tr>
</tbody>
</table>

In this example:

NCHs has a timeout value of 30
CSMs has a timeout value of 0 (no entry in column 7)
DOCs has a timeout value of 100
INXs has a timeout value of 0 (no entry in column 7)
PRIs has a timeout value of 0 (0 = no timeout)  
BESs has a timeout value of 555

The timeout value for each incoming RPC is set during prefetch period. Just before the RPC goes into the service queue, the prefetch thread sets the timeout value in a new structure field in COR handle. The COR abstract loads the serverConfig information once at startup into shared memory and the information is available for COR_Listen(s) through a new COR entry point.

A value of 0 in serverConfig means no timeout. If timeout value is not specified, it means no timeout. If a timeout is reached, a syslog entry is generated:

2010/10/15 12:35:35.023 155,20,4 <fnsw> COR_Listen -pt -s32769 -t3600 -d100 (991476.515.39 0xf20f4.203) ...

Courier error: recv() timed out (15 secs) from ip address 9.65.17.89 [1850]. request handler type = 134231040

Add new utility "COR_set_recv_timeout"

This new tool takes zero, one or two parameters. If run with no parameter, it displays the current value of the timeout for all request handlers. If run with one parameter, it displays the timeout value for a request handler type. If run with two parameters, it dynamically sets the new timeout value in seconds for a request handler type.

For example:

COR_set_recv_timeout display timeout values for all stubs
COR_set_recv_timeout CSMs display timeout value for CSM
only
COR_set_recv_timeout DOCs 100 set timeout value for DOCs to 100

The range of the timeout value must be in between 0 and MAXINT. If the value specified is out of the range, then it's a no-op and the application exits with a value of 1. Having a discipline on the exit return value of the utility makes it scriptable.

If run with -h or illegal parameters, it displays this usage message.

Usage: COR_set_recv_timeout [stub type] [timeout in seconds]

Add optional timeout value for prefetch threads

In all previous Image Services releases, COR_Listen launched with a -d3600 options. This is the timeout value used for prefetch threads. 3600 seconds is one hour. In Image Services 4.2.0, this number can be reconfigured to be in the range between 90 and 3600. The optional prefetch timeout value (in seconds) and max instance number can be specified in the configuration file: /fnsw/local/trigger/COR_prefetch.txt

MAX value must be between 75 and 200
TIMEOUT value must be between 90 and 3600

For example:

MAX   100
TIMEOUT  300

Under rare occasions where COR_Listen is launched without the -d option and not optionally configured, the timeout value is 7200 (two hours).
If a timeout is reached, a syslog entry is generated:

2010/10/25 11:06:29.096 155,18,4 <fnsw> COR_Listen -pt -s32769 -t3600 -d100 (1474810.42663.241 0x1680fa.a6a7) ...

COR_Listen Warning: client @ 9.39.40.29 [62784] did not send RPC.

Connection terminated by server due to timeout (180 secs). error = 0, socket = 8

**Add new utility "COR_set_prefetch_timeout"**

This new tool takes zero or one parameter. If run with no parameters, the tool displays the current value of the prefetch timeout. When run with one parameter, the tool dynamically sets the new timeout value in seconds for prefetch threads.

For example:

COR_set_prefetch_timeout display current timeout value

COR_set_prefetch_timeout 300 set timeout value to 300

The range of the timeout value must be between 90 and 3600. If the value specified is out of range, then it is not operative and the application exits with a value of 1. Having a discipline on the exit return value of the utility makes it scriptable.

If run with -h or illegal parameters, it displays this usage message:

Usage: COR_set_prefetch_timeout [time in seconds]
Add optional maximum prefetch thread count

The maximum count can be between 75 and 200.

See Add optional timeout value for prefetch threads for configuration details.

ssn

The ssn file contains the current system serial number. The valid range is 1000 - 2147483646.

Important

Never modify the contents of the ssn file. The ssn is written onto all storage media, and 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.

```
12620417
```

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

**UNIX**

/fnsw/bin

**WIN**

\fnsw\bin

---

**Attention**

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

---

An IS 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 serverConfig 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 BESI 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 server-Config 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.).
3 Applications
Application Program Descriptions

**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.
3 Applications
Application Program Descriptions

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 can run as determined by the number of remote domains that are configured for remote committal.

sas

Support and development personnel use the sas program to determine if session timeouts are working properly.

Important

The sas tool should only be run for diagnostics and troubleshooting. Do not run this tool without the advice and consent of your IBM FileNet technical support representative.

The sas tool uses the following syntax:
3 Applications
Application Program Descriptions

sas { –d | –n | –s | –a }
  –d     Set session timeout to 0 (debug)
  –n     Set session timeout to 600 (normal)
  –s     Print session statistics
  –p     Print sessions active
  –a     Print all sessions

Stop the IS software before setting the session timeout value with the
–d or –n options.

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 INXI 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.
3 Applications
Application Program Descriptions

INXs
The INXs program is a server stub program. This program receives remote procedure calls, does server-side session handling, and calls INXI 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.
3 Applications
Application Program Descriptions

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.
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_trn

The dtp_trn 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 server-Config file.
3 Applications
Application Program Descriptions

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 from 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 88.

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.

timeoutmonitor
The timeoutmonitor program monitors timeout waits on interlocks that are used by FileNet Image Services processes. The timeoutmonitor program, which is started automatically, must be running on all servers running FileNet Image Services and ISTK software.
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.

**Tip** 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.

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

The programs used by the FileNet 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 FileNet 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.

<table>
<thead>
<tr>
<th>Platform</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>AIX/6000</td>
<td>BES</td>
</tr>
<tr>
<td>HP-UX</td>
<td>libBES.sl</td>
</tr>
<tr>
<td>Solaris</td>
<td>libBES.so</td>
</tr>
<tr>
<td>Windows</td>
<td>BES.dll</td>
</tr>
</tbody>
</table>
Shared library descriptions

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

**Important**

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 84.
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 FileNet 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.
4 Shared libraries
Shared library descriptions

**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.

**DLII**

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 FileNet 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.
4 Shared libraries

Shared library descriptions

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.
**INXI**

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.
4 Shared libraries

Shared library descriptions

SSD
SSD provides interface between SECI 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.
Shared library descriptions

**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.
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:

- annot_log
- annotations
- cluster_map
- docs
- family_disk
- family_locator
- lib_surfaces
- od_stats
- remote_family
- scalar_numbers
- surf_dyn_info
- surf_info
- surf_locator
- surf_stat_info
- surface_activity
5 Permanent database tables and contents

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.

annot_log

The annot_log (annotation export log) table maintains a record of each annotation that is exported to the Content Engine of a P8 system in a Content Federation Services for FileNet Image Services (CFS-IS) environment. The annotation text is also stored in the annotations table.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>seq_key</td>
<td>Unique search key composed of ce_os_id (Content Engine Object Store ID), seq1, and seq2 keys - (group).</td>
</tr>
<tr>
<td>cd_os_id</td>
<td></td>
</tr>
<tr>
<td>seq1</td>
<td></td>
</tr>
<tr>
<td>seq2</td>
<td></td>
</tr>
<tr>
<td>doc_id</td>
<td>The identification number of the document that contains the annotation.</td>
</tr>
<tr>
<td>page</td>
<td>The page number on which the annotation occurs.</td>
</tr>
<tr>
<td>ssn</td>
<td>The serial number of the FileNet Image Services system on which the document originated.</td>
</tr>
<tr>
<td>annot_id</td>
<td>A system-assigned identification number for the annotation, based on the document identification number, page number, and ssn of the document. The first annotation on a page always has a value of 1 in this column.</td>
</tr>
<tr>
<td>action</td>
<td>The action performed on the annotation:</td>
</tr>
<tr>
<td></td>
<td>1 = insert</td>
</tr>
<tr>
<td></td>
<td>2 = export</td>
</tr>
<tr>
<td></td>
<td>3 = delete</td>
</tr>
<tr>
<td></td>
<td>4 = update</td>
</tr>
<tr>
<td>sec_rd</td>
<td>The id of the group that has read access to the annotation.</td>
</tr>
</tbody>
</table>
The annotations table maintains a record of each annotation and its associated document and page. The actual annotation text is stored in a hexadecimal format.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>annot_key</td>
<td>Unique search key composed of doc_id, page, ssn, and annot_id keys - (group).</td>
</tr>
<tr>
<td>doc_id</td>
<td>The identification number of the document that contains the annotation.</td>
</tr>
<tr>
<td>page</td>
<td>The page number of the document.</td>
</tr>
<tr>
<td>ssn</td>
<td>The serial number of the FileNet Image Services system on which the document originated.</td>
</tr>
<tr>
<td>annot_id</td>
<td>A system-assigned identification number for the annotation, based on the document identification number, page number, and ssn of the document. The first annotation on a page always has a value of 1 in this column.</td>
</tr>
<tr>
<td>create_date</td>
<td>The date and time that the annotation was created.</td>
</tr>
</tbody>
</table>
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]....0x04000200010100040000000002000400000000c0001020300
```

### cluster_map

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

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cluster_key</td>
<td>Unique search key composed of cluster_space and cluster_id keys - (group).</td>
</tr>
<tr>
<td>cluster_space</td>
<td>0,1,... Identifies the cluster number of the surface.</td>
</tr>
</tbody>
</table>
The following example shows a typical record from the `cluster_map` table:

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

docs

The `docs` 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.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>doc_id</td>
<td>Unique search key - document id number.</td>
</tr>
<tr>
<td>status_1</td>
<td>NULL or 0 = normal</td>
</tr>
<tr>
<td></td>
<td>1 = bad (unreadable)</td>
</tr>
<tr>
<td></td>
<td>2 = deleted</td>
</tr>
<tr>
<td></td>
<td>3 = not migrated to optical disk</td>
</tr>
<tr>
<td>status_2</td>
<td>NULL or 0 = normal</td>
</tr>
<tr>
<td></td>
<td>1 = bad (unreadable)</td>
</tr>
<tr>
<td>pages</td>
<td>The number of pages in the document.</td>
</tr>
<tr>
<td>contig_sectors</td>
<td>The number of contiguous optical disk sectors in the last page of the document.</td>
</tr>
<tr>
<td>Field Name</td>
<td>Field Description, Continued</td>
</tr>
<tr>
<td>-----------------</td>
<td>-----------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>surface_id_1</td>
<td>The surface number of the “family” disk for this document.</td>
</tr>
<tr>
<td>offset_1</td>
<td>The sector of the “family” disk containing this document’s long descriptor file.</td>
</tr>
<tr>
<td>surface_id_2</td>
<td>The surface number of the “tranlog” disk for this document.</td>
</tr>
<tr>
<td>offset_2</td>
<td>The sector of the “tranlog” disk containing this document’s long descriptor file.</td>
</tr>
<tr>
<td>security_len</td>
<td>The number of bytes in the security_info column.</td>
</tr>
<tr>
<td>security_info</td>
<td>Encoded access restrictions - read, write, execute/append.</td>
</tr>
<tr>
<td>orig_doc</td>
<td>Unique search key composed of orig_ssn and orig_doc_id keys - (group).</td>
</tr>
<tr>
<td>orig_ssn</td>
<td>System serial number of the system the document was originally from. This number is NULL if</td>
</tr>
<tr>
<td></td>
<td>the documents are local.</td>
</tr>
<tr>
<td>orig_doc_id</td>
<td>Document id number on original system. This number is NULL if the documents are local.</td>
</tr>
<tr>
<td>cache_id</td>
<td>Cache id when not on optical disk.</td>
</tr>
<tr>
<td>back_contig</td>
<td>Number of contiguous sectors (page 0 last).</td>
</tr>
<tr>
<td>duration</td>
<td>Time at which the object is to be written to optical disk. This field was added to accommodate</td>
</tr>
<tr>
<td></td>
<td>delayed migration.</td>
</tr>
<tr>
<td>ce_os_id</td>
<td>Content Engine Object Store id</td>
</tr>
<tr>
<td>sds_id and date</td>
<td>SDS object id and date composed of sds_id and sds_obj_date (group)</td>
</tr>
<tr>
<td>sds_id</td>
<td>SDS unit number</td>
</tr>
<tr>
<td>sds_obj_date</td>
<td>Time at which the SDS object was archived to the ISDS device.</td>
</tr>
<tr>
<td>sds_obj_id_len</td>
<td>Length of the SDS object id number.</td>
</tr>
<tr>
<td>sds_obj_id</td>
<td>SDS object id number. (Centera or DR550 CLIP ID)</td>
</tr>
<tr>
<td>sds_obj_blob_sz</td>
<td>Size of an SDS object blob (binary large object).</td>
</tr>
<tr>
<td>sds_orig_doc_id</td>
<td>SDS document id number on original system. Indicates a foreign document. This number is NULL</td>
</tr>
<tr>
<td></td>
<td>if the document is local.</td>
</tr>
</tbody>
</table>
The following example shows a typical record from the `docs` table.

```
<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Description, Continued</th>
</tr>
</thead>
<tbody>
<tr>
<td>sds_retent_typ</td>
<td>SDS retention type</td>
</tr>
<tr>
<td></td>
<td>1 = none</td>
</tr>
<tr>
<td></td>
<td>2 = infinite</td>
</tr>
<tr>
<td></td>
<td>3 = chronological (f_deletedate or f_archivedate is set)</td>
</tr>
<tr>
<td></td>
<td>4 = only on Centera with ARM (EBR + EBR supported device)</td>
</tr>
<tr>
<td></td>
<td>5 = virtual EBR (EBR + non-EBR supported device)</td>
</tr>
<tr>
<td>sds_retent_value</td>
<td>SDS retention expiration date and time.</td>
</tr>
<tr>
<td></td>
<td>Not set for none (1), infinite (2), and EBR + EBR supported device (4).</td>
</tr>
<tr>
<td></td>
<td>Always set for chronological (3) and set initially to default setting for virtual EBR (5).</td>
</tr>
<tr>
<td>sds_held</td>
<td>If set to 1, document is held.</td>
</tr>
<tr>
<td>sds_migrating</td>
<td>If set to 1, document is migrating to SDS.</td>
</tr>
<tr>
<td>last_annot_id</td>
<td>If non-zero, tracks the last annotation id.</td>
</tr>
</tbody>
</table>
```

The following example shows a record from the `docs` table on a system with an Hitachi Content Archive Platform (HCAP) SDS device.

```
doc_id............122177  status_1.............3  pages.............5
surface_id_1........1002  offset_1.............0  status_2.............0
offset_2..............0  cache_id...............1  sds_id...............3
sds_obj_blob_sz.........7348  sds_retent_typ.......1
sds_obj_date............1223676236 => 10/10/2008 15:03:56
```
The following examples show records from the `docs` table on a system with an EMC Centera SDS device with Advanced Retention Management (ARM).

<table>
<thead>
<tr>
<th>doc_id</th>
<th>status_1</th>
<th>pages</th>
<th>contig_sectors</th>
<th>surface_id_1</th>
<th>offset_1</th>
<th>cache_id</th>
<th>sds_id</th>
<th>sds_retent_typ</th>
<th>sds_obj_id_len</th>
<th>sds_obj_blob_sz</th>
<th>sds_obj_date</th>
<th>sds_obj_id</th>
</tr>
</thead>
<tbody>
<tr>
<td>.104845</td>
<td></td>
<td></td>
<td>2</td>
<td>1002</td>
<td>0</td>
<td></td>
<td>2</td>
<td>3</td>
<td>53</td>
<td>1952</td>
<td>1215541164 =&gt; 07/08/2008 11:19:24</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>4</td>
<td>53</td>
<td>1952</td>
<td>1215541164 =&gt; 07/08/2008 11:19:24</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>doc_id</th>
<th>status_1</th>
<th>pages</th>
<th>contig_sectors</th>
<th>surface_id_1</th>
<th>offset_1</th>
<th>cache_id</th>
<th>sds_id</th>
<th>sds_retent_typ</th>
<th>sds_obj_id_len</th>
<th>sds_obj_blob_sz</th>
<th>sds_obj_date</th>
<th>sds_obj_id</th>
</tr>
</thead>
<tbody>
<tr>
<td>.104846</td>
<td></td>
<td></td>
<td>2</td>
<td>3044</td>
<td>4002</td>
<td></td>
<td>2</td>
<td>4</td>
<td>53</td>
<td>1952</td>
<td>1215541164 =&gt; 07/08/2008 11:19:24</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0</td>
<td>0</td>
<td></td>
<td>2</td>
<td>4</td>
<td>53</td>
<td>1952</td>
<td>1215541164 =&gt; 07/08/2008 11:19:24</td>
<td></td>
</tr>
</tbody>
</table>
The *family_disk* 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.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>family_name</td>
<td>User-assigned optical disk family name.</td>
</tr>
<tr>
<td>family_id</td>
<td>System-assigned optical disk family identification number.</td>
</tr>
<tr>
<td>sides</td>
<td>1 = single sided disk 2 = double sided disk</td>
</tr>
<tr>
<td>interleaved</td>
<td>Write this number of side A’s before side B’s.</td>
</tr>
<tr>
<td>des_cur_surfs</td>
<td>Desired number of current optical disk surfaces that are available for a write operation.</td>
</tr>
<tr>
<td>num_cur_surfs</td>
<td>Actual number of current optical disk surfaces that are available for a write operation.</td>
</tr>
<tr>
<td>current_surfs</td>
<td>Current optical disk surface number available for a write operation.</td>
</tr>
<tr>
<td>prev_surfs</td>
<td>The optical disk surface number previously written to. (Used when interleaving).</td>
</tr>
<tr>
<td>preferred_osar</td>
<td>ID (a through h) of the storage library that contains the optical disk called out in the “current_surfs” column.</td>
</tr>
<tr>
<td>num_futr_surfs</td>
<td>The number of future surfaces reserved for this optical disk family.</td>
</tr>
<tr>
<td>future_surfs</td>
<td>Future optical disk surface number reserved for this family.</td>
</tr>
<tr>
<td>nextOrdinal</td>
<td>(not used)</td>
</tr>
<tr>
<td>disk_type</td>
<td>0 = unspecified disk type 1 or NULL = Hitachi 12” 2.6 GB WORM 2 = Hitachi 5” 600 MB WORM 3 = Hitachi 5” 600 MB erasable 4 = OSI (LMSI) 12” 2 GB WORM 5 = Hitachi 12” 7 GB high density WORM</td>
</tr>
<tr>
<td>Field Name</td>
<td>Field Description, Continued</td>
</tr>
<tr>
<td>------------</td>
<td>------------------------------</td>
</tr>
<tr>
<td>6</td>
<td>OSI (LMSI) 12” 5.5 GB high density WORM</td>
</tr>
<tr>
<td>7</td>
<td>HP 5” erasable</td>
</tr>
<tr>
<td>8</td>
<td>HP 5” WORM</td>
</tr>
<tr>
<td>9</td>
<td>Standard 5” 1.3 GB Erasable</td>
</tr>
<tr>
<td>10</td>
<td>Standard 5” 1.3 GB WORM</td>
</tr>
<tr>
<td>11</td>
<td>LMSI 12” 12 GB LM6000 WORM</td>
</tr>
<tr>
<td>12</td>
<td>Standard 5” 2.6 GB Erasable</td>
</tr>
<tr>
<td>13</td>
<td>Standard 5” 2.6 GB WORM</td>
</tr>
<tr>
<td>14</td>
<td>IBM 5” 2.6 GB permanent WORM</td>
</tr>
<tr>
<td>15</td>
<td>Standard 5” 5.2 GB Erasable</td>
</tr>
<tr>
<td>16</td>
<td>Standard 5” 5.2 GB WORM</td>
</tr>
<tr>
<td>17</td>
<td>IBM 5” 5.2 GB permanent WORM</td>
</tr>
<tr>
<td>18</td>
<td>LMSI 12” 30 GB LM8000 WORM</td>
</tr>
<tr>
<td>19</td>
<td>Standard 5” 9.1 GB Erasable</td>
</tr>
<tr>
<td>20</td>
<td>Standard 5” 9.1 GB WORM</td>
</tr>
<tr>
<td>21</td>
<td>MSAR 1 GB</td>
</tr>
<tr>
<td>22</td>
<td>MSAR 2 GB</td>
</tr>
<tr>
<td>23</td>
<td>MSAR 4 GB</td>
</tr>
<tr>
<td>24</td>
<td>MSAR 8 GB</td>
</tr>
<tr>
<td>25</td>
<td>MSAR 16 GB</td>
</tr>
<tr>
<td>26</td>
<td>MSAR 32 GB</td>
</tr>
<tr>
<td>27</td>
<td>5” UDO 30 GB Erasable disk</td>
</tr>
<tr>
<td>28</td>
<td>5” UDO 30 GB WORM disk</td>
</tr>
<tr>
<td>29</td>
<td>5” UDO 60 GB Erasable disk</td>
</tr>
<tr>
<td>30</td>
<td>5” UDO 60 GB WORM disk</td>
</tr>
</tbody>
</table>

**is_primary**

Is this the primary family?
0 = False (transaction)
1 = True (primary)

**prev_0_surfs**

Previous surfaces for cur_surf (current surfaces).

**tran_families**

Family ids of the tranlogs for the family. If all zero, no tran logging.
5 Permanent database tables and contents

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**

The `family_locator` 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.

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

**lib_surfaces**

The **lib_surfaces** table is used to keep track of the location of optical drive surfaces.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>surface_id</td>
<td>Optical disk surface id number.</td>
</tr>
<tr>
<td>loc_key</td>
<td>The location key of the surface, composed of the library id, unit_indicator, and unit number - (group). This field is NULL when the surface is “out of box.”</td>
</tr>
<tr>
<td>library</td>
<td>The library identifier</td>
</tr>
<tr>
<td>unit_indicator</td>
<td>The unit indicator</td>
</tr>
<tr>
<td></td>
<td>1 = lsi_in_slot</td>
</tr>
<tr>
<td></td>
<td>2 = lsi_in_drive</td>
</tr>
<tr>
<td></td>
<td>3 = lsi_in_gripper</td>
</tr>
<tr>
<td>unit_number</td>
<td>The unit number is composed of the slot number, drive number and gripper number.</td>
</tr>
</tbody>
</table>
The `od_stats` table 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.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>surface_id</td>
<td>Optical disk surface number that incurred an error.</td>
</tr>
<tr>
<td>osardrive</td>
<td>Unique search key composed of osar_num and drive_num keys - (group).</td>
</tr>
<tr>
<td>osar_num</td>
<td>The number of the storage library that incurred the error.</td>
</tr>
<tr>
<td>drive_num</td>
<td>The optical drive which incurred the error.</td>
</tr>
<tr>
<td>num_rewrites</td>
<td>The number of 1k sectors that incurred an error during a write operation.</td>
</tr>
<tr>
<td>last_day_sec</td>
<td>The date and time of the last I/O error that occurred.</td>
</tr>
<tr>
<td>num_elt</td>
<td>The number of elements expressed in the days_errors and total_errors columns.</td>
</tr>
</tbody>
</table>
The presence of records in the `remote_family` table indicates that committal operations must be done on the remote domain and family name indicated in addition to the local family.

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

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

The scalar_numbers 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).

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

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

The surf_dyn_info 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.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>surface_id</td>
<td>Unique search key - optical disk surface id number.</td>
</tr>
<tr>
<td>next_sector</td>
<td>Next available sector on the optical disk surface that can be written to. This is also known as the high-water mark.</td>
</tr>
<tr>
<td>num_act_docs</td>
<td>Number of retrievable documents already on optical disk surface. See the important note at the end of this table.</td>
</tr>
<tr>
<td>num_del_docs</td>
<td>Number of non-retrievable (deleted) documents on optical disk surface.</td>
</tr>
<tr>
<td>num_clusters</td>
<td>Number of clusters involved with this optical disk surface.</td>
</tr>
<tr>
<td>last_desc_sect</td>
<td>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.</td>
</tr>
<tr>
<td>nxt_short_desc</td>
<td>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.</td>
</tr>
<tr>
<td>num_unwrt_desc</td>
<td>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.</td>
</tr>
<tr>
<td>short_doc_ids</td>
<td>The document id numbers of the documents that have been written to disk, but have not yet had a short descriptor file written.</td>
</tr>
<tr>
<td>short_sects</td>
<td>The sector numbers of the long descriptor files that have not yet had a short descriptor file written.</td>
</tr>
<tr>
<td>num_used_sects</td>
<td>The number of 1024 byte sectors occupied on optical disk surface.</td>
</tr>
<tr>
<td>num_pages</td>
<td>Total number of document pages on the surface.</td>
</tr>
<tr>
<td>next_fid_sect</td>
<td>The next available sector that can be written to in the optical disk directory file.</td>
</tr>
<tr>
<td>last_fid_sect</td>
<td>The last available sector that can be written to in the optical disk directory file.</td>
</tr>
</tbody>
</table>
The following example shows a typical record from the `surf_dyn_info` table.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Description, Continued</th>
</tr>
</thead>
<tbody>
<tr>
<td>cur_fid_off</td>
<td>Offset in the directory of the current entry.</td>
</tr>
<tr>
<td>inbox_priority</td>
<td>Priority which keeps the surface in the storage library. This is only set for side 1 surfaces.</td>
</tr>
</tbody>
</table>

Important

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 following command 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

The `surf_locator` 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. This table is referenced during committal and document retrieval. Data I/O with this table is primarily accomplished through the use of the SLT shared library.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>surface_id</td>
<td>Optical disk surface id number.</td>
</tr>
<tr>
<td>server_id</td>
<td>System assigned number of the Storage Library server controlling the library that contains this optical disk.</td>
</tr>
<tr>
<td>orig_surf</td>
<td>Unique search key composed of orig_ssn and orig_surf_id keys - (group).</td>
</tr>
<tr>
<td>orig_ssn</td>
<td>If imported, the original system serial number that this disk came from.</td>
</tr>
<tr>
<td>orig_surf_id</td>
<td>ID of the original surface. This column is used in conjunction with “orig_ssn” for certain search operations.</td>
</tr>
<tr>
<td>fam_id</td>
<td>Family ID of the surface.</td>
</tr>
</tbody>
</table>
### Field Name | Field Description, Continued
--- | ---
disk_type | 0 = unspecified disk type
1 or NULL = Hitachi 12” 2.6 GB WORM
2 = Hitachi 5” 600 MB WORM
3 = Hitachi 5” 600 MB erasable
4 = OSI (LMSI) 12” 2 GB WORM
5 = Hitachi 12” 7 GB high density WORM
6 = OSI (LMSI) 12” 5.5 GB high density WORM
7 = HP 5” erasable
8 = HP 5” WORM
9 = Standard 5” 1.3 GB Erasable
10 = Standard 5” 1.3 GB WORM
11 = LMSI 12” 12 GB LM6000 WORM
12 = Standard 5” 2.6 GB Erasable
13 = Standard 5” 2.6 GB WORM
14 = IBM 5” 2.6 GB permanent WORM
15 = Standard 5” 5.2 GB Erasable
16 = Standard 5” 5.2 GB WORM
17 = IBM 5” 5.2 GB permanent WORM
18 = LMSI 12” 30 GB LM8000 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
27 = 5” UDO 30 GB Erasable disk
28 = 5” UDO 30 GB WORM disk
29 = 5” UDO 60 GB Erasable disk
30 = 5” UDO 60 GB WORM disk
num_act_docs | Number of documents for each side.
surf_stat_info

The surf_stat_info 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.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>surface_id</td>
<td>Optical disk surface id number.</td>
</tr>
<tr>
<td>family_id</td>
<td>System assigned optical disk family id number for the surface.</td>
</tr>
<tr>
<td>series_ordinal</td>
<td>Used to preformat disks.</td>
</tr>
</tbody>
</table>
| 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 composed 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. |
| need_verify    | Whether the imported disk was verified.                |
## 5 Permanent database tables and contents

### Tables and contents

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Description, Continued</th>
</tr>
</thead>
<tbody>
<tr>
<td>disk_type</td>
<td>0 = unspecified disk type</td>
</tr>
<tr>
<td></td>
<td>1 or NULL = Hitachi 12” 2.6 GB WORM</td>
</tr>
<tr>
<td></td>
<td>2 = Hitachi 5” 600 MB WORM</td>
</tr>
<tr>
<td></td>
<td>3 = Hitachi 5” 600 MB erasable</td>
</tr>
<tr>
<td></td>
<td>4 = OSI (LMSI) 12” 2 GB WORM</td>
</tr>
<tr>
<td></td>
<td>5 = Hitachi 12” 7 GB high density WORM</td>
</tr>
<tr>
<td></td>
<td>6 = OSI (LMSI) 12” 5.5 GB high density WORM</td>
</tr>
<tr>
<td></td>
<td>7 = HP 5” erasable</td>
</tr>
<tr>
<td></td>
<td>8 = HP 5” WORM</td>
</tr>
<tr>
<td></td>
<td>9 = Standard 5” 1.3 GB Erasable</td>
</tr>
<tr>
<td></td>
<td>10 = Standard 5” 1.3 GB WORM</td>
</tr>
<tr>
<td></td>
<td>11 = LMSI 12” 12 GB LM6000 WORM</td>
</tr>
<tr>
<td></td>
<td>12 = Standard 5” 2.6 GB Erasable</td>
</tr>
<tr>
<td></td>
<td>13 = Standard 5” 2.6 GB WORM</td>
</tr>
<tr>
<td></td>
<td>14 = IBM 5” 2.6 GB permanent WORM</td>
</tr>
<tr>
<td></td>
<td>15 = Standard 5” 5.2 GB Erasable</td>
</tr>
<tr>
<td></td>
<td>16 = Standard 5” 5.2 GB WORM</td>
</tr>
<tr>
<td></td>
<td>17 = IBM 5” 5.2 GB permanent WORM</td>
</tr>
<tr>
<td></td>
<td>18 = LMSI 12” 30 GB LM8000 WORM</td>
</tr>
<tr>
<td></td>
<td>19 = Standard 5” 9.1 GB Erasable</td>
</tr>
<tr>
<td></td>
<td>20 = Standard 5” 9.1 GB WORM</td>
</tr>
<tr>
<td></td>
<td>21 = MSAR 1 GB</td>
</tr>
<tr>
<td></td>
<td>22 = MSAR 2 GB</td>
</tr>
<tr>
<td></td>
<td>23 = MSAR 4 GB</td>
</tr>
<tr>
<td></td>
<td>24 = MSAR 8 GB</td>
</tr>
<tr>
<td></td>
<td>25 = MSAR 16 GB</td>
</tr>
<tr>
<td></td>
<td>26 = MSAR 32 GB</td>
</tr>
<tr>
<td></td>
<td>27 = 5” UDO 30 GB Erasable disk</td>
</tr>
<tr>
<td></td>
<td>28 = 5” UDO 30 GB WORM disk</td>
</tr>
<tr>
<td></td>
<td>29 = 5” UDO 60 GB Erasable disk</td>
</tr>
<tr>
<td></td>
<td>30 = 5” UDO 60 GB WORM disk</td>
</tr>
</tbody>
</table>
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......63840831 => 99/3/26 11:47:11
```

**surface_activity**

The `surface_activity` 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.
<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Description, Continued</th>
</tr>
</thead>
<tbody>
<tr>
<td>mounts</td>
<td>Number of mounts of this surface on this day.</td>
</tr>
<tr>
<td>reads</td>
<td>Number of reads for surface on this day.</td>
</tr>
<tr>
<td>writes</td>
<td>Number of writes for surface on this day.</td>
</tr>
</tbody>
</table>
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:

- **batch tables**
  - batch_ctl
  - batch_doc
  - batch_dyn_hdr
  - batch_hdr
  - batch_image
  - batch_ixdir
  - batch_ixval
  - batch_stat_hdr
  - batch_data
  - batch_folder

- **cache tables**
  - csm_caches
  - csm_free_space
  - csm_temp_id
  - csm_used_space

- **print tables**
  - print_docs
  - print_options
  - print_svcs
  - print_reqopt
  - print_requests

- **requests tables**
  - bkg_request
  - write_request

The tables are detailed in sections separated according to the FileNet 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**

The **batch_ctl** 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.

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

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

```
ctl_index  batch_id  batch_name_id  batch_open_id
--------  --------  -------------  --------------
   1       11             6         757791177
```
The `batch_doc` table 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.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>doc_index</td>
<td>Unique search key composed of batch_id and doc_id keys - (group).</td>
</tr>
<tr>
<td>batch_id</td>
<td>Identification number of the batch.</td>
</tr>
<tr>
<td>doc_id</td>
<td>The document id number.</td>
</tr>
</tbody>
</table>
| 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 131. |
| phase_error    | 0 = no error  
|                | 1 = error occurred while performing this “phase” of the document.               |
| phase_status   | The current status for each of the phases that the document could experience. See the table “Phase Status” on page 131. |
| 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. |
The following example shows a typical record from the \texttt{batch\_doc} table:

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Description, Continued</th>
</tr>
</thead>
<tbody>
<tr>
<td>cluster_id</td>
<td>Identification number of the cluster that is involved. Value is 0 if clustering is not involved.</td>
</tr>
<tr>
<td>ocr_schema</td>
<td>Reserved for use by OCR.</td>
</tr>
<tr>
<td>text_len</td>
<td>The number of bytes in the text_data column.</td>
</tr>
<tr>
<td>text_data</td>
<td>Reserved. If the value of text_len = 0, the field is not displayed.</td>
</tr>
</tbody>
</table>

\texttt{batch\_id} ...............8 \texttt{doc\_id} ...............5 \texttt{doc\_type} ...............0
\texttt{num\_pages} ..............2 \texttt{num\_indices} ...........1 \texttt{num\_reqd\_indcs} ........ 1
\texttt{phase\_num} ..............0 \texttt{phase\_error} ............0 \texttt{fam\_id} ...............0
\texttt{cluster\_space} ...........0 \texttt{ocr\_schema} ............0 \texttt{text\_len} ............0
\texttt{phase\_status} [0]...0,0,0,0,0,3,0,0,0,0,0,0
\texttt{cluster\_id} [0]...0x000000000000

\textbf{batch\_dyn\_hdr}

A dynamic record is created within the \texttt{batch\_dyn\_hdr} 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.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>batch_id</td>
<td>Unique search key for the table - batch identification number.</td>
</tr>
<tr>
<td>batch_name</td>
<td>Name of the batch entered by the user at the start of document entry.</td>
</tr>
<tr>
<td>exp_pages</td>
<td>Expected number of pages to be scanned in for the batch.</td>
</tr>
<tr>
<td>exp_docs</td>
<td>Expected number of documents in the batch.</td>
</tr>
<tr>
<td>pages_per_doc</td>
<td>Number of pages/document in batch. 0 = varies per document.</td>
</tr>
<tr>
<td>Field Name</td>
<td>Field Description, Continued</td>
</tr>
<tr>
<td>---------------------</td>
<td>-----------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>double_sided</td>
<td>0 = single sided pages</td>
</tr>
<tr>
<td></td>
<td>1 = double sided pages</td>
</tr>
<tr>
<td>verify_images</td>
<td>0 = no image verify necessary</td>
</tr>
<tr>
<td></td>
<td>1 = image verification needed</td>
</tr>
<tr>
<td>verify_indices</td>
<td>0 = no index verify necessary</td>
</tr>
<tr>
<td></td>
<td>1 = index verification needed</td>
</tr>
<tr>
<td>committal_type</td>
<td>0 = do not migrate</td>
</tr>
<tr>
<td></td>
<td>1 = migrate to optical disk</td>
</tr>
<tr>
<td>batch_total</td>
<td>0 = batch totaling unnecessary</td>
</tr>
<tr>
<td></td>
<td>1 = batch totaling needed</td>
</tr>
<tr>
<td>this_phase</td>
<td>Current phase that the document is in. See the <a href="#">“Document Phases” on page 131</a> for values.</td>
</tr>
<tr>
<td>next_phase</td>
<td>Next phase for that document. See the <a href="#">“Document Phases” on page 131</a> for values.</td>
</tr>
<tr>
<td>phase_status</td>
<td>The current status for each of the eleven possible phases that the document could experience.</td>
</tr>
<tr>
<td></td>
<td>See the table <a href="#">“Phase Status” on page 131</a> for possible values.</td>
</tr>
<tr>
<td>phase_1st_time</td>
<td>The last time each of the eleven possible phases for this document was attempted.</td>
</tr>
<tr>
<td></td>
<td>See the table <a href="#">“Phase Status” on page 131</a> for possible values.</td>
</tr>
<tr>
<td>phase_user_id</td>
<td>ID number of operator who last worked on given phase.</td>
</tr>
<tr>
<td>sort_order</td>
<td>Sort order for the pages in cache.</td>
</tr>
<tr>
<td></td>
<td>0 = Sequential from Page 0</td>
</tr>
<tr>
<td></td>
<td>1 = Reverse order</td>
</tr>
<tr>
<td>ocr_enable</td>
<td>1 = OCR enabled</td>
</tr>
<tr>
<td>auto_index</td>
<td>Used by OCR</td>
</tr>
<tr>
<td>text_len</td>
<td>The number of characters in the text_data field.</td>
</tr>
<tr>
<td>text_data</td>
<td>Reserved.</td>
</tr>
<tr>
<td>migr_dest_name</td>
<td>Name of the cache that an object will be migrated to (other than page cache).</td>
</tr>
<tr>
<td>migrate_delay</td>
<td>Number of seconds to delay before migrating an object from the cache to the optical disk.</td>
</tr>
<tr>
<td></td>
<td>Only meaningful if committal type is COMMIT_NORMAL. A value of 4294967295 = do not migrate.</td>
</tr>
</tbody>
</table>
The following example shows a typical record from the `batch_dyn_hdr` table:

```plaintext
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.

<table>
<thead>
<tr>
<th>Number</th>
<th>Phase</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Batch Creation</td>
</tr>
<tr>
<td>1</td>
<td>Scanning</td>
</tr>
<tr>
<td>2</td>
<td>Image Verification</td>
</tr>
<tr>
<td>3</td>
<td>Rescan</td>
</tr>
<tr>
<td>4</td>
<td>Assembly</td>
</tr>
<tr>
<td>5</td>
<td>Indexing</td>
</tr>
<tr>
<td>6</td>
<td>Index Verification</td>
</tr>
<tr>
<td>7</td>
<td>Batch Totaling</td>
</tr>
<tr>
<td>8</td>
<td>Committal</td>
</tr>
<tr>
<td>9</td>
<td>Cataloging</td>
</tr>
<tr>
<td>10</td>
<td>Recommit</td>
</tr>
</tbody>
</table>

Phase Status

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

<table>
<thead>
<tr>
<th>Number</th>
<th>Phase</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Not started</td>
</tr>
<tr>
<td>1</td>
<td>In progress</td>
</tr>
<tr>
<td>2</td>
<td>Interrupted</td>
</tr>
<tr>
<td>3</td>
<td>Completed</td>
</tr>
<tr>
<td>4</td>
<td>Unnecessary</td>
</tr>
<tr>
<td>5</td>
<td>Error occurred</td>
</tr>
<tr>
<td>6</td>
<td>Optional</td>
</tr>
</tbody>
</table>
**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.

```sql
select batch_hdr * where batch_id = b7
```

**batch_image**

The `batch_image` 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.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>image_index</td>
<td>Unique search key composed of batch_id and image_id keys - (group).</td>
</tr>
<tr>
<td>batch_id</td>
<td>Identification number of the batch. Searches on this column can be done in conjunction with the &quot;image id&quot; column.</td>
</tr>
<tr>
<td>image_id</td>
<td>Image id number.</td>
</tr>
<tr>
<td>doc_index</td>
<td>Unique search key composed of batch_id2, doc_id, and page_id keys - (group).</td>
</tr>
<tr>
<td>batch_id2</td>
<td>Same value as that of the &quot;batch_id&quot; column. However searches on this column can be done in conjunction with the columns &quot;doc_id&quot; and &quot;page_id&quot;.</td>
</tr>
<tr>
<td>doc_id</td>
<td>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.</td>
</tr>
<tr>
<td>page_id</td>
<td>Page number of a specific document.</td>
</tr>
<tr>
<td>image_length</td>
<td>The size of the image expressed in bytes.</td>
</tr>
<tr>
<td>Field Name</td>
<td>Field Description, Continued</td>
</tr>
<tr>
<td>---------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| 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.                                                                                                           |
| index_value   | This column is used in conjunction with images that contain text data. The indexing information is displayed in hexadecimal form.                          |
| text_len      | The number of bytes in the text_data column.                                                                                                               |
| text_data     | Reserved                                                                                                                                                    |

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

```
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
```
The **batch_ixdir** table 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.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ixdir_index</td>
<td>The batch and index ids. Composed of batch_id and index_id - (group).</td>
</tr>
<tr>
<td>batch_id</td>
<td>Identification number of the batch.</td>
</tr>
<tr>
<td>index_id</td>
<td>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.)</td>
</tr>
<tr>
<td>index_name</td>
<td>The name of the indexing field.</td>
</tr>
<tr>
<td>index_type</td>
<td>The type of index field. 49 = numeric 50 = string 52 = menu 56 = date</td>
</tr>
<tr>
<td>max_string_len</td>
<td>This field only applies if the index_type is 50 which is the maximum number of characters for a string index.</td>
</tr>
<tr>
<td>menu_name</td>
<td>The name of an indexing menu field.</td>
</tr>
<tr>
<td>verify_flag</td>
<td>0 = no index verify necessary 1 = index verification needed</td>
</tr>
<tr>
<td>required_flag</td>
<td>0 = optional 1 = index info must be entered</td>
</tr>
<tr>
<td>batch_tot_flag</td>
<td>0 = batch totaling unnecessary 1 = batch totaling needed</td>
</tr>
<tr>
<td>up_case_flag</td>
<td>0 = no auto uppercase conversion of string data 1 = Convert string data automatically to uppercase when cataloging.</td>
</tr>
</tbody>
</table>
6 Transient database tables and contents
Batch tables and contents

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

```plaintext
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**

The **batch_ixval** 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.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ixval_index</td>
<td>Unique search key composed of batch_id, doc_id, and index_id keys - (group).</td>
</tr>
<tr>
<td>batch_id</td>
<td>Identification number of the batch.</td>
</tr>
<tr>
<td>doc_id</td>
<td>The document id number.</td>
</tr>
<tr>
<td>index_id</td>
<td>Index id number from the Index database table doctaba.</td>
</tr>
<tr>
<td>index_type</td>
<td>The type of index field.</td>
</tr>
<tr>
<td></td>
<td>49 = numeric</td>
</tr>
<tr>
<td></td>
<td>50 = string</td>
</tr>
<tr>
<td></td>
<td>52 = menu</td>
</tr>
<tr>
<td></td>
<td>56 = date</td>
</tr>
<tr>
<td>index_len</td>
<td>The number of bytes in the index_value column.</td>
</tr>
<tr>
<td>index_value</td>
<td>The index information (in hex).</td>
</tr>
<tr>
<td>data_status</td>
<td>Reserved for use by OCR.</td>
</tr>
<tr>
<td>text_len</td>
<td>The number of bytes in the text_data column.</td>
</tr>
<tr>
<td>text_data</td>
<td>Reserved. If the value of text_len = 0, the field is not displayed.</td>
</tr>
</tbody>
</table>

The following example shows a typical record from the `batch_ixval` table:

```
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
```
A general record is created within the `batch_stat_hdr` 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.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>batch_id</td>
<td>Batch identification number.</td>
</tr>
<tr>
<td>queue_index</td>
<td>Non-unique search key composed of queue and qtime keys - (group).</td>
</tr>
</tbody>
</table>
| queue            | 0 = not queued  
|                  | 1 = Uncommit queue  
|                  | 2 = queued for committal  
|                  | 3 = queue in progress                                            |
| qtime            | When the batch was queued for committal.                         |
| service_name     | Organization/domain/object.                                      |
| open_time        | When the batch was opened.                                       |
| open_user_id     | Identification number of the operator that opened the batch.     |
| open_flag        | 0 = batch not currently open  
<p>|                  | 1 = batch open                                                   |
| create_time      | When the batch was created.                                      |
| 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.|
| doc_class_name   | Document class name under which this batch was created.          |
| 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.                 |
| indexing_form    | Name of the indexing form used with the document class.          |</p>
<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Description, Continued</th>
</tr>
</thead>
<tbody>
<tr>
<td>access_restrct</td>
<td>Default doc security</td>
</tr>
<tr>
<td>cluster_space</td>
<td>Identification number of cluster operation that is involved. This feature is currently not used as denoted by the number 65535.</td>
</tr>
<tr>
<td>cluster_index</td>
<td>Cluster column number.</td>
</tr>
<tr>
<td>fam_id</td>
<td>Optical disk family number for this document. 0=clustering</td>
</tr>
<tr>
<td>wfl_queue_name</td>
<td>WorkFlo queue name for documents entering the system.</td>
</tr>
<tr>
<td>wfl_sys_name</td>
<td>WorkFlo system name for documents entering the system.</td>
</tr>
<tr>
<td>retent_disp</td>
<td>How to handle document once retention offset is reached.</td>
</tr>
<tr>
<td></td>
<td>0 = delete</td>
</tr>
<tr>
<td></td>
<td>1 = archive</td>
</tr>
<tr>
<td>retent_base</td>
<td>When to start the retention offset “clock” running.</td>
</tr>
<tr>
<td></td>
<td>0 = date of closing</td>
</tr>
<tr>
<td></td>
<td>1 = date of entry</td>
</tr>
<tr>
<td>retent_offset</td>
<td>Number of months until the document will be deleted or archived.</td>
</tr>
<tr>
<td>act_pages</td>
<td>Total number of pages in the batch.</td>
</tr>
<tr>
<td>act_docs</td>
<td>Total number of documents in the batch.</td>
</tr>
<tr>
<td>next_doc_id</td>
<td>Next doc id to assign (0 relative).</td>
</tr>
<tr>
<td>target_ims</td>
<td>Indicates the target FileNet Image Services system in a cross committal system.</td>
</tr>
<tr>
<td></td>
<td>NULL = local</td>
</tr>
<tr>
<td>csum_state</td>
<td>Indicates the state of checksumming that is being used on this batch.</td>
</tr>
<tr>
<td></td>
<td>0 = unknown</td>
</tr>
<tr>
<td></td>
<td>1 = undetermined</td>
</tr>
<tr>
<td></td>
<td>2 = no checksumming</td>
</tr>
<tr>
<td></td>
<td>3 = checksumming</td>
</tr>
<tr>
<td></td>
<td>4 = resetting checksum</td>
</tr>
<tr>
<td>delete</td>
<td>0 = not marked for deletion</td>
</tr>
<tr>
<td></td>
<td>1 = delete fast batch</td>
</tr>
<tr>
<td></td>
<td>2 = delete normal batch</td>
</tr>
</tbody>
</table>
The following example shows a typical record from the **batch_stat_hdr** table:

```
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_indices.......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 => 11/1/5 08:20:38
create_time.....757619523 => 11/1/3 09:52:03
doc_class_name...."test"
indexing_form........"F_IXFORM1"
access_restrct[0]....0x000000010000000100000001
```

### batch_data

The **batch_data** table is the Batch services batch object data table. There is one entry in this table per object data.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>data_index</td>
<td>Unique search key composed of batch_id, object_type, object_id, and sequence_id keys - (group).</td>
</tr>
<tr>
<td>batch_id</td>
<td>Batch identifier.</td>
</tr>
</tbody>
</table>
6 Transient database tables and contents
Batch tables and contents

The `batch_folder` table is the Batch services folder node table. There is one entry in this table for each folder node.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>node_id</td>
<td>Folder node id.</td>
</tr>
<tr>
<td>relation</td>
<td>The parent and base node. Composed of parent_id and base_node_name - (group).</td>
</tr>
<tr>
<td>parent_id</td>
<td>Parent Identifier.</td>
</tr>
<tr>
<td>base_node_name</td>
<td>Base node name.</td>
</tr>
<tr>
<td>batch_id</td>
<td>Batch Identifier. 0= non-batch node. Batch id does not need to be unique.</td>
</tr>
<tr>
<td>last_update</td>
<td>Time stamp of creation or update of the node.</td>
</tr>
<tr>
<td>info_len</td>
<td>Length associated with the information.</td>
</tr>
<tr>
<td>info_data</td>
<td>Actual information.</td>
</tr>
</tbody>
</table>
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**

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

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cache_cache_id</td>
<td>System-assigned identification number for a specific type of cache.</td>
</tr>
<tr>
<td>cache_name_len</td>
<td>Number of characters in the cache’s name.</td>
</tr>
<tr>
<td>cache_name</td>
<td>Name of the cache.</td>
</tr>
<tr>
<td>inuse_objects</td>
<td>Total number of objects in use.</td>
</tr>
<tr>
<td>inuse_sectors</td>
<td>Total number of sectors in use.</td>
</tr>
<tr>
<td>locked_objects</td>
<td>Total number of locked objects currently in use.</td>
</tr>
<tr>
<td>locked_sectors</td>
<td>Total number of locked sectors currently in use.</td>
</tr>
</tbody>
</table>
The following example shows a typical record from the **csm_caches** table:

<table>
<thead>
<tr>
<th>cache_cache_id</th>
<th>cache_name_len</th>
<th>cache_name</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>11</td>
<td>&quot;page_cache1&quot;</td>
</tr>
<tr>
<td>6</td>
<td>16</td>
<td>&quot;sys_print_cache1&quot;</td>
</tr>
<tr>
<td>7</td>
<td>13</td>
<td>&quot;fillin_cache1&quot;</td>
</tr>
<tr>
<td>8</td>
<td>10</td>
<td>&quot;bes_cache1&quot;</td>
</tr>
<tr>
<td>9</td>
<td>16</td>
<td>&quot;app_print_cache1&quot;</td>
</tr>
</tbody>
</table>

**csm_free_space**

The **csm_free_space** 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.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>free_block</td>
<td>Unique search key composed of the size and group_start keys - (group).</td>
</tr>
<tr>
<td>size</td>
<td>The number of available contiguous sectors from a given start sector.</td>
</tr>
<tr>
<td>group_start</td>
<td>Same as &quot;start&quot;.</td>
</tr>
<tr>
<td>start</td>
<td>The last three bytes indicate the starting sector number (relative to the partition), of available free space.</td>
</tr>
</tbody>
</table>
The following example shows a typical record from the **csm_free_space** table:

<table>
<thead>
<tr>
<th>size</th>
<th>group_start</th>
<th>start</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0x010001cf</td>
<td>0x010001cf</td>
</tr>
<tr>
<td>1</td>
<td>0x0100052f</td>
<td>0x0100052f</td>
</tr>
<tr>
<td>9</td>
<td>0x0100033c</td>
<td>0x0100033c</td>
</tr>
<tr>
<td>37</td>
<td>0x01000277</td>
<td>0x01000277</td>
</tr>
<tr>
<td>45</td>
<td>0x01000711</td>
<td>0x01000711</td>
</tr>
<tr>
<td>38626</td>
<td>0x0100091e</td>
<td>0x0100091e</td>
</tr>
</tbody>
</table>

**csm_temp_id**

The **csm_temp_id** 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.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>temp_id</td>
<td>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.</td>
</tr>
</tbody>
</table>
The following example shows a typical record from the csm_temp_id table:

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

**csm_used_space**

The csm_used_space 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.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>object</td>
<td>Unique search key composed of the cache_id, ssn, obj_id, and page keys - (group).</td>
</tr>
<tr>
<td>cache_id</td>
<td>Number indicating the type of cache. The number can be cross referenced in the table “csm_caches”.</td>
</tr>
<tr>
<td>ssn</td>
<td>System serial number where document (or image) was created.</td>
</tr>
<tr>
<td>obj_id</td>
<td>Image or document id number.</td>
</tr>
<tr>
<td>page</td>
<td>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.</td>
</tr>
<tr>
<td>duration</td>
<td>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.</td>
</tr>
</tbody>
</table>
### Field Name | Field Description, Continued
--- | ---
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.
maxlen | Amount of space reserved for this image or page in the cache.
curlen | Current length. The size of this particular page (in bytes).
created | Date and time when the image (page), was scanned.
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.
temp_flag | Not used.  
0 = not temporary  
1 = temporary
sec_len | Length (in bytes) of the "sec_info"
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’.
refcnt | Reference count. The number of times this page has been accessed while in cache.
client_attr | Type of document, Image Form, Mixed etc.
seqnum | The seqnum field keeps track of how often an object has been opened. It is for internal IS use only.
umchunk | Number of contiguous chunks used to hold the object.
startary | The last three bytes of information indicate the starting sector number of the page relative to the partition.
lenary | Number of sectors used to hold the object.
csum | Check on the image. Value is no if there is no checksumming.
The following example shows a typical record from the *csm_used_space* table:

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Description, Continued</th>
</tr>
</thead>
<tbody>
<tr>
<td>lastupdate</td>
<td>Date and time that the object was last updated.</td>
</tr>
<tr>
<td>incomplete</td>
<td>Replaces temp_flag.</td>
</tr>
<tr>
<td></td>
<td>NULL or 0 = completed object</td>
</tr>
<tr>
<td></td>
<td>non-NUL = incomplete object</td>
</tr>
</tbody>
</table>

```
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 the print_docs table. One record exists for each document to be printed. Data I/O with this table is primarily accomplished through the PRII shared library.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>print_doc</td>
<td>Unique search key composed of the request_id and doc_order keys - (group).</td>
</tr>
<tr>
<td>request_id</td>
<td>System-assigned request identification number.</td>
</tr>
<tr>
<td>doc_order</td>
<td>The order in which the selected documents will be printed (starting from 0 for the first document).</td>
</tr>
<tr>
<td>doc_id</td>
<td>The document id number of the selected document.</td>
</tr>
<tr>
<td>first_page</td>
<td>Beginning page number of a document.</td>
</tr>
<tr>
<td>last_page</td>
<td>Ending page number of a document.</td>
</tr>
<tr>
<td>ssn</td>
<td>System serial number if the column “svc_type” has a value of 2.</td>
</tr>
</tbody>
</table>
| 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

The print_options 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.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>request_id</td>
<td>Unique search key - system-assigned request identification number.</td>
</tr>
<tr>
<td>print_time</td>
<td>Non-unique search key - time at which page fetching will begin.</td>
</tr>
<tr>
<td>priority</td>
<td>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.</td>
</tr>
</tbody>
</table>
| paper_size | Paper size to be used. NULL = default (8.5 x 11)
|            | 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)

orientation | NULL = default (does not appear)
1 = landscape
2 = portrait
3 = no rotation

annotations | 0 = no annotations
1 = print annotations

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.

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.

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.
## print_svcs

The **print_svcs** 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.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>svc_id</td>
<td>Unique search key - TBD</td>
</tr>
</tbody>
</table>
| svc_type     | 1 = Document services  
2 = Cache services |
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`

The `print_reqopt` table 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`

The `print_requests` table keeps a record of each outstanding print request. Data I/O with this table is primarily accomplished through the use of the PRll 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.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>request_id</td>
<td>Unique search key - system-assigned request identification number.</td>
</tr>
<tr>
<td>done_time</td>
<td>Time at which the request was completed, cancelled, or failed.</td>
</tr>
<tr>
<td>request_time</td>
<td>When the print request was placed into the queue.</td>
</tr>
<tr>
<td>request_length</td>
<td>If printing Text=size of print job (in bytes); Image=# of pages.</td>
</tr>
<tr>
<td>print_err</td>
<td>Reason for print job failure. fn_msg command decodes hex number.</td>
</tr>
<tr>
<td>pages_printed</td>
<td>Number of pages successfully printed.</td>
</tr>
<tr>
<td>notify_addr</td>
<td>Address of server to notify when print job completes.</td>
</tr>
<tr>
<td></td>
<td>0 = do not notify</td>
</tr>
<tr>
<td>fax_request</td>
<td>NULL = print request</td>
</tr>
<tr>
<td></td>
<td>1 = fax request</td>
</tr>
<tr>
<td>security</td>
<td>Document or text file security information.</td>
</tr>
<tr>
<td>doc_id</td>
<td>Document id number of document being printed. Number 4160000001, and up indicates</td>
</tr>
<tr>
<td></td>
<td>text, not a document, is being printed.</td>
</tr>
<tr>
<td>first_page</td>
<td>Number of the first page that will be printed.</td>
</tr>
<tr>
<td>last_page</td>
<td>Number of the last page that will be printed.</td>
</tr>
<tr>
<td>ssn</td>
<td>System serial number if the svc_type column has a value of 2.</td>
</tr>
<tr>
<td>svc_type</td>
<td>The service that requested the print job.</td>
</tr>
<tr>
<td></td>
<td>1 = Document services</td>
</tr>
<tr>
<td></td>
<td>2 = Cache services</td>
</tr>
<tr>
<td>delete_after</td>
<td>0 = do not delete from cache</td>
</tr>
<tr>
<td></td>
<td>1 = delete after printing</td>
</tr>
<tr>
<td>svc_id</td>
<td>ID of service holding data to print</td>
</tr>
<tr>
<td>user_id</td>
<td>ID of user issuing print request</td>
</tr>
<tr>
<td>Field Name</td>
<td>Field Description, Continued</td>
</tr>
<tr>
<td>---------------</td>
<td>------------------------------------------------------------------------</td>
</tr>
<tr>
<td>terminal_id</td>
<td>ID of terminal print request issued from</td>
</tr>
<tr>
<td>print_order</td>
<td>Unique search key composed of the printer_num, inverse_prio, and</td>
</tr>
<tr>
<td></td>
<td>sequence_num keys - (group)</td>
</tr>
<tr>
<td>printer_num</td>
<td>Printer number assigned this print request</td>
</tr>
<tr>
<td>inverse_prio</td>
<td>Inverse priority (10-priority #) if job ready to print. Else, 10 if</td>
</tr>
<tr>
<td></td>
<td>print job is suspended or is scheduled for printing at later time.</td>
</tr>
<tr>
<td>sequence_num</td>
<td>Sequence number of print request. ex. 5010</td>
</tr>
<tr>
<td>request_stat</td>
<td>0= unknown 3 = completed 6 = waiting 1 = queued 4 = failed 7 =</td>
</tr>
<tr>
<td></td>
<td>fetching 2 = printing 5 = cancelled 8 = suspended</td>
</tr>
<tr>
<td>copies</td>
<td>Number of copies to print. NULL for default (1 copy)</td>
</tr>
<tr>
<td>notify_addr_len</td>
<td>Number of bytes in the notify_addr_str column.</td>
</tr>
<tr>
<td>notify_addr_str</td>
<td>Notification address for IPv6. Address of server to notify when print</td>
</tr>
<tr>
<td></td>
<td>job completes.</td>
</tr>
</tbody>
</table>
Requests tables and contents

bkg_request

The bkg_request 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.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>job_number</td>
<td>Job number.</td>
</tr>
</tbody>
</table>
| job_type        | 1 = Copy Documents 4 = Create Archive 7 = Migrate Documents  
|                 | 2 = Import Documents 5 = Find Open Docs 8 = Scan Surfaces  
|                 | 3 = Copy Annotations 6 = Import Archive 9 = Update Surfaces|
| prevjob         | Previous job in queue.                                 |
| jobqueuenum     | Job number to use for messages.                        |
| surfid          | ID of the current optical disk surface.                |
|                 | 0 = multiple surfaces are required.                    |
| suspended       | 0 = running 1 = suspended 2 = queued                   |
| num_long_wds    | Number of words in the array.                          |
| long_wd_ary     | Data relative to task type.                            |
| pid             | Process ID of background job.                          |
| start_time      | The time that the background job was started.          |

write_request

The write_request 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.

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

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>obj_class</td>
<td>The type of object: 1 = system, 3 = user, 2 = group, 4 = device</td>
</tr>
<tr>
<td>obj_name</td>
<td>ASCII string containing object name</td>
</tr>
<tr>
<td>obj_id</td>
<td>Unique object ID</td>
</tr>
<tr>
<td>admin_class</td>
<td>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 &quot;not administrator.&quot; If someone is &quot;not administrator&quot; then he or she is not any of the other administrator types. This field is not generally human-readable due to the way it is stored.</td>
</tr>
<tr>
<td>dev_class</td>
<td>The type of device: 1 = terminal, 2 = printer, 5 = fax, NULL = not a device</td>
</tr>
<tr>
<td>epasswd</td>
<td>Encrypted password</td>
</tr>
<tr>
<td>prim_group</td>
<td>Primary group ID</td>
</tr>
<tr>
<td>admin_group</td>
<td>Administrative group ID</td>
</tr>
<tr>
<td>language</td>
<td>Native user language</td>
</tr>
</tbody>
</table>
### 7 Security database tables and contents

#### Tables and contents

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.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description, Continued</th>
</tr>
</thead>
<tbody>
<tr>
<td>dev_security</td>
<td>Database creation date</td>
</tr>
<tr>
<td>creation time</td>
<td>Last successful login time</td>
</tr>
<tr>
<td>from_min</td>
<td>Last successful login location</td>
</tr>
<tr>
<td>to_min</td>
<td>Last failed login time</td>
</tr>
<tr>
<td>from_hour</td>
<td>Last failed login location</td>
</tr>
<tr>
<td>from_dweek</td>
<td>Log successful logons</td>
</tr>
<tr>
<td>to_hour</td>
<td>log all</td>
</tr>
<tr>
<td>to_dweek</td>
<td>calculated UNIX expiration time</td>
</tr>
<tr>
<td>exp_date</td>
<td>Total number of logons</td>
</tr>
<tr>
<td>succ_log_time</td>
<td>1 (bit value 1)</td>
</tr>
<tr>
<td>success_where</td>
<td>01 (bit value 2)</td>
</tr>
<tr>
<td>nbr_logons</td>
<td>log security changes 001 (bit value 3)</td>
</tr>
<tr>
<td>log_bits</td>
<td>log all 111</td>
</tr>
<tr>
<td>pwd_exp_time</td>
<td>calculated UNIX expiration time</td>
</tr>
<tr>
<td>pwd_fail_time</td>
<td>Time of password failure</td>
</tr>
</tbody>
</table>
### 7 Security database tables and contents

#### Tables and contents

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description, Continued</th>
</tr>
</thead>
<tbody>
<tr>
<td>pwd_fail_atts</td>
<td>Number of failed password attempts</td>
</tr>
<tr>
<td>sys_override</td>
<td>0 = use system, 1 = use users</td>
</tr>
<tr>
<td>max_session</td>
<td>Number of allowed nbr sessions</td>
</tr>
<tr>
<td>comments</td>
<td>General comments</td>
</tr>
<tr>
<td>sess_group</td>
<td>Session group</td>
</tr>
<tr>
<td>dn_len</td>
<td>The number of bytes in the dn column.</td>
</tr>
<tr>
<td>dn</td>
<td>Distinguished name</td>
</tr>
<tr>
<td>dn_hash</td>
<td>Hash key of the distinguished name.</td>
</tr>
<tr>
<td>concurrent_hwm</td>
<td>High-water mark for concurrent logons</td>
</tr>
</tbody>
</table>
| pwd_never_expire | Password never expire flag  
|                 | 0 = off  
|                 | 1 = on                                                      |
| hash_type      | 0 = NULL column  
|                 | 1 = SEC_HASH_TYPE_PED_SHA1  
|                 | 2 = SEC_HASH_TYPE_FIPS_SHA1                                  |
| hashpasswd_len | The number of bytes in the hashpasswd column.                |
| hashpasswd     | Hash key of the password.                                   |

**sec_system**

The **sec_system** table contains the system record for the Security service.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>system_id</td>
<td>Record which equals unique key</td>
</tr>
<tr>
<td>ceiling_id</td>
<td>Number to track assigned user ids</td>
</tr>
<tr>
<td>ceiling_snbr</td>
<td>Value of last session number assigned</td>
</tr>
</tbody>
</table>
### Security database tables and contents

#### Tables and contents

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description, Continued</th>
</tr>
</thead>
<tbody>
<tr>
<td>ceiling_fid</td>
<td>Value of next function number</td>
</tr>
<tr>
<td>ceiling_lan</td>
<td>Value of current language ID</td>
</tr>
<tr>
<td>dev_security</td>
<td>0 = off, 1 = on</td>
</tr>
<tr>
<td>no_func_def_ok</td>
<td>0 = not ok, 1 = ok</td>
</tr>
<tr>
<td>update_time</td>
<td>Stamp of time db last updated</td>
</tr>
<tr>
<td>from_min</td>
<td>Start time in minutes (0-59)</td>
</tr>
<tr>
<td>to_min</td>
<td>End time in minutes (0-59)</td>
</tr>
<tr>
<td>from_hour</td>
<td>Start time in hours (0-23: military)</td>
</tr>
<tr>
<td>to_hour</td>
<td>End time in hours (0-23: military)</td>
</tr>
<tr>
<td>from_dweek</td>
<td>Day of week (0-6: 0 = Sunday)</td>
</tr>
<tr>
<td>to_dweek</td>
<td>Day of week (0-6: 0 = Sunday)</td>
</tr>
<tr>
<td>max_sessions</td>
<td>The maximum number of current logons per user. This ensures that one user cannot use up all the SLUs for a given server.</td>
</tr>
<tr>
<td>highwater</td>
<td>Highest number ever logged on. This field is currently not used.</td>
</tr>
<tr>
<td>pwd_spec_char</td>
<td>Special character required (0 = no, 1 = yes)</td>
</tr>
<tr>
<td>pwd_min_len</td>
<td>Maximum password length (0 = minimum, 8 = maximum)</td>
</tr>
<tr>
<td>pwd_mwl_days</td>
<td>Password renewal from last change (0 = not set, 1 = minimum, 365 = maximum)</td>
</tr>
<tr>
<td>pwd_grace</td>
<td>Time (in days) until lack of password change causes account suspension (0 = not set, 1 = minimum, 90 = maximum)</td>
</tr>
<tr>
<td>pwd_attempts</td>
<td>Used to identify nbr of password attempts until account is disabled (0 = not set, 1 = minimum, 100 = maximum)</td>
</tr>
<tr>
<td>pwd_fail_mins</td>
<td>0 = not set, 1 = minimum, 100 = maximum</td>
</tr>
<tr>
<td>log_bits</td>
<td>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</td>
</tr>
</tbody>
</table>
The sec_deleted table is not currently used. It was designed to contain one record for each deleted object on the system.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>obj_id</td>
<td>Deleted object ID</td>
</tr>
<tr>
<td>obj_name</td>
<td>ASCII string containing object, domain, and organization</td>
</tr>
<tr>
<td>obj_class</td>
<td>Object class</td>
</tr>
<tr>
<td>dev_class</td>
<td>Deleted device class</td>
</tr>
<tr>
<td>del_time</td>
<td>Deletion time</td>
</tr>
</tbody>
</table>
sec_groups

The sec_groups table contains one record for each direct membership occurrence.

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

sec_functions

The sec_functions 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 some user-defined functions can have function security that restricts access to these functions.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fun_name</td>
<td>ASCII string containing object, domain, and organization.</td>
</tr>
<tr>
<td>fun_id</td>
<td>A unique numeric ID of the function name specified in the fun_name field.</td>
</tr>
</tbody>
</table>
sec_funcmbr

The sec_funcmbr 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.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fun_owner</td>
<td>This is a field grouping in MKF that indicates a membership. Contains the same value as the fun_member field. Does not display.</td>
</tr>
<tr>
<td>fun_id</td>
<td>Indicates the MKF ID of the function from the functions table. Contains the same value as the fun_id2 field.</td>
</tr>
<tr>
<td>member_id</td>
<td>Indicates the MKF object ID of the member of the function. Contains the same value as the member_id2 field.</td>
</tr>
<tr>
<td>fun_member</td>
<td>This is a field grouping in MKF that indicates a membership. Contains the same value as the fun_owner field. Does not display.</td>
</tr>
<tr>
<td>member_id2</td>
<td>Indicates the MKF object ID of the member of the function. Contains the same value as the member_id field.</td>
</tr>
<tr>
<td>fun_id2</td>
<td>Indicates the MKF ID of the function from the functions table. Contains the same value as the fun_id field.</td>
</tr>
</tbody>
</table>
| member_class | Indicates the object class of the member:  
0 = anyone  
1 = admin  
2 = group  
3 = user |

sec_namemap

The sec_namemap 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.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>filenet_map</td>
<td>Field grouping in MKF that indicates a mapping. Contains the same value as</td>
</tr>
<tr>
<td></td>
<td>the db_map field. Does not display.</td>
</tr>
<tr>
<td>obj_id</td>
<td>Indicates the object id of the FileNet user that is to be mapped to a</td>
</tr>
<tr>
<td></td>
<td>database user. Contains the same value as the obj_id2 field.</td>
</tr>
<tr>
<td>db_pseudo_id</td>
<td>An index of a row in the sec_dbinfo table indicating the database user that</td>
</tr>
<tr>
<td></td>
<td>this FileNet user is mapped to.</td>
</tr>
<tr>
<td>db_map</td>
<td>Field grouping in MKF that indicates a mapping. Contains the same value as</td>
</tr>
<tr>
<td></td>
<td>the filenet_map field. Does not display.</td>
</tr>
<tr>
<td>db_pseudo_id2</td>
<td>An index of a row in the sec_dbinfo table indicating the database user that</td>
</tr>
<tr>
<td></td>
<td>this FileNet Image Services user is mapped to.</td>
</tr>
<tr>
<td>obj_id2</td>
<td>Indicates the object id of the FileNet Image Services user that is to be</td>
</tr>
<tr>
<td></td>
<td>mapped to a database user. Contains the same value as the obj_id field.</td>
</tr>
</tbody>
</table>

**sec_dbinfo**

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

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>db_name</td>
<td>Native DB user name</td>
</tr>
<tr>
<td>db_pseudo_id</td>
<td>Native DB pseudo ID</td>
</tr>
<tr>
<td>db_pwd</td>
<td>Native DB encrypted password</td>
</tr>
<tr>
<td>db_creator_id</td>
<td>ID of creator</td>
</tr>
</tbody>
</table>
sec_rm_config

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

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key_id</td>
<td>A numeric identifier that defines the security level. The current support is for system-level security only.</td>
</tr>
<tr>
<td>declare_level</td>
<td>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)</td>
</tr>
<tr>
<td>rm_group_id</td>
<td>The object ID of the configured “Record Management” group</td>
</tr>
<tr>
<td>purge_level</td>
<td>Not used.</td>
</tr>
<tr>
<td>scrub_level</td>
<td>Not used.</td>
</tr>
<tr>
<td>log_level</td>
<td>A numeric identifier that defines the “log level” mode. Level 0 = Minimal (errors only) Level 1 = Verbose</td>
</tr>
<tr>
<td>lock_sysadming</td>
<td>Not used.</td>
</tr>
</tbody>
</table>

sec_map_prin_to_dn

The sec_map_prin_to_dn table holds mappings of FileNet Image Services objects to their LDAP distinguished names (dn) for exporting FileNet Image Services annotations to Content Engine in a CFS-IS environment.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id_to_dn</td>
<td>Unique search key composed of obj_ud and ce_dom_id keys - (group).</td>
</tr>
<tr>
<td>obj_id</td>
<td>Object id from the sec_object table</td>
</tr>
<tr>
<td>ce_dom_id</td>
<td>Links to the ce_dom_to_id table</td>
</tr>
</tbody>
</table>
sec_ce_dom_to_id

The `sec_ce_dom_to_id` internal table contains all unique Content Engine GUIDs.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dn_to_id</td>
<td>Unique search key composed of dn_hash and ce_dom_id2 keys - (group).</td>
</tr>
<tr>
<td>dn_hash</td>
<td>Distinguished name hash key</td>
</tr>
<tr>
<td>ce_dom_id2</td>
<td>Links to ce_dom_to_id table</td>
</tr>
<tr>
<td>dn_len</td>
<td>Distinguished name string length</td>
</tr>
<tr>
<td>dn</td>
<td>Actual distinguished name</td>
</tr>
</tbody>
</table>

sec_map_cn_to_prin

The `sec_map_cn_to_prin` table maps LDAP common names to FileNet Image Services Principals (Object Names) for Extensible User Authentication.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ce_dom_GUID</td>
<td>Unique GUID key for a Content Engine domain</td>
</tr>
<tr>
<td>ce_dom_id</td>
<td>Independent internal key number</td>
</tr>
<tr>
<td>obj_id</td>
<td>Object id from the sec_object table</td>
</tr>
<tr>
<td>ldap_cn</td>
<td>LDAP common name</td>
</tr>
<tr>
<td>prin_to_cn</td>
<td>Unique search key composed of obj_name and ldap_cn2 keys - (group).</td>
</tr>
<tr>
<td>obj_name</td>
<td>Object name from sec_object table</td>
</tr>
<tr>
<td>ldap_cn2</td>
<td>Same as ldap_cn (MKF requirement for multiple key setup). Does not display.</td>
</tr>
</tbody>
</table>
Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user’s responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Corporation
J74/G4
555 Bailey Avenue
San Jose, CA 95141
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:
Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan, Ltd.
1623-14, Shimotsuruma, Yamato-shi
Kanagawa 242-8502 Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.
Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
J46A/G4
555 Bailey Avenue
San Jose, CA 95141-1003
U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims.
related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.
Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at “Copyright and trademark information” at www.ibm.com/legal/copytrade.shtml.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Other company, product, and service names might be trademarks of IBM or other companies.

U.S. Patents Disclosure

This product incorporates technology covered by one or more of the following patents: U.S. Patent Numbers: 6,094,505; 5,768,416; 5,625,465; 5,369,508; 5,258,855.
Index

<table>
<thead>
<tr>
<th>Indexable Terms</th>
<th>Page Numbers</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td></td>
</tr>
<tr>
<td>application program</td>
<td></td>
</tr>
<tr>
<td>request handler</td>
<td>73</td>
</tr>
<tr>
<td>stub</td>
<td>73</td>
</tr>
<tr>
<td>application programs</td>
<td>73</td>
</tr>
<tr>
<td>Application server</td>
<td>23</td>
</tr>
<tr>
<td>C</td>
<td></td>
</tr>
<tr>
<td>Combined server</td>
<td>21</td>
</tr>
<tr>
<td>configuration database</td>
<td>43</td>
</tr>
<tr>
<td>configuration files</td>
<td></td>
</tr>
<tr>
<td>generated files</td>
<td>50</td>
</tr>
<tr>
<td>supplied</td>
<td>46</td>
</tr>
<tr>
<td>D</td>
<td></td>
</tr>
<tr>
<td>database tables</td>
<td></td>
</tr>
<tr>
<td>annot_log</td>
<td>103</td>
</tr>
<tr>
<td>annotations</td>
<td>104</td>
</tr>
<tr>
<td>batch_ctl</td>
<td>126</td>
</tr>
<tr>
<td>batch_data</td>
<td>139</td>
</tr>
<tr>
<td>batch_doc</td>
<td>127</td>
</tr>
<tr>
<td>batch_dyn_hdr</td>
<td>128</td>
</tr>
<tr>
<td>batch_folder</td>
<td>140</td>
</tr>
<tr>
<td>batch_hdr</td>
<td>132</td>
</tr>
<tr>
<td>batch_image</td>
<td>132</td>
</tr>
<tr>
<td>batch_ixdir</td>
<td>134</td>
</tr>
<tr>
<td>batch_ixval</td>
<td>135</td>
</tr>
<tr>
<td>batch_stat_hdr</td>
<td>137</td>
</tr>
<tr>
<td>bkg_request</td>
<td>154</td>
</tr>
<tr>
<td>cluster_map</td>
<td>105</td>
</tr>
<tr>
<td>csm_caches</td>
<td>141</td>
</tr>
<tr>
<td>csm_free_space</td>
<td>142</td>
</tr>
<tr>
<td>csm_temp_id</td>
<td>143</td>
</tr>
<tr>
<td>csm_used_space</td>
<td>144</td>
</tr>
<tr>
<td>dbinfo</td>
<td>164, 165, 166</td>
</tr>
<tr>
<td>docs</td>
<td>106</td>
</tr>
<tr>
<td>family_disk</td>
<td>110</td>
</tr>
<tr>
<td>family_locator</td>
<td>112</td>
</tr>
<tr>
<td>lib_surfaces</td>
<td>113</td>
</tr>
<tr>
<td>od_stats</td>
<td>114</td>
</tr>
<tr>
<td>print_docs</td>
<td>147</td>
</tr>
<tr>
<td>print_options</td>
<td>148</td>
</tr>
<tr>
<td>print_reqopt</td>
<td>151</td>
</tr>
<tr>
<td>print_requests</td>
<td>151</td>
</tr>
<tr>
<td>print_svcs</td>
<td>150</td>
</tr>
<tr>
<td>remote_family</td>
<td>115</td>
</tr>
<tr>
<td>scalar_numbers</td>
<td>116</td>
</tr>
<tr>
<td>sec_deleted</td>
<td>161</td>
</tr>
<tr>
<td>sec_funcmbr</td>
<td>163</td>
</tr>
<tr>
<td>sec_functions</td>
<td>162</td>
</tr>
<tr>
<td>sec_groups</td>
<td>162</td>
</tr>
<tr>
<td>sec_namemap</td>
<td>163</td>
</tr>
<tr>
<td>sec_object</td>
<td>157</td>
</tr>
<tr>
<td>sec_system</td>
<td>159</td>
</tr>
<tr>
<td>surf_dyn_info</td>
<td>116</td>
</tr>
<tr>
<td>surf_info</td>
<td>119</td>
</tr>
<tr>
<td>surf_locator</td>
<td>119</td>
</tr>
<tr>
<td>surf_stat_info</td>
<td>121, 123</td>
</tr>
<tr>
<td>write_request</td>
<td>154</td>
</tr>
<tr>
<td>databases</td>
<td></td>
</tr>
</tbody>
</table>
Index

Permanent database tables 102
Security database tables 156
Transient database tables 125
descriptions
  application programs 73
  Permanent database tables 102
  Security database tables 156
  shared libraries 88
  subsystems 35
  Transient database tables 125
directory contents 28
directory structure 27
Dual server configuration 22

I
IS directories
  contents 28
  structure 27

P
Permanent database tables
  annot_log 103
  annotations 104
  cluster_map 105
  docs 106
  family_disk 110
  family_locator 112
  lib_surfaces 113
  od_stats 114
  remote_family 115
  scalar_numbers 116
  surf_dyn_info 116
  surf_info 119
  surf_locator 119
  surf_stat_info 121, 123

R
  Remote Entry server 25
  request handler 73
  request handlers 61

S
  Security database tables
    sec_ce_dom_to_id 166
    sec_dbinfo 164
    sec_deleted 161
    sec_funcmbr 163
    sec_functions 162
    sec_groups 162
    sec_map_cn_to_prin 166
    sec_map_prin_to_dn 165
    sec_namemap 163
    sec_object 157
    sec_rm_config 163
    sec_system 159
server configurations 21
  Application server 23
  Combined server 21
  Dual server 22
  Entry server 25
shared libraries
  descriptions 88
  locations 87
stub 73
subsystem descriptions 35

T
  Transient database tables
    batch_ctl 126
    batch_data 139
    batch_doc 127
batch_dyn_hdr 128
batch_folder 140
batch_hdr 132
batch_image 132
batch_ixdir 134
batch_ixval 135
batch_stat_hdr 137
bkg_request 154
csm_caches 141
csm_free_space 142
csm_temp_id 143
csm_used_space 144
print_docs 147
print_options 148
print_reqopt 151
print_requests 151
print_svcs 150
write_request 154