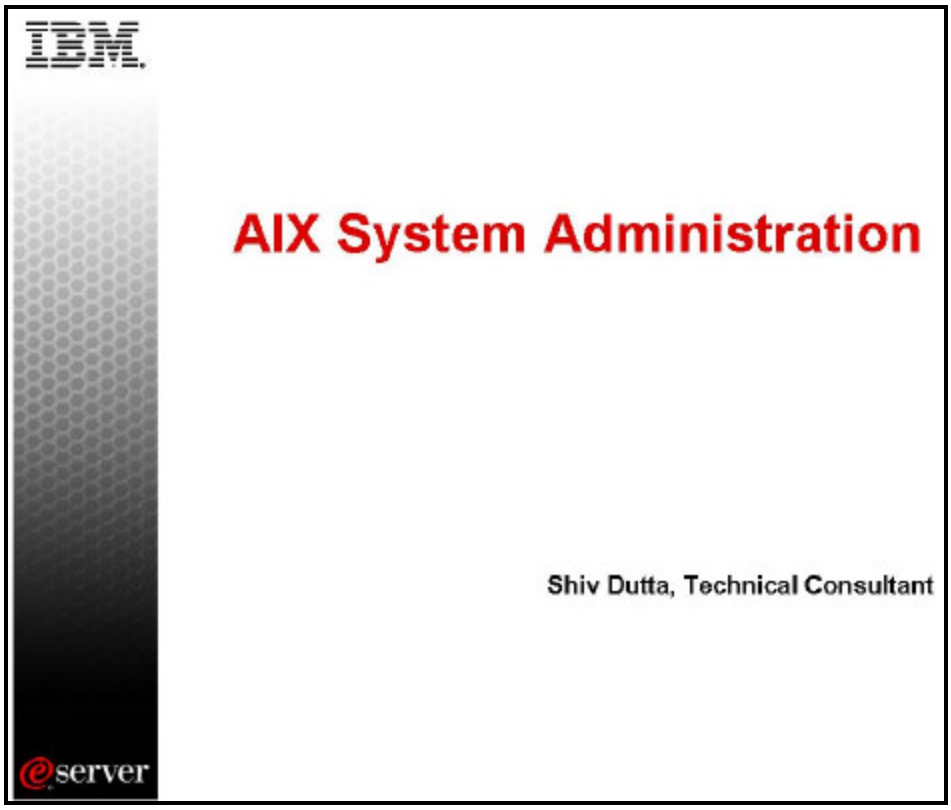
Contents

Author bio:
Shiv Dutta
Technical Consultant, IBM

Shiv Dutta is a technical consultant in IBM's Systems and Technology Group where he assists independent software vendors with the enablement of their applications on pSeries servers. Shiv has considerable experience as a software developer, system administrator, and an instructor. He provides AIX support in the areas of system administration, problem determination, performance tuning and sizing guides. Shiv has worked with AIX from its inception. He holds a Ph.D. in Physics from Ohio University and can be reached at sdutta@us.ibm.com.

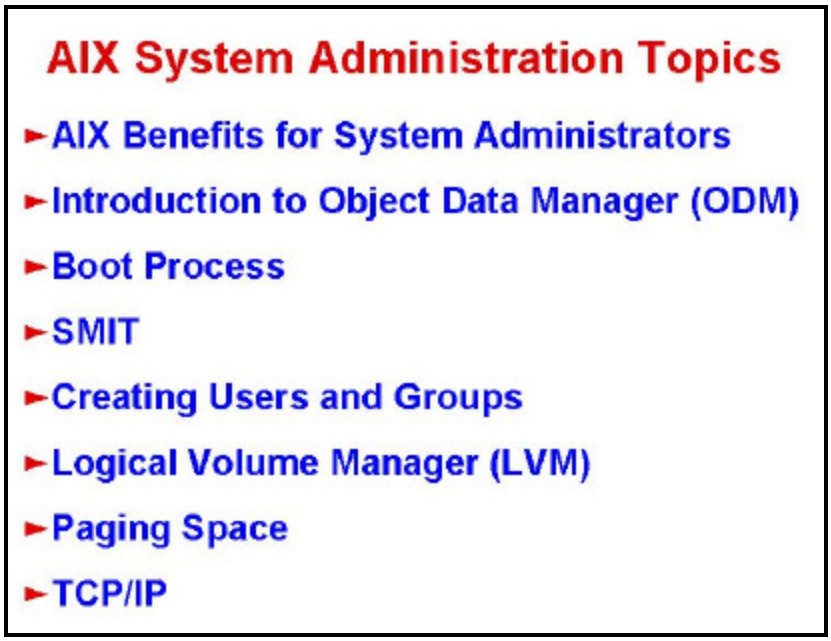


Introduction

AIX 5L™ is a highly popular platform for both UNIX® users and independent software vendors. This robust and easy-to-use operating system, which runs on the IBM® eServer™ pSeries™ and eServer p5 and eServer i5 family of platforms, delivers industrial strength UNIX reliability, availability, and security. It supports innovative virtualization and micro-partitioning features that became available with AIX 5L Version 5.3. And, AIX® offers flexible system administration tools and facilities—which is what you will be learning about today as you proceed through this less-than-one-hour-long course.

This online course is not an installation course, nor is its primary focus on performance management. However, elements of both of these topics will be addressed multiple times in the pages that follow. This material is truly ideal for those administrators who wish to gain a fundamental and broad-based understanding of the tools available on this powerful UNIX-based operating system.

Prerequisites: *There are no prerequisites for this course. It does not make reference to the POWER™ hardware on which AIX runs. It may be helpful, though, if the student has some familiarity with UNIX in general, though even this is not a requirement.*



AIX System Administration Topics

This course will begin by presenting an overview of the benefits offered to administrators who are able to run their UNIX applications under this state-of-the-art operating system.

There will be an introduction to Object Data Manager (ODM), which is an AIX exclusive offering. ODM is a database where system information is stored to facilitate the management of data and devices.

Various boot processes will be explained through examples. This includes initializing and managing boot logs, accessing and listing boot information, creating multiple boot images, and booting from different boot images that contain various AIX releases and maintenance levels.

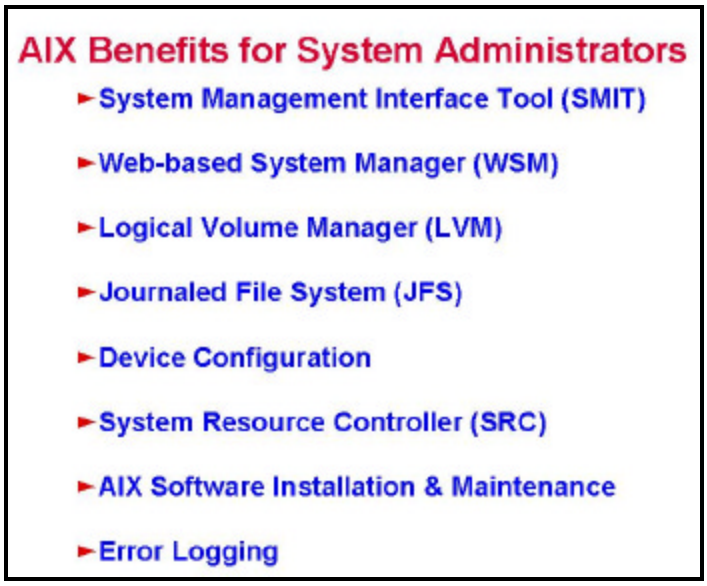
The System Management Interface Tool will be discussed—in all its implementations: the traditional ASCII-based smitty, an Xwindows version, and a Web-based System Manager (WSM) version. In fact, the smitty command will be used throughout this course to show examples of how certain commands and functions can be executed.

You will learn how to create users and user groups under the AIX purview.

There will be an explanation of physical and logical files, views, volumes, and partitions—with ample diagrams to help you understand how each of these terms, as well as others, interrelate with and build upon each other.

Paging spaces are a means of providing a secondary storage area for over-committed system memory. Though these spaces add flexibility, if not monitored and managed, they can also influence performance degradation. By the time you finish this section, you will have a clear understanding of how to use paging spaces to your advantage.

Installing a TCP/IP connection to provide the AIX system access to a network will be explained, as will a few additional miscellaneous points.



AIX Benefits for System Administrators

Many tools and benefits are offered by the AIX operating system for systems administrators to use for configuring and managing resources and clients. By way of introduction, let's mention these benefits briefly now.

A very powerful and unique tool, System Management Interface Tool (SMIT), is available for administrators to manage AIX servers. Written as an interactive interface application, SMIT displays a hierarchy of menus (i.e., a set of panels) that lead to interactive dialogues. SMIT has two interfaces: an ASCII (non-graphical) interface and an Xwindows interface. There is also a Web-based client interface, called Web-based System Manager (WSM), which only needs a browser to provide a graphical user interface for managing an AIX server.

Administrators also have access to the Logical Volume Manager (LVM). You will learn that a logical volume is associated with every virtual shared disk drive that is defined and configured on an AIX server.

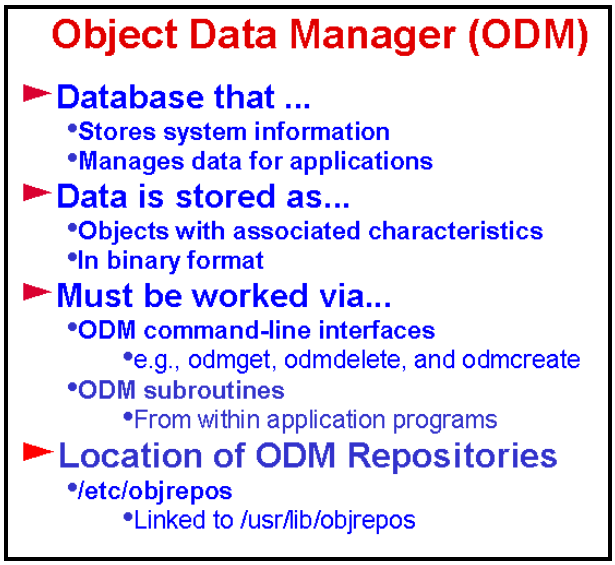
The Journaled File System (JFS) delivers a means of creating a fast restart database file system through the implementation of sophisticated database journaling.

The Device Configuration database is an object-oriented database designed to establish and maintain configuration details for resources attached to the server. This database is managed by the Object Data Manager (ODM).

The System Resource Controller (SRC) is a set of commands and subroutines used to create and manage interface system devices.

AIX software installation and maintenance is facilitated by one of the SMIT panels.

AIX provides an error facility that records entries in the system error log for any software or hardware failures that need to be available either for informational purposes or for fault detection and corrective action.



Object Data Manager (ODM)

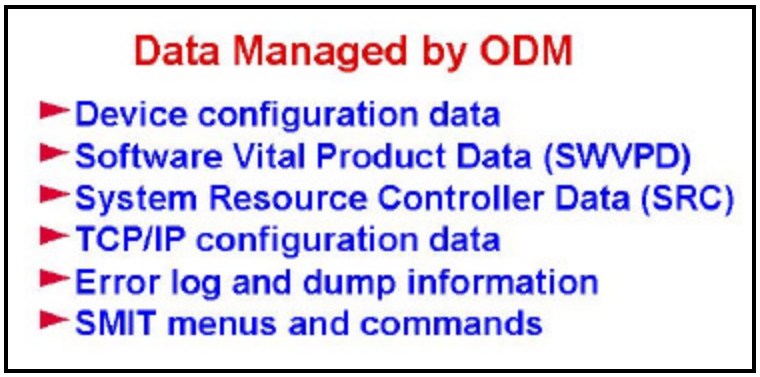
Object Data Manager (ODM) is unique to AIX. It is a database where system information is stored to manage devices. As mentioned, it can also store data for application programs. The information is represented (along with its set of characteristics) within objects and objects classes.

An **object class** is a group of objects with the same definition. An object class is similar to an array of structures, with each object being a structure that is an element of the array. Values are associated with the descriptors of an object when the object is added to an object class. An **object** is a member of an object class. It is similar to a logical record in a database.

ODM data is stored within an object in binary format. This means that its contents are not readable as would be the case with an ASCII or EBCIDC set of data. Because of its binary nature, therefore, it is necessary to use the ODM commands to act upon the data. You can create, add, lock, store, change, get, show, delete, and drop objects and object classes via ODM. ODM supports two methods for accessing these functions: (1) command-line interfaces and (2) ODM subroutines embedded in an application program.

Some of the many ODM commands include: odmget, odmdelete, and odmcreate. For example, if you wanted to look at the characteristics of a device attached to an AIX server, you would use the odmget command.

The ODM repositories are located in /etc/objrepos which is linked to /usr/lib/objrepos.

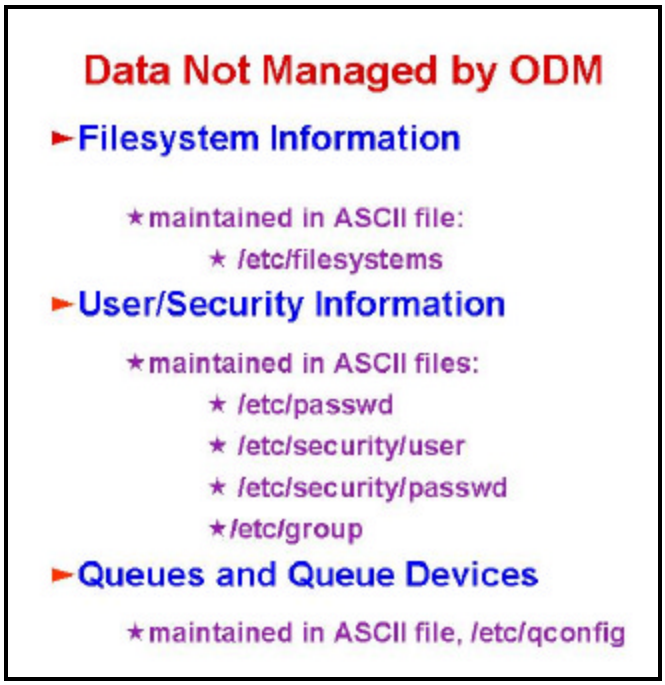


Data Managed by ODM

The types of system data that can be managed by ODM is related to:

- Device configuration information
- Software Vital Product Data (SWVPD) for installation and update procedures
- System Resource information (SRC data)
- Communications (TCP/IP) configuration information and setup information
- Error log and dump information, and
- Display information for SMIT (including: menus, selectors, and dialogs)

Many of these types of data will be explained in more detail as we move further into this course.



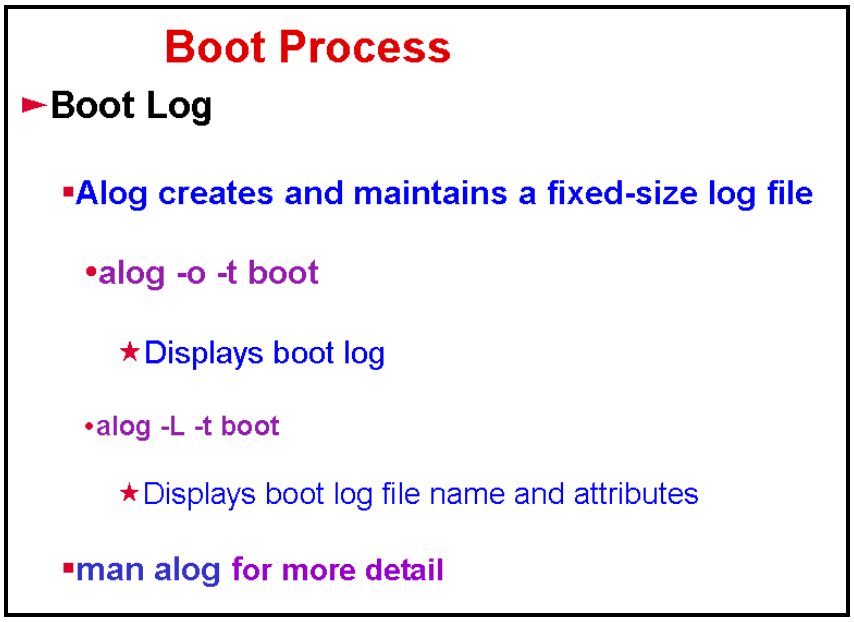
Object Data Manager (continued)

Certain types of AIX system and application information are not managed in ODM.

Filesystem information (the complete directory structure, including the root directory and any directories and files beneath it) is stored and maintained in an ASCII file and is located in /etc/filesystems. The filesystems data frees the administrator from having to enter and remember items such as the device where the filesystem resides. This data is managed via the mkfs, fsck, and mount commands, as well as other AIX commands.

User and security information is stored and maintained in four ASCII files (as shown on this chart). These user, password, and security sets of data are managed via the mkuser and pwdadm commands and other AIX commands.

Queue and queue device attributes are stored and maintained in ASCII files. This queue data is located in /etc/qconfig and is managed via the enq and qdaemon AIX commands.

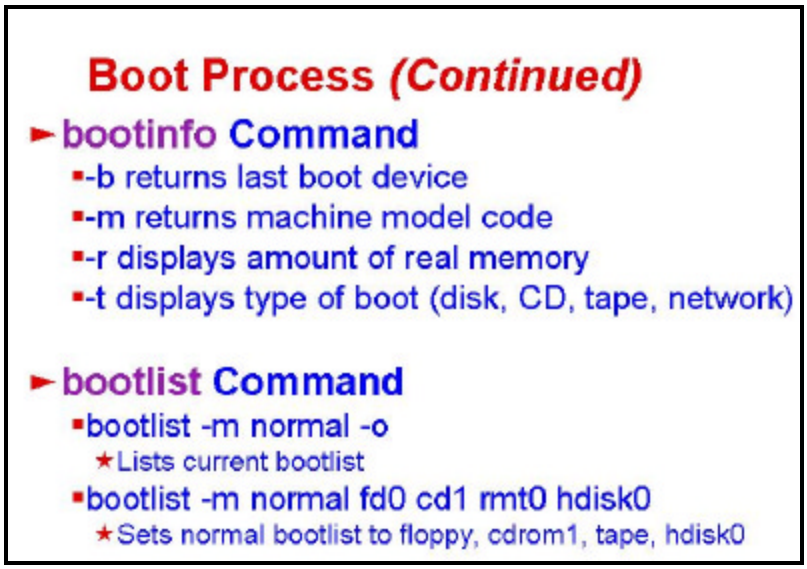


Boot Process

It is possible to maintain a log of the boot processes that occur during the boot of an AIX system. This is done via an alog command, which creates and maintains log files. If you wish to capture what has occurred during a boot process, enter **alog –o –t boot**. The –o flag causes the logfile contents to be listed. The –t flag establishes the logtype. The 'boot' parameter specifies the name of the log to be displayed. *[NOTE: the –o and –t flags must be entered in the sequence shown here.]*

The alog command can also be used with the -L flag (**alog –L –t boot**) to display the attributes defined for the boot log file. The attribute values that are listed as a result of entering this command include: File, Size, and Verbosity.

The **man** command provides an online display of reference information on various topics (e.g., commands, subroutines, and files). You can use the man command (as shown in the last bullet on this chart) to get an online detailed display of how you would enter the alog command.



Boot Process (continued)

The bootinfo command can be quite helpful to the administrator during boot. This is not a user-level command and is NOT supported in AIX 4.2 or later. However, it is available on the AIX operating system and is commonly used for quick viewing of critical boot information:

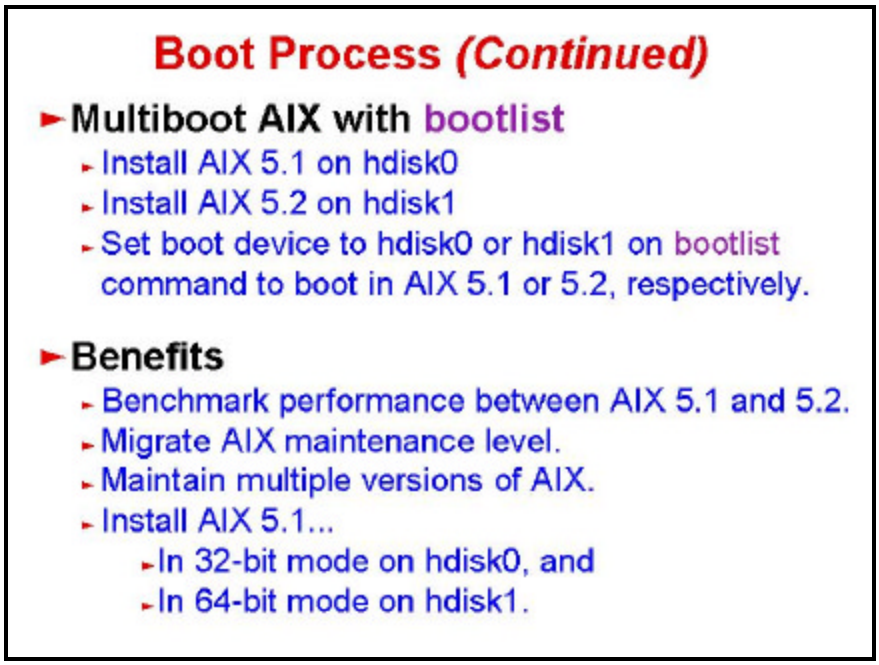
- The –b flag returns the last boot device.
- The –m flag lists the machine model code.
- The –r flag displays real memory in kilobytes.
- The –t flag provides the type of boot, whether it is from disk, CD, tape, or network.

[NOTE: As of AIX 5.2, the getconf command has been enhanced to provide access to the same information that was previously available via the unsupported bootinfo command. In particular, it will show real memory and type of boot.]

The bootlist command is a supported AIX command. It can be used by the administrator to display and alter the list of possible boot devices from which the system may be booted. When the system is booted, bootlist will scan the devices in the list and attempt to boot from the first device it finds containing a boot image.

The –m flag specifies whether the boot list to be returned is one of the following: normal, service, both, or prevboot. The –o flag indicated that the requested boot list is to be displayed after any specified alterations have been performed. The first bootlist command shown here (**bootlist –m normal –o**) will list the current boot list, while the second bootlist command shown here (**bootlist -m normal fd0 cd1 rmt0 hdisk0**) sets the normal boot list to attempt a boot by seeking a boot image sequentially from each of the following devices listed. Obviously, once a boot image is successfully encountered, no further searching of the devices is needed for this particular boot.

[NOTE: Be careful when specifying possible boot devices. A future reboot may fail if those devices become unbootable. Also, do not power off or reset the server while bootlist is executing, because the boot list may become corrupted or lost.]



Boot Process (continued)

The bootlist command provides the administrator with a great deal of flexibility. Let's imagine that you have decided to run multiple images of the AIX operating system on the same pSeries server. You can do this easily by installing AIX 5.1, for example, on hdisk0; and also putting a boot image of AIX 5.2 on hdisk1. Then, simply by setting the boot device sequence via the bootlist command (**bootlist –m normal hdisk0 hdisk1**), you can boot to these two release levels respectively. If you wish to operate under AIX 5.1, then you need do nothing else—since hdisk0 (where the AIX 5.1 image exists) is the first boot device listed in the bootlist command. If you need to operate under AIX 5.2, simply vary off the hdisk0 device before booting the server. Then the bootlist will seek a boot image from hdisk1. After the boot has completed, you can vary hdisk0 back online.

There are many reasons you may want to do this. It is an easy way to conduct benchmarks between release levels. It also facilitates migration to a newer AIX maintenance release, while ensuring that you have a fallback position should that maintenance upgrade cause any problems with your application code. You may even have applications that run only in 32-bit mode. If this is the case, you can maintain a 32-bit version of AIX for that purpose.

Developers love this flexibility!

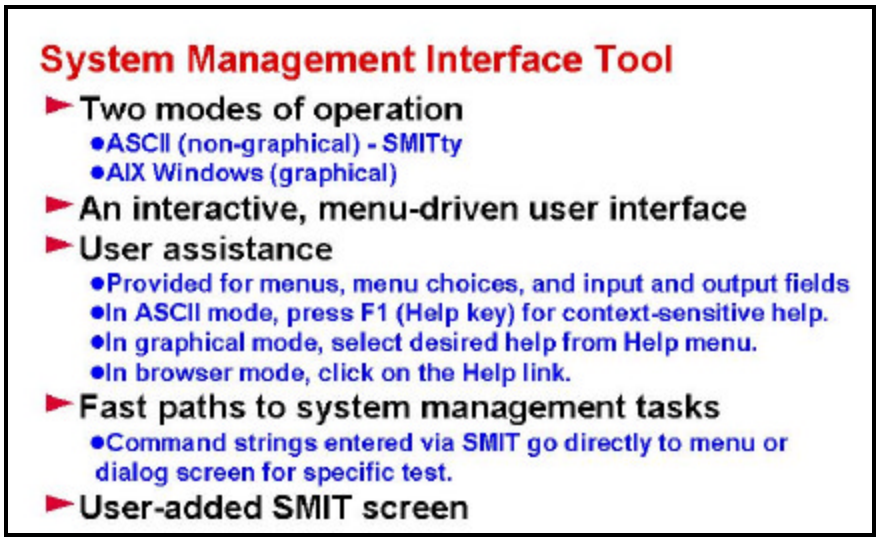
Boot Process (continued)

The bosboot command creates a complete boot image that is written to the default boot logical volume on the fixed disk from which the server is booted. This boot image interfaces with the machine boot Read-Only Storage (ROS) EPROM (Erasable Programmable Read-Only Memory).

In essence, the bosboot command creates a boot file from a RAM disk file system and a kernel. This boot file contains device configuration routines that make the machine's preconfigured devices and file system available. The devices on which a boot image can be created includes: CD-ROM, disk, tape, and network. The boot image does change for each type of device from which it is booted. This means that a boot image written to CD-ROM cannot merely be copied to tape. A separate boot image must be created for each device that will appear in the boot list.

The only action flag is the –a flag which causes the bosboot command to create a complete boot image. The –d device flag is used to specify the boot device for which this boot image is being created. This flag is not required if the boot device is from hard disk. Thus, the **bosboot –a** example shown here will create a complete boot image that will be stored on the hard disk.

Some optional flags are further explained in the Commands Reference (found online at the pSeries Information Center... a link to this site is provided at the end of this course).



System Management Interface Tool

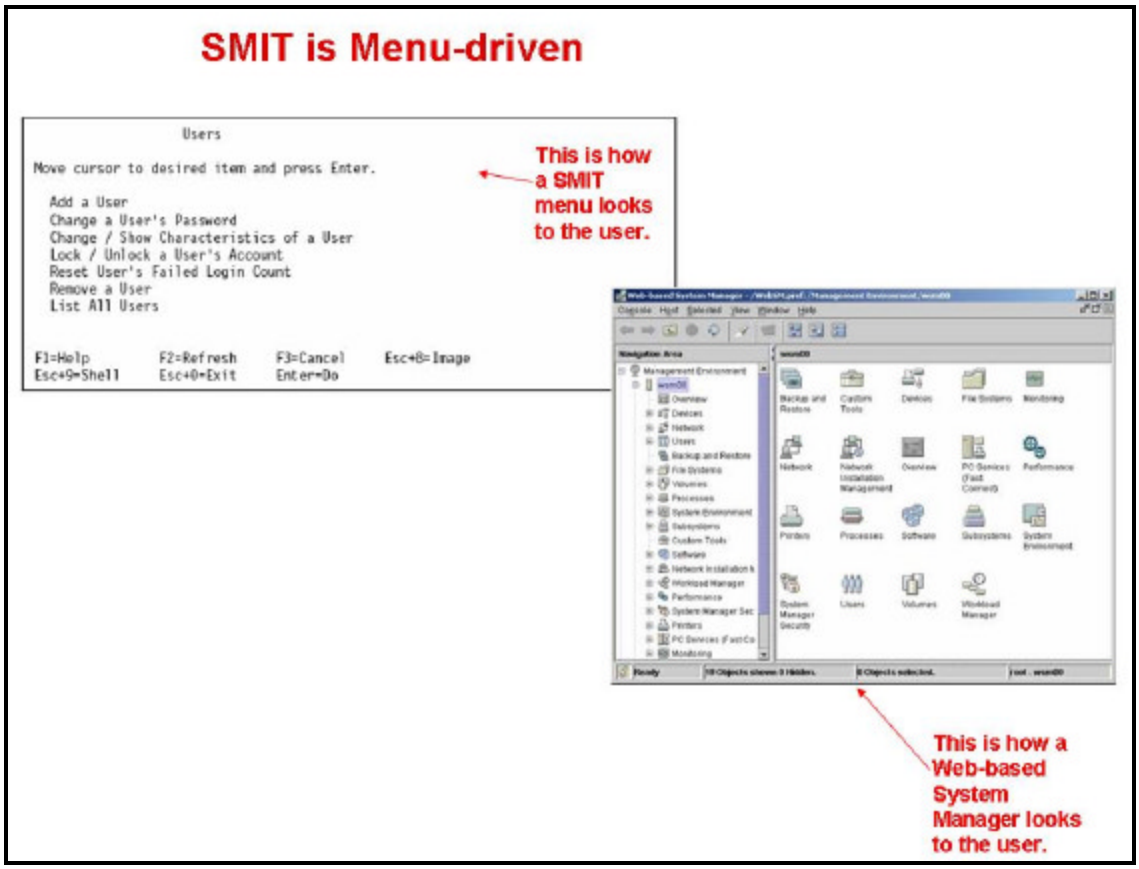
This tool offers an easy, powerful alternative to the typical method of entering complex command syntax, valid parameter values, and custom shell path names to manage and maintain an AIX configuration. SMIT is an easy interface not available on other UNIX implementations. It has two interactive, menu-driven operational modes to provide assistance when using menus, menu options, and input and output fields

- A non-graphical (ASCII) interface displays a hierarchy of menus (i.e., a set of panels) that lead to interactive dialogues between AIX and the administrator. Assistance is accessed by pressing the F1 Help key to gain context-sensitive help.
- An Xwindows-based client version of SMIT provides a graphical user interface. Assistance is available by clicking on the Help menu.

As mentioned, there is also a Web-based client interface called Web-based System Manager (WSM). WSM needs only a browser to provide a graphical user interface for managing an AIX server.

You can store fast paths to system management tasks via the smitty command. When you hit a pre-stored fast path, command strings are entered via SMIT and go directly to the appropriate menu or dialog screen for a specific task—bypassing preliminary dialog and menu screens.

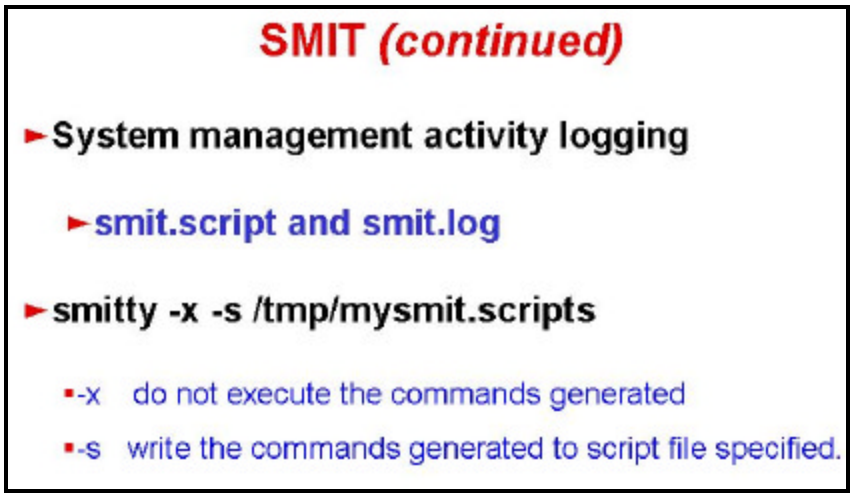
It is also possible to add SMIT screens to the SMIT database that provide interactive help for end-user applications. This is done via ODM and requires quite a bit of familiarity with ODM. As you might expect, this is very popular with developers who need to make their applications easier to use and quicker to learn.



SMIT is Menu-driven

As mentioned, a menu is the basic entry point into SMIT and can be followed by another menu, selector, or dialog screen. Menus present a list of tasks, which when selected, lead to another menu, selector, or dialog box.

You can see examples of how the menus will appear, depending on whether you use SMIT interface or Web-based System Manager.



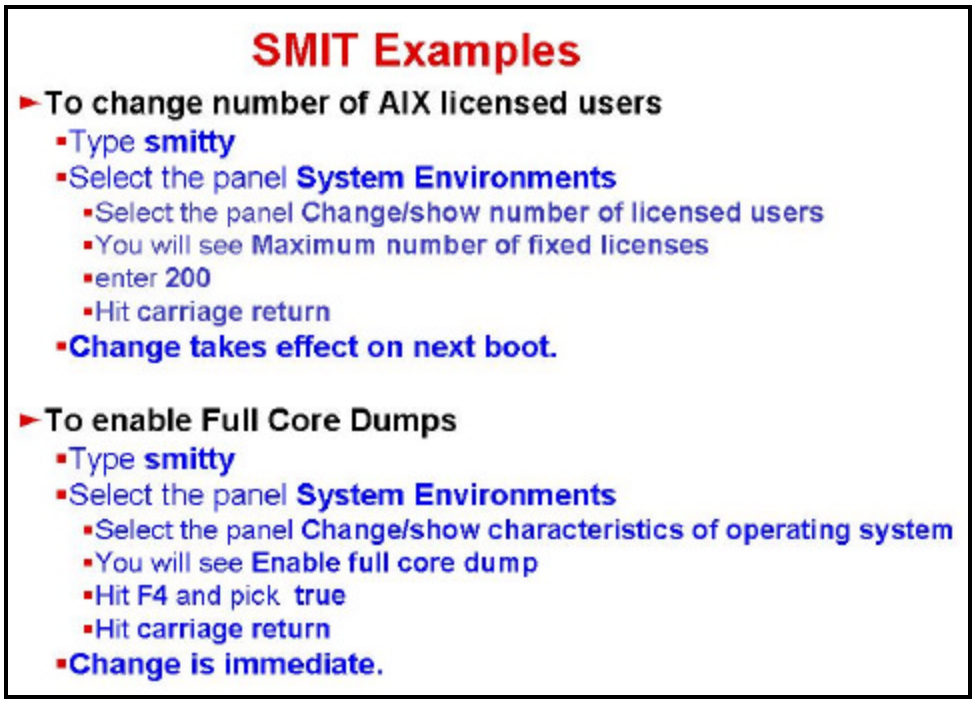
SMIT (continued)

SMIT logs all system management activity in two files, which are located in the home directory of the user. The smit.log file records SMIT actions, such as the name of each screen displayed, the command string it ran, the output from the command string, and any error output. The smit.script file records high-level command strings that are executed. Entries in both files are also date stamped.

[NOTE: These files grow very large when not managed. However, the administrator can set up a task via the cron command to specify that these two files are cleaned up at administrator-specified time intervals.]

It is possible to specify an alternative name and an alternative location for the smit.log and smit.script files to be stored. This would be done via the fast path smitty command shown here where the –x flag specifies that the commands generated should not be executed. Instead, the –s flag indicates that the commands generated should be written to a script file that, in this example, is named /tmp/mysmit.scripts. You might want to do this so that you can repeat the same script in the future.

[NOTE: The smit command and the smitty command perform identical functions. Thus, in this course, every time we talk about the smitty command, we are also referring to the smit command. The smitty command gives an Xwindows-based interface, whereas smit gives a text-based interface.]

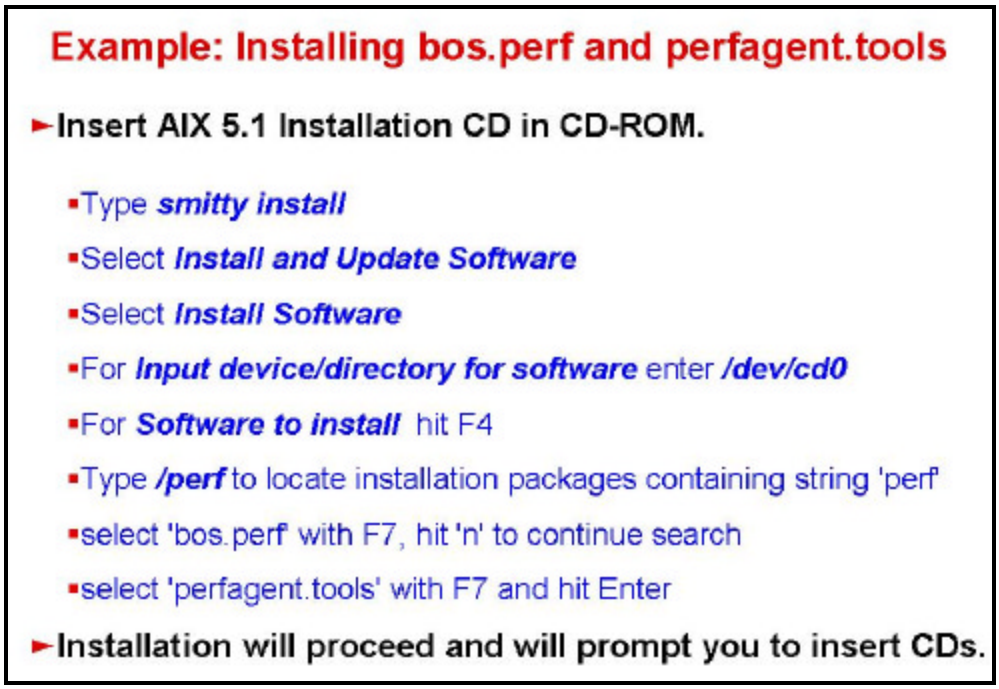


SMIT Examples

The smitty command can do many things.

It can be used to change the number of AIX licensed users on the server by typing **smitty** at the command line. Then, one of the choices in the first SMIT panel is **System Environments**—which you can select. The panel you are next shown will have a choice to **Change/show number of licensed users**. When you click on this, you will be allowed to enter a new maximum number of fixed licenses. After hitting **carriage return**, whatever new value you entered will become effective at the next system boot.

Similarly, to enable a full core dump, type smitty at the command line. Then, on the first SMIT panel, choose **System Environment**. This will take you to another panel with a choice to **Change/show characteristics of operating system**. If you choose this, the next option you will be shown is **Enable full core dump**. By hitting **F4** and then picking **true**, and lastly hitting the **carriage return**, full core dumps will be enabled immediately.



SMIT Example: Installing bos.perf and perfagent.tools

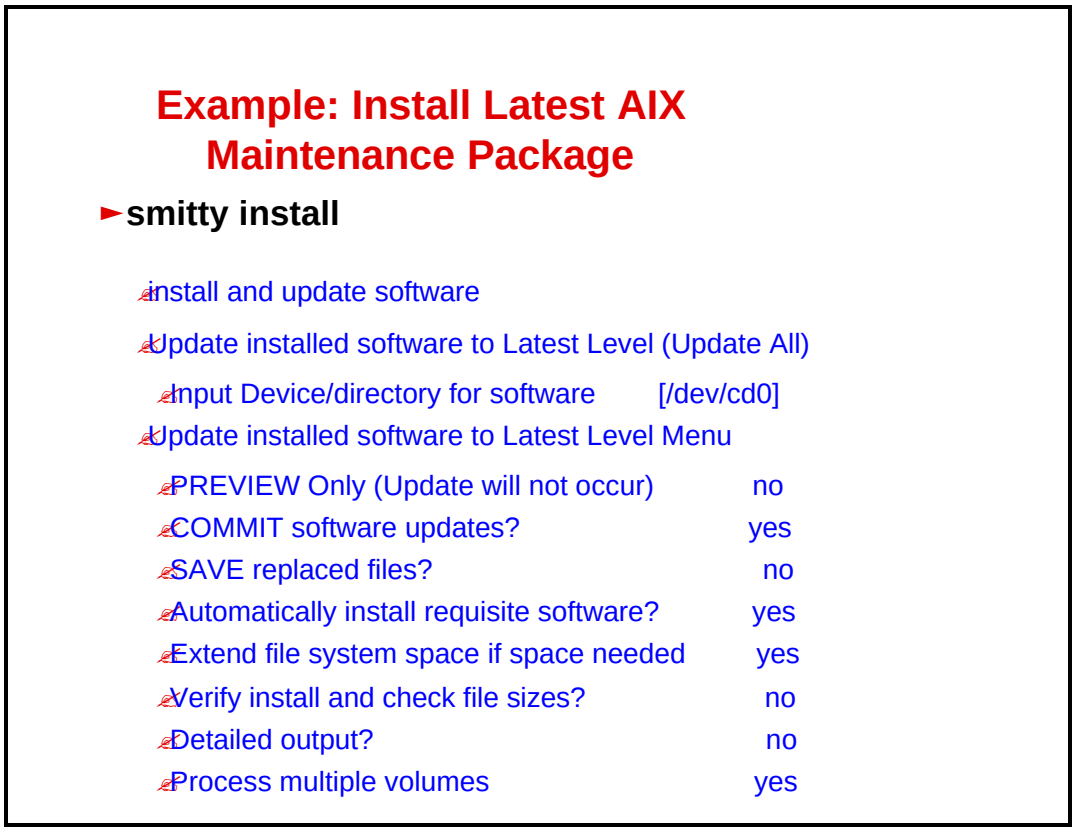
In this example of using the smitty command, the administrator is already managing an operational AIX system, but has determined the need for a couple of additional software files—**bos.perf** and **perfagent.tools**. These files are used for performance management and are not normally installed by default.

Since you do not know which installation CD contains the two needed performance files, you would put the first AIX installation CD in the CD-ROM and start the smit tool. You would next select **Software Installation and Maintenance**, and then pick **Install and Update Software**.

Or, you could use the fast path **smitty install**, which will take you directly to the **Install and Update Software** panel where you can choose **Install Software**. Then, you will see a prompt for **Input device/directory for software**. Though you will see more than one choice (hard drive, tape, CD), you will enter **/dev/cd0** since that is where you have loaded the first AIX installation CD.

If you hit F4, you will be shown a list of **Software to Install**. The F4 key allows you to pick and choose only the specific software you wish to install (from among the numerous software available on this CD). You can type in **/perf** so that smitty will locate all packages that contain the string 'perf.' From this returned list of packages, you can select **bos.perf** with the **F7** key, and then hit '**n**' to continue the search since you will also need to select **perfagent.tools** (by hitting the **F7** key). It will ask you to insert another installation CD until it has successfully located the two files you have requested.

After you have done all of this, smitty will proceed with the installation of these two files.

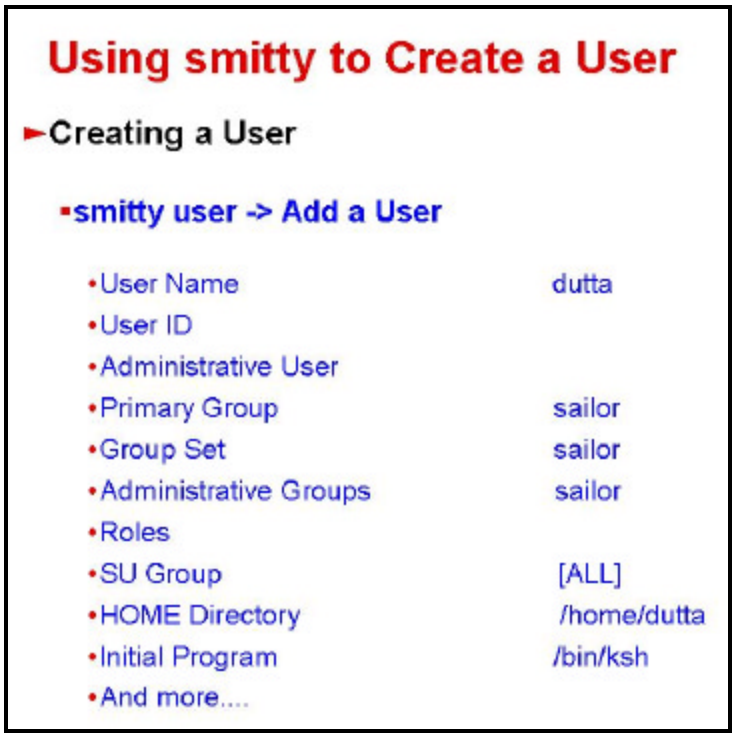


Example: Install Latest AIX Maintenance Package

The smitty command can also be used to install the latest maintenance level of AIX. By using the fast path **smitty install**, you will be taken immediately to the **Install and Update Software** panel. Next, you will select **Update installed software to latest level (Update All)**. Again, for the input device, you will select **/dev/cd0**. Then, you will see several options, as shown at the bottom of this chart:

- **Preview Only** means that the updates will not actually occur. Usually, the answer to this is 'no,' but this is a method for you to determine if these are the update files you really want to install.
- **COMMIT software updates?** means that you want to replace the old with the new... there is no going back. An alternative choice would be **apply**, which would mean that you want to use the updates, but you want to be able to reverse them if they do not work properly.
- **Save replaced files?** refers to whether or not you want to save the replaced files. If you have selected Commit, you would say 'no' to 'Save.' But if you chose 'Apply,' then you would say yes to 'save.'
- **Automatically install requisite software?** is usually answered by 'yes.'
- **Extend system space if space needed?** is usually answered by 'yes.'
- **Verify install and check file sizes?** is usually answered by 'no.'
- **Detailed output?** is usually answered by 'no.'
- **Process multiple volumes** is usually answered by 'no.'

In other words, the only options you will generally need to be concerned with are Preview, Commit, and Save.



Using smitty to Create a User

The **smitty user** fast path can be used to add a user. You will be prompted for the user name (for example, the user's first initial and last name). The user ID will be assigned by the server. The user will generally be assigned as an Administrative User. The user's Primary Group will usually be automatically selected (e.g., 'staff') by the server—unless you specifically want to assign this user to a separate group altogether. Users can belong to more than one group.

The Home Directory will be important because it must reflect the directory mandates and standards set up on your AIX system. By default, the Initial Program invoked for all users is /bin/ksh. This is the korn shell. If you need to specify another program instead, you would do so with this attribute.

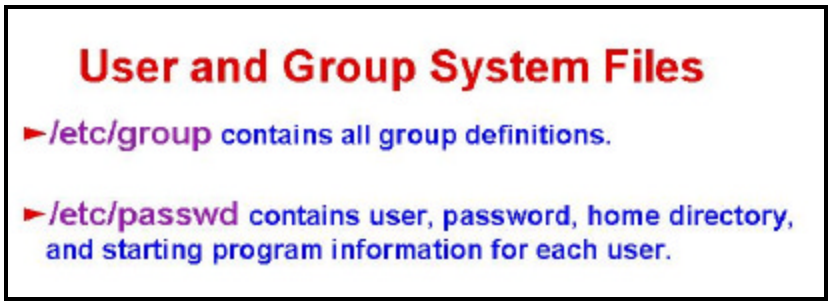
There are other user attributes that can also be assigned, but these are the ones most commonly used.

Using smitty to Create Groups

You can use smitty to create a user group. All users on AIX belong to the default user group 'staff.' However, the administrator may wish to assign other groups from time to time, for example, when there is a special project that only certain people are working on. In this case, it may be ideal to grant this group of people special access to certain objects, databases, applications, etc. This can be done using the **smitty group** fast path. This will take you directly to the **Add a Group** panel. From here, you can specify a group name, group ID, or a user list of users who belong to this group. If you do not want to include a list of users here, you can assign users individually to this new group by using the smitty user fast path mentioned previously.

Using smitty to Set a User Password

The fast path **smitty passwd** allows you to set user passwords. You must provide the user name, then select to change the password for that user, after which, you will be allowed to enter that user's new password.



User and Group System Files

The ***/etc/group*** *file* stores group definitions. Similarly, the ***/etc/passwd*** file stores user-related information (e.g., user name, password*, home directory, and starting program information).

*Actually, the /etc/passwd file has a blank character instead of the actual password—which is stored in a separate file for security reasons.

Logical Volume Manager Definitions

- **Physical volume (PV)**
  - **Hard disk**
- **Volume group (VG)**
  - **Collection of PVs that share one physical partition size**
- **Physical partition (PP)**
  - **Fixed size, contiguous set of bytes on a PV.**
  - **Physical partitions (PP) must be the same size across an entire VG.**
  - **A PP can be 1-1024 MB in powers of 2. In AIX, PPs default to 4MB.**
- **Logical partition (LP)**
  - **Mapped 1-to-1 to physical partitions unless mirroring**
- **Logical volume (LV)**
  - **Must be contained in one VG, can span multiple PVs.**
  - **Can be used for a file system, paging space, or raw logical volume.**

Logical Volume Manager Definitions

Let's review some fundamental definitions that relate to logical volumes in the context of the AIX operating system.

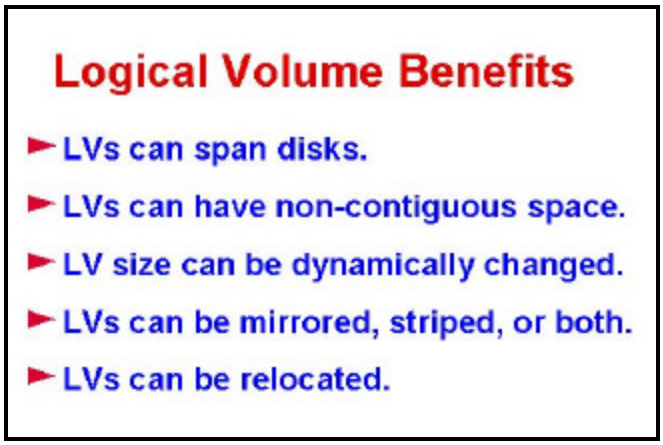
A **physical volume** (**PV**) refers to hard disk.

A **volume group** (**VG**) is a collection of physical volumes (e.g., hdisk0, hdisk1, and hdisk3) that share one physical partition size. A volume group can also contain as little as one physical volume.

A **physical partition** (**PP**) has a fixed size with a contiguous set of bytes on a physical volume—visualize them as slices of a physical volume. Physical partitions must be the same size across an entire volume group. A physical partition can be between one and 1,024 megabytes in size (in incremental powers of two). In AIX, physical partitions default to four megabytes.

A **logical partition** (**LP**) is mapped one-to-one to physical partitions unless mirroring is being done. A logical partition is not a physical entity... it is a concept. A physical partition is, of course, a physical entity. The only time there is not a one-to-one correlation between LPs and PPs is when there is mirroring—in which case, two copies of the physical file are created (for high availability reasons). That is, if mirroring is specified, normally, there are two physical partitions for each logical partition (a two-to-one ratio).

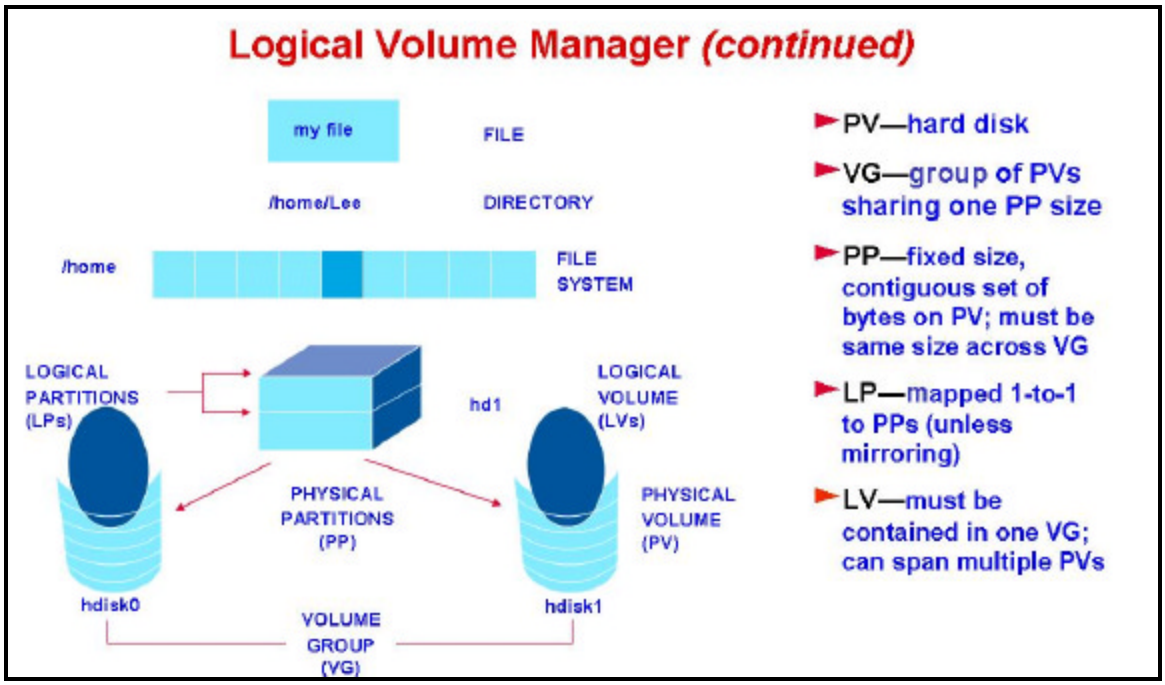
A **logical volume** (**LV**) consists of logical partitions. Logical volumes must be contained in one volume group and can span multiple physical volumes. A logical volume can be used for a file system, a paging space, or a raw logical volume.



Logical Volume Benefits

Many benefits can be realized from using Logical Volumes, some of which we have briefly mentioned. But, let's review them all here together.

Logical volumes can span multiple disks. They can include non-contiguous space. The size of LVs can be changed dynamically—as long as there is space available. Logical volumes can be mirrored. And/or they can be striped, which means the data is intentionally written across a group of disks. This might be done to increase read/write speed. And, finally, logical volumes can be backed up and relocated.



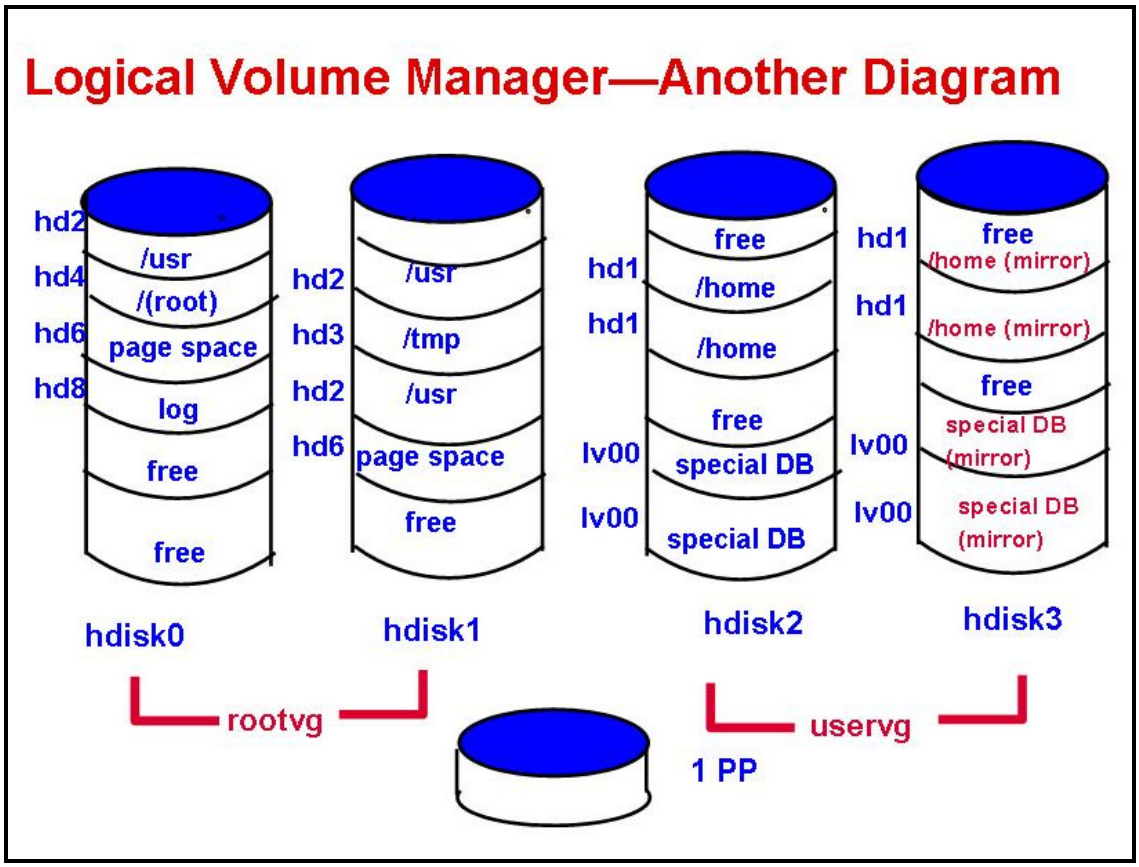
Logical Volume Manager Diagram

This is a visual representation of the logical volume concepts we have just discussed. We have included a synopsized set of definitions on the right for easy reference.

At the bottom of this chart, there is a volume group that consists of two hard disks (hdisk0 and hdisk1). If you look carefully, you can see white lines on each of the disk stacks. They indicate physical partitions. The logical partitions (hd1) are represented by the rectangular box in the center and are comprised of the physical partitions on hdisk0 and hdisk1.

Whenever you create a file system, they are installed on logical volumes. So, a file system is a layer on top of a logical volume. In this example, the file system is called **/home**. On top of the file system, there is a directory (called **/home/Lee**) which contains a file named “my file.”

The Logical Volume Manager is the key to all of these physical and logical elements fitting together.



Logical Volume Manager—Another Diagram

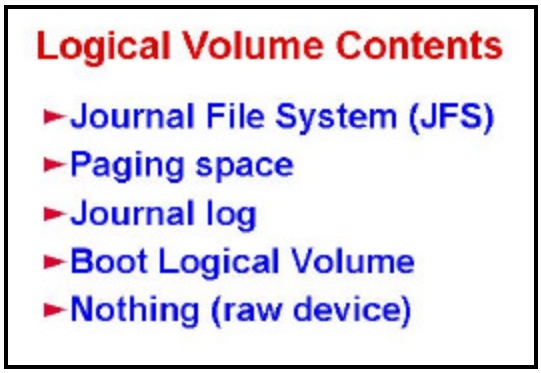
This drawing offers a different way to look at logical volumes. There are two volume groups: **rootvg** is made up of hdisk0 and hdisk1; **uservg** consists of hdisk2 and hdisk3.

rootvg has logical volumes, logs, page space, root, and usr—all residing on hdisk0.

hdisk1 has tmp, usr, and page space.

You will notice that there are usr and page space LVs on both hard disks. This is because, within a volume group, the logical partitions can span more than one physical disk. *[NOTE: Free spaces are not being used.]*

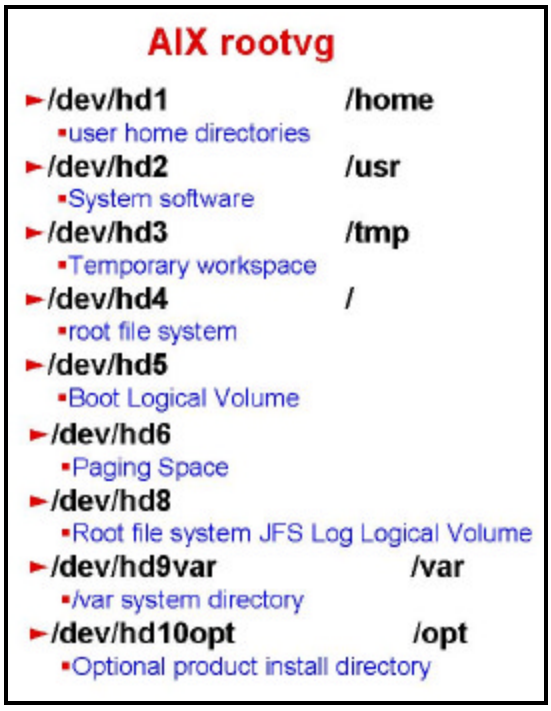
If you look at the uservg volume group, you will notice that it is similarly comprised.



Logical Volume Contents

A logical volume can contain a file system. In the case of AIX, we call it the Journal File System (JFS). An LV can also contain one or more paging spaces. It can contain journaled logs of the file system, and it can contain a boot logical volume.

Or, a logical volume can also simply contain nothing—existing only as a raw device. Raw devices are quite often used by the database system. The reason for this is that when using the Journal File System, some amount of overhead is required, and thus affects performance. If the database (DB2 and other databases) can use the raw device, instead, to perform temporary and intermediate calculations on data that will not be saved permanently, then the JFS is not accessed, and journaling (and its associated overhead) is reduced.



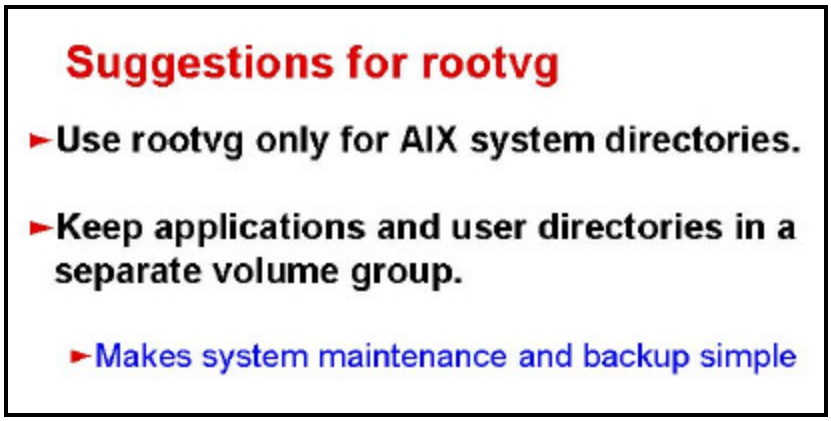
AIX rootvg

This chart lists the logical volumes that are automatically created when installing AIX. By default during installation, a volume group is created, called **rootvg**. (The administrator does not need to do anything to ensure that this volume group is created.) Within this volume group, by default, there will also be a number of logical volumes that are created. For instance, all the logical volumes shown here (**/dev/hd2**, **/dev/hd3**, **/dev/hd4**, etc.) are automatically created and put within the rootvg volume group.

These logical volumes are mounted, again automatically and by default, to different file systems. For example, **/dev/hd1** is mounted on a file system called **/home**. Similarly, **/dev/hd2** is mounted on the **/usr** file system. **/dev/hd3** is mounted on the **/tmp** file system... and so forth.

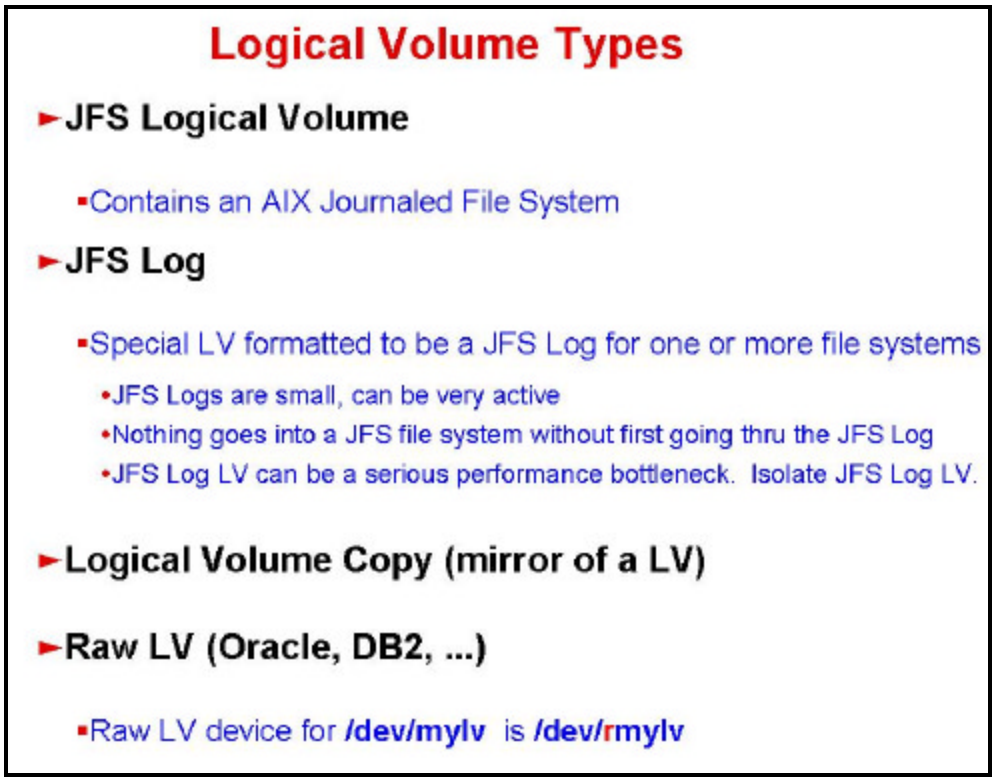
- **/dev/hd1** will contain user home directories. Its name is rather self-explanatory. The /home file system is where you would put user directories for your AIX users.
- **/dev/hd2** is intended to contain system software. So, additional installations of AIX system software will be automatically installed in the /usr file system.
- **/dev/hd3** provides file system resources for temporary and intermediate workspace.
- **/dev/hd4** is the root file system. It will end up containing a lot of system files, too.
- **/dev/hd5** is the place where the Boot Logical Volume is located.
- **/dev/hd6** is where the paging space is created.
- **/dev/hd8** is where the root file system JFS log logical volume will be created.
- **/dev/hd9var** will contain the var system directory.
- **/dev/dh10opt** is where optional product installations will put their software files. Optional products include freeware, Linux, etc.

[NOTE: Each logical volume is created with a default size, though these sizes can be changed after installation to better suit your organization's needs.]



Suggestions for rootvg

As you can see, the rootvg volume group has a large number of system-related directories. We highly advise you to keep your applications and user directories in a separate volume group. Use rootvg just as it is installed and intended. This will make system maintenance and backup much simpler.



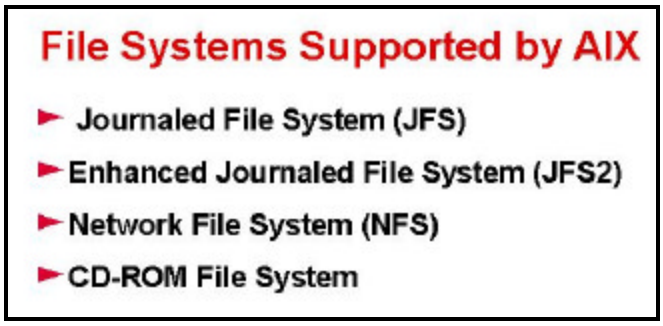
Logical Volume Types

The JFS logical volume contains an AIX journaled file system.

The JFS log is a special logical volume that is formatted to act as a JFS log for one or more file systems. These logs are important when there has been a system crash because they can help reconstruct the data. JFS logs are small, but can be very active. This is because, as mentioned, nothing goes into a JFS file system without first going through the JFS log. As a result, the JFS log can become a serious performance bottleneck. The best way to alleviate this performance issue is to isolate the JFS log logical volume.

A Logical Volume Copy is a mirror of a logical volume. As explained, mirroring creates two copies of the physical file.

Also as already discussed, a raw logical volume device contains no permanent information. It is often used for temporary and intermediate calculations by the database—avoiding the overhead that would be created by the JFS if these same calculations were performed there. The naming convention is shown here at the bottom of the chart.



File Systems Supported by AIX

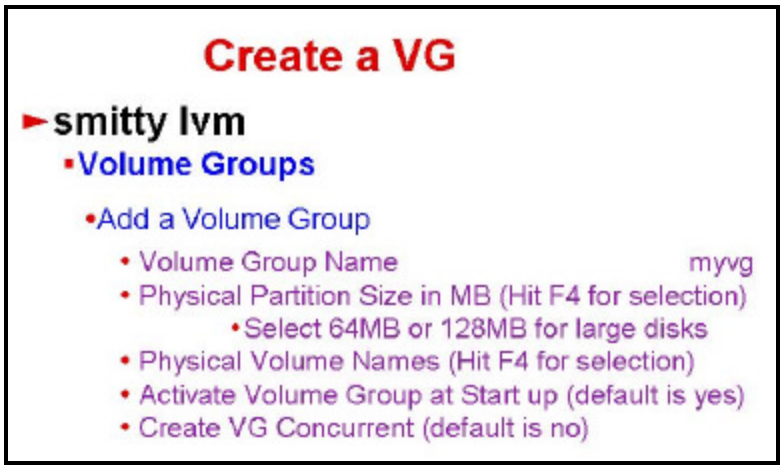
There are four file systems that are supported by AIX.

The **Journaled File System** (JFS) uses database journaling techniques to maintain its structural consistency.

In addition to the journaling techniques, the **Enhanced Journaled File System** (JFS2) supports multiple file system block sizes and larger file sizes. JFS2 is only available with AIX versions 4.2 and later.

The **Network File System** (NFS) allows users to access files and directories located on remote computers and to use the files and directories as if they were local.

The **CD-ROM File System** allows access to the contents of a CD-ROM through the normal file system interfaces. It is a read-only local file system.

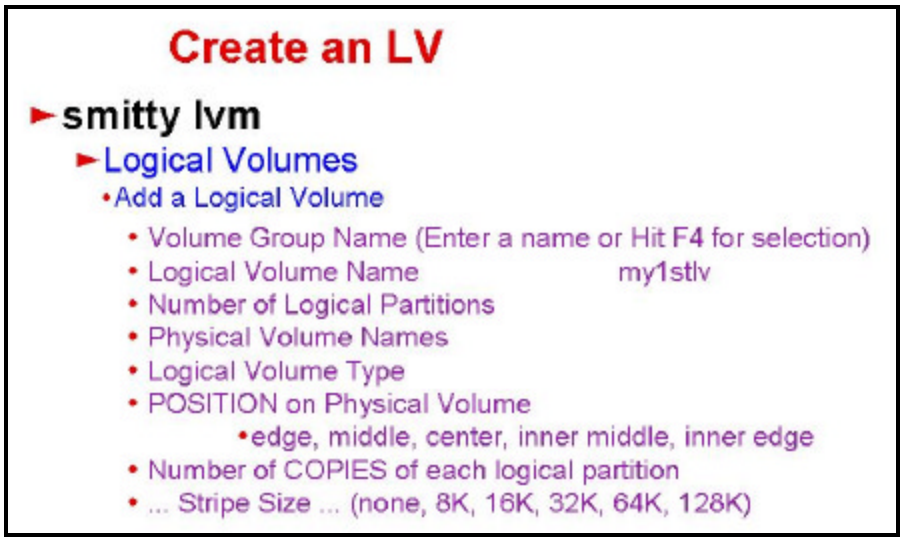


Create a VG

Now that you have a clear understanding of volume groups, let's look at another smitty fast path (**smitty lvm**) example that allows you to create a volume group. From the first panel, you would choose **Volume Groups**, and then select **Add a Volume Group**. The subsequent panels will show you choices for:

- **Volume Group Name**—in our example, we would key in 'myvg.'
- **Physical Partition Size**—is specified in megabytes. You can hit F4 to select 64 megabytes, or 128 megabytes for large disks.
- **Physical Volume Names**—Again, you can hit F4 to make a selection.
- **Activate Volume Group at Start up**—The default is 'yes.' You will usually leave this default value 'as is.'
- **Create VG Concurrent**—The default is 'no.' A concurrent volume group is a VG that is accessible by more than one server at a time. Concurrent volume groups are generally used in High Availability Cluster Multi-Processing (HACMP™) environments. Therefore, usually, you will leave this default value as is.

[NOTE: Some of the choices shown on the panels will have asterisks behind them, which means the values are required. Additionally, if there is a plus (+) sign on the right-hand side of the panel, it means that you can hit the F4 key to select from a list of choices.]

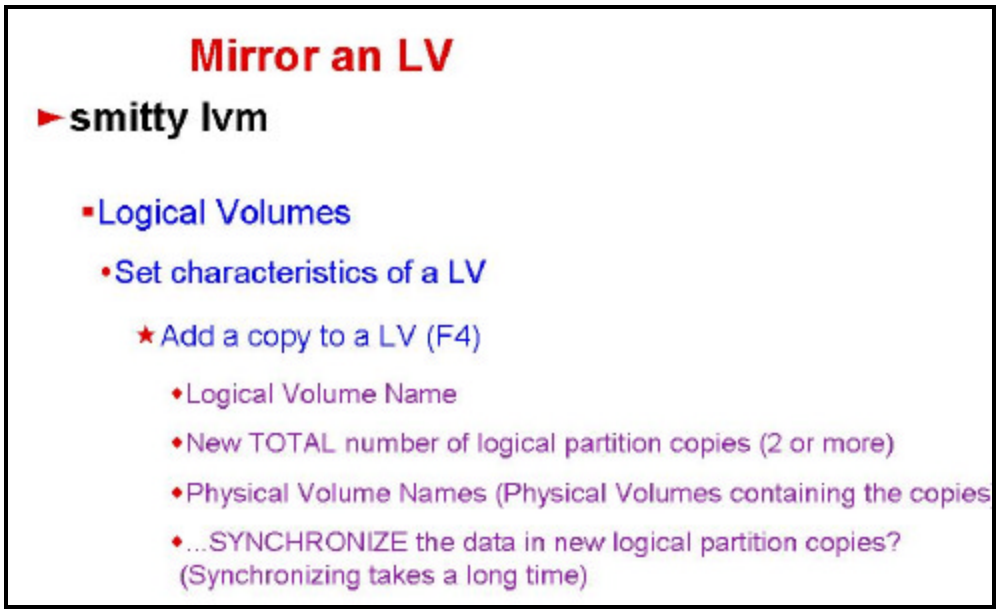


Create an LV

Once the volume group has been created, you can create a logical volume that will reside within that VG. The fast path **smitty lvm** can be used to create a logical volume. From the first panel, choose **Logical Volumes**, then select **Add a Logical Volume**. The subsequent panels will show you choices for:

- **Volume Group Name**—You can enter a name or hit F4 for the selection of the available VG names for this system. Then, hit Enter.
- **Logical Volume Name**—You must enter an LV name. In our example, we would key in 'my1stlv.'
- **Number of Logical Partitions**—Enter the number of logical partitions you need for this logical volume.
- **Physical Volume Names**—You can enter the physical volume names for this LV. Or, you can let the system assign the names from the available physical volume names that are already on the server.
- **Logical Volume Type**—You can also take the default value here, which is 'JFS.' Or, you can specify another logical volume type (either: JFS Log, Logical Volume Copy, or Raw LV).
- **POSITION on Physical Volume**—This value tells the server where on the physical volume you want this logical volume to be positioned. The choices here are: edge, middle, center, inner middle, and inner edge. The default value is 'middle'.
- **Number of COPIES of each logical partition**—The default value is '1'.
- **Stripe Size**—The choices are: none, 8K, 16K, 32K, 64K, and 128K. The default value is 'Not Striped'.

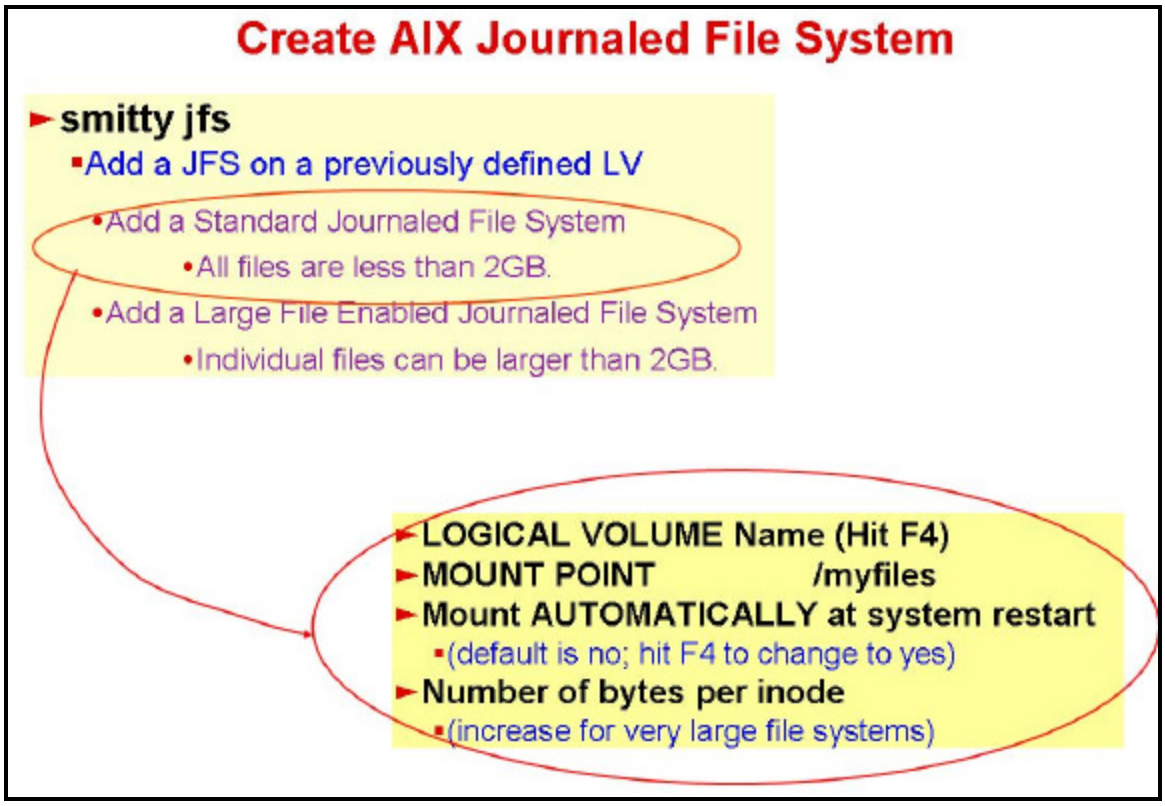
[NOTE: The last three values are best left at the default value unless you are savvy with performance tuning issues.]



Mirror an LV

There are times when you will need to make a copy of a logical volume, perhaps for mirroring or backup purposes. To do this, the fast path, **smitty lvm**, can be used to create a mirroring file for a logical volume. From the first panel you are shown, select **Logical Volumes**. Then choose **Set characteristics of a LV**. After that, you would choose **Add a copy to a LV** by hitting F4—which will return the options for you to make one, two, or more copies of the logical volume. You will also need to provide additional values...

- **Logical Volume Name**—This is a required field.
- **New TOTAL number of logical partition copies**—This value represents both the original logical volume and its mirror(s). Thus, you must enter at least two or more values.
- **Physical Volume Names**—This is the name of the physical volume that will contain the copy (or copies) of the original logical volume.
- **...SYNCHRONIZE the data in new logical partition copies?**—Be aware that if you answer 'yes,' synchronization takes a long time and will affect performance.

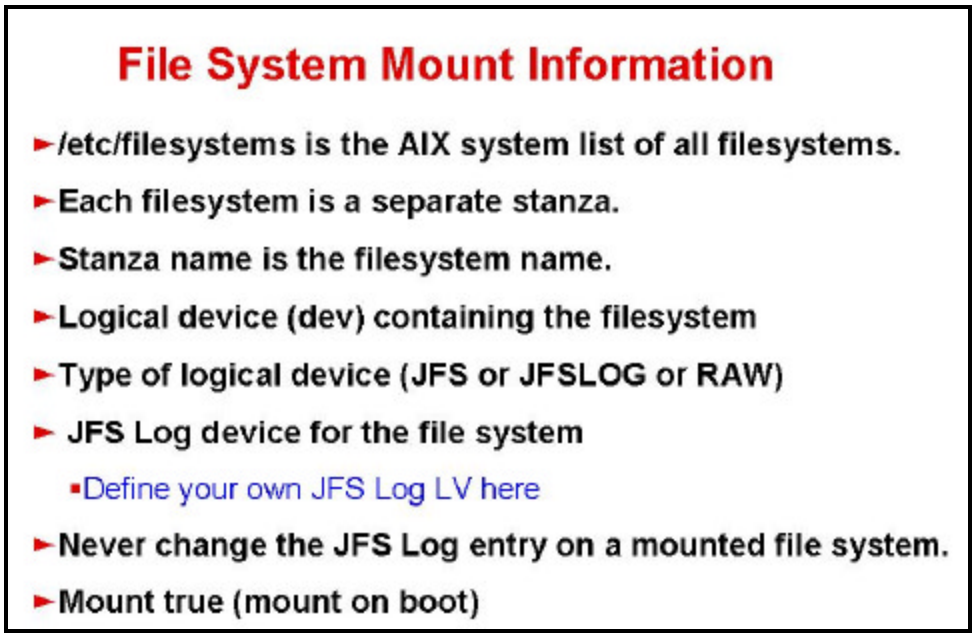


Create AIX Journaled File System

You can create a journal file system by using the fast path **smitty jfs**. From the first panel you will see, you should choose **Add a JFS on a previously defined LV**. Then, choose **Add a Standard Journaled File System** if each of the files that will be contained in this JFS will be less than two gigabytes in size. Otherwise, choose **Add a Large File Enabled Journaled File System** which will allow individual files to be larger than two gigabytes. Then, hit Enter.

Let's assume we have decided to add a standard journal file system. Here are the values we must enter...

- **LOGICAL VOLUME Name**—If you hit the F4 key, you will be provided with a display of the LV names from which you can choose.
- **MOUNT POINT**—Enter the mount point of your choice. For our example, we would enter /myfiles.
- **Mount AUTOMATICALLY at system restart**—The default is 'no.' However, if you hit F4, this value will change to 'yes.'
- **Number of bytes per inode**—You may want to increase this value for very large file systems. But otherwise, you would probably leave this alone, that is, you would use the default value. Then, hit Enter.



File System Mount Information

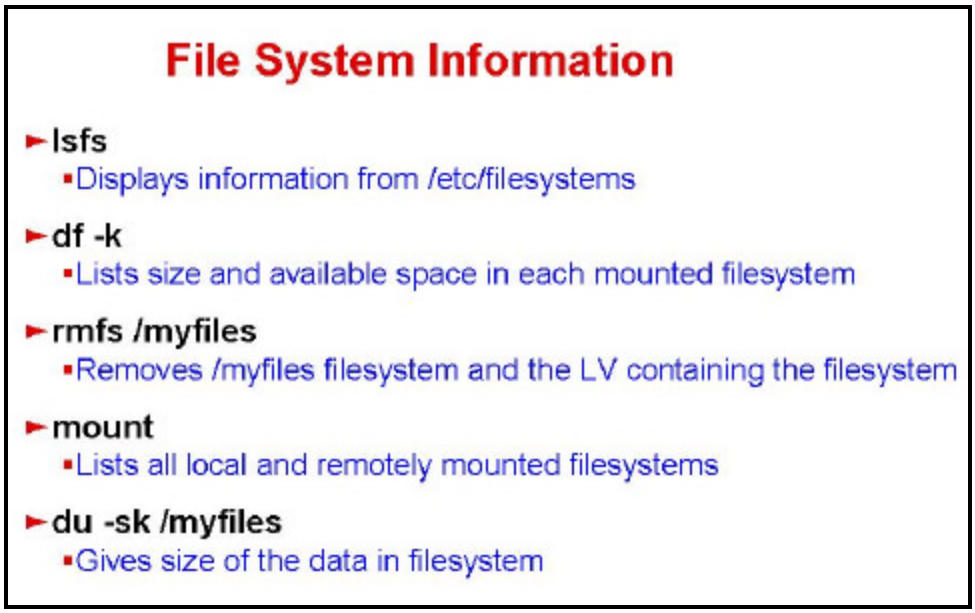
At this point, the file system has been mounted. The administrator now must create and dump the directories within this mount point. You might need to find out details about what files are loaded and where they are loaded so you can use that information to easily move or copy certain files into the new filesystem.

All of the information about the file system is centralized in the /etc/filesystems file. Most of the file system maintenance commands take their defaults from this file. The filesystems file documents the layout characteristics of the file systems. It also frees the administrator from having to enter and remember items such as the device where the file system resides.

Data about the characteristics of each individual filesystem is stored in a separate stanza. Each stanza name is the same as the name of the filesystem name for which it contains file system mount information. The stanza holds the name of the logical device (dev) containing the filesystem. It also has the type of the logical device (JFS, JFSLOG, or RAW).

Further, the stanza contains the JFS log device for the file system it represents. You can define your own JFS log logical volume here, if you would like. One piece of advice is that you never change the JFS Log entry on a mounted file system. In fact, no one should ever manually change anything in the /etc/filesystems file. This is because the entries to this file come in automatically through the various AIX commands.

There is a mount attribute which can have several different values. You will probably want to use the ‘true’ value since this will specify that the filesystem is to be mounted on boot. We will discuss the mounting file systems in more detail shortly.



File System Information

Several commands can help you review and manage the information in the /etc/filesystems file.

For example, the **lsfs** command displays information from /etc/filesystems.

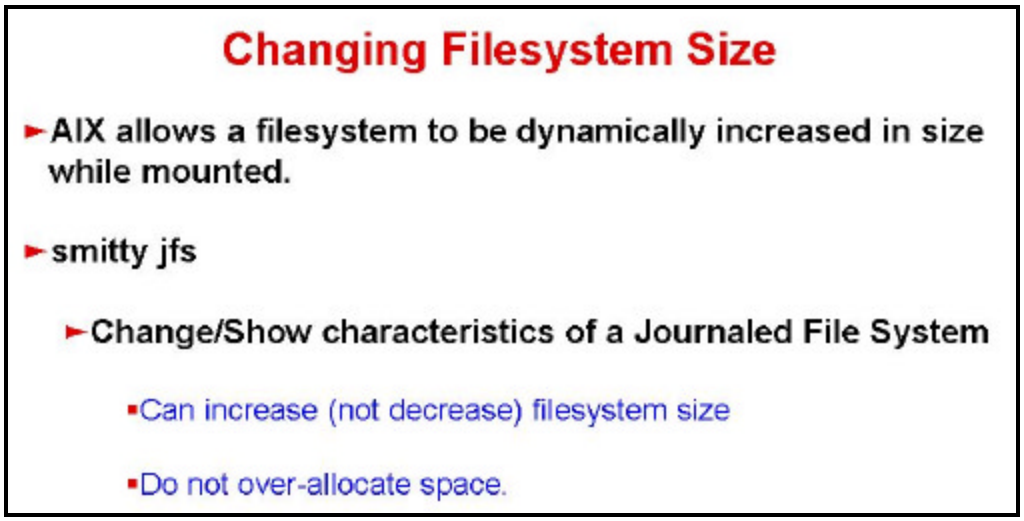
The **df –k** command lists the size and available space in each mounted filesystem. This can be helpful if you need to find out how much space is available to be reallocated to another filesytem.

The **df –k /usr** command lists the size and available space in user filesystems only.

The **rmfs /myfiles** command removes the /myfiles filesystem and also removes the Logical Volume that contains the filesystem.

The **mount** command lists all mounted filesystems, both local and remote.

The **du -sk /myfiles** command gives the size of the data in the /myfile filesystem.

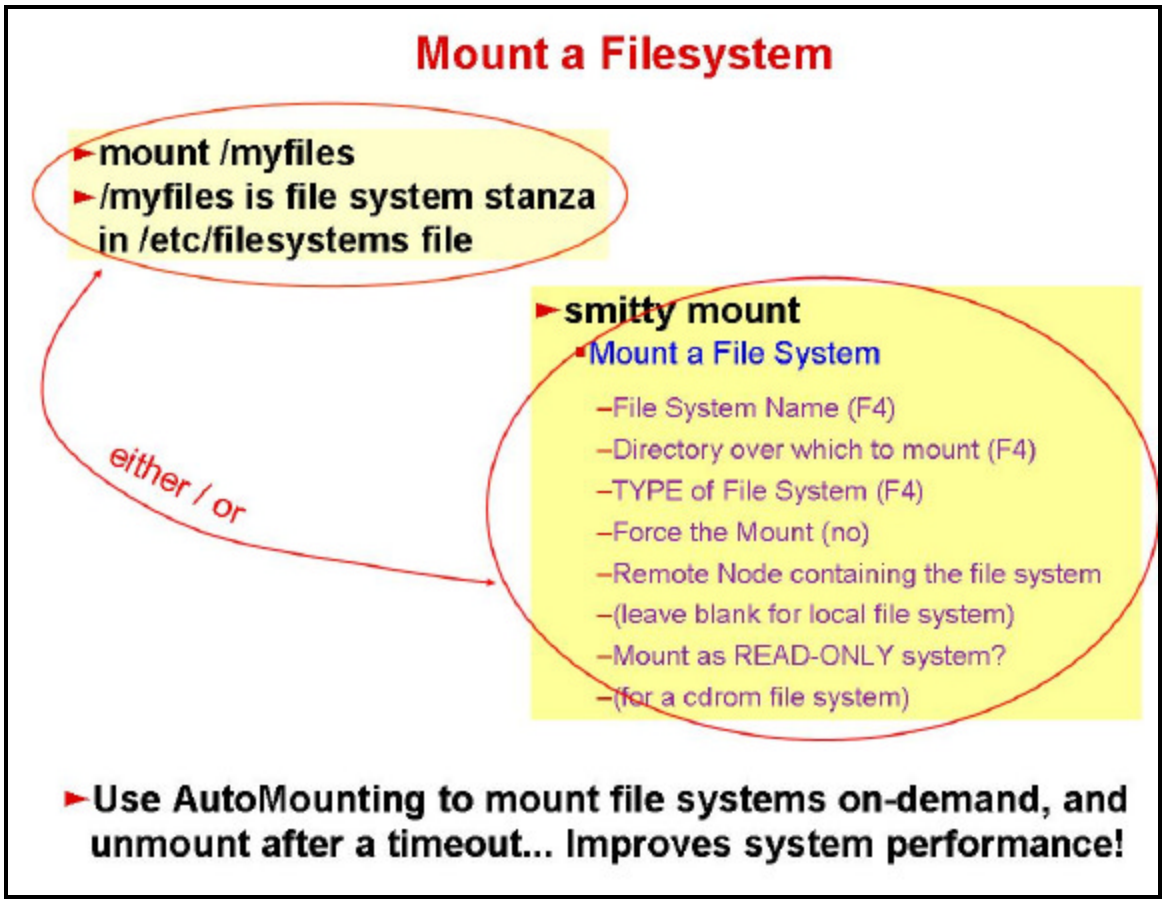


Changing Filesystem Size

The AIX operating system allows a filesystem to be dynamically increased in size while mounted—as long as there is space that is still available in that logical volume. The obvious point to make here is that the logical volume should always be larger than the filesystem that is mounted on it. If the LV and its mounted filesystem are equal in size, then dynamic allocation of additional storage will not be successful. In this case, a new and larger logical volume would have to be created elsewhere, then the filesystem would have to be mounted to it. Then, the original LV would be made larger, and finally, the filesystem would be remounted on the original LV that has now been laboriously made larger.

But, for most situations, the LV will be larger than the filesystem(s) it contains, and therefore, the ability to dynamically increase the filesystem's size is an excellent feature. It allows you to design the system so that there is not a hard stop when an application or system command needs additional storage.

To utilize this dynamic feature, you would use the fast path **smitty jfs**. Choose **Change/Show characteristics of a Journaled File System** from the first panel that is displayed. It will then ask you how much you would like to increase the size of the filesystem. You can only increase its size, not decrease it. You should also consider whether the amount of space you are increasing will later cause the logical volume to run short of free space to allocate to other filesystems that are also mounted on it.



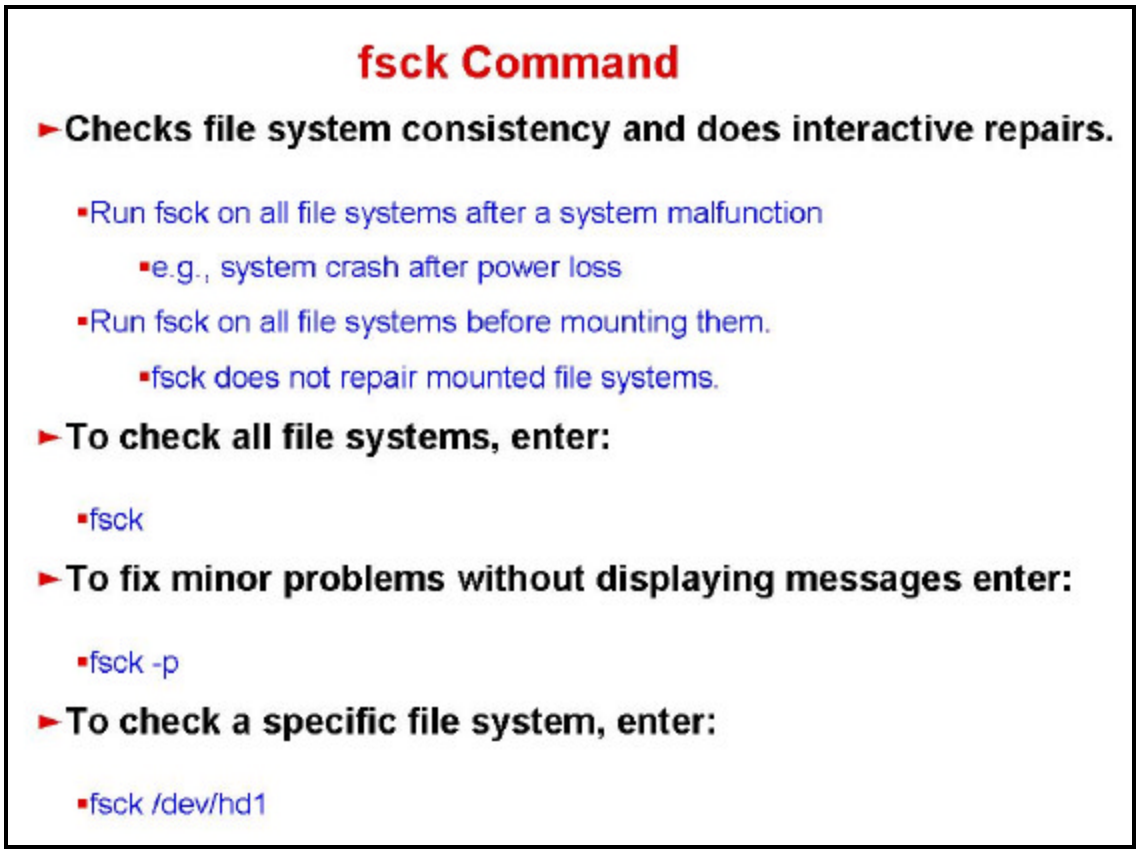
Mount a Filesystem

The mount command, shown here, will mount a filesystem to a logical volume.

You can also use the fast path **smitty mount**. Select **Mount a File System** from the first panel that is displayed. Then supply the following values...

- **File System Name**—Enter the name of the filesystem or hit the F4 key to view a list of filesystems.
- **Directory over which to mount**—Enter the directory name or hit the F4 key to select from a list.
- **TYPE of File System**—Hit the F4 key to select the filesystem type.
- **Force the Mount**—The default is 'no.'
- **Remote Node containing the file system**—Leave this blank if you are mounting a local filesystem.
- **Mount as READ-ONLY system?**—Specify 'yes' if the filesystem resides on a CD-ROM.

One additional note regarding mounting a filesystem: AIX supports AutoMounting—a nice feature that allows you to mount filesystems on-demand, and unmount them automatically after a timeout period. This function lets you mount only those filesystems that are needed at a point in time. Since unused filesystems will remain unmounted, system overhead will be reduced. AutoMount is a performance tuning device.



fsck Command

The standard UNIX File System Check command (fsck) can be used to verify the file system's consistency and interactively repairs the file system. fsck does not repair mounted file systems, so it should be run on all file systems before mounting them. And, it should always be run on file systems after a system malfunction, such as a system crash after a power loss.

To check the consistency of all file systems, enter **fsck**.

To display minor problems without displaying messages, enter **fsck –p**.

To check a specific file system, enter fsck **/dev/hd1**.

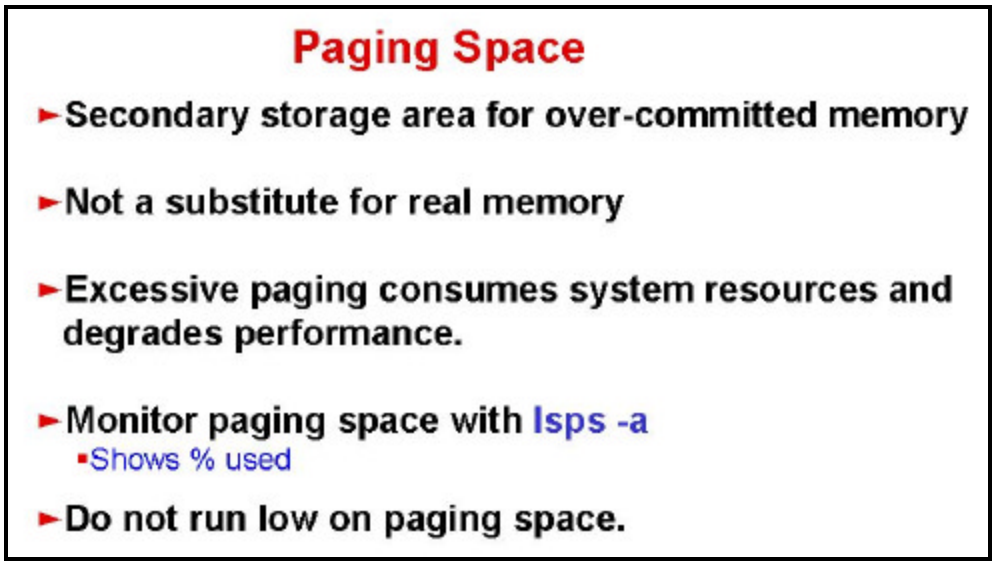
Create a JFS Log for a File System

- **Create an LV of type 'JFSLOG' with two logical partitions.**
- **Format JFSLOG LV using 'logform' command.**
- **Use smitty jfs to create a file system to be added on the new LV.**
- **If the file system is mounted, unmount it...do NOT modify /etc/filesystems while file system is mounted.**
  - Modifying JFSLOG entry of a mounted file system will corrupt the file system.
- **Modify JFSLOG entry for file system in /etc/filesystems.**
- **Mount file system.**

Create a JFS Log for a File System

If you want to create a log file for a file system, you must follow this procedure...

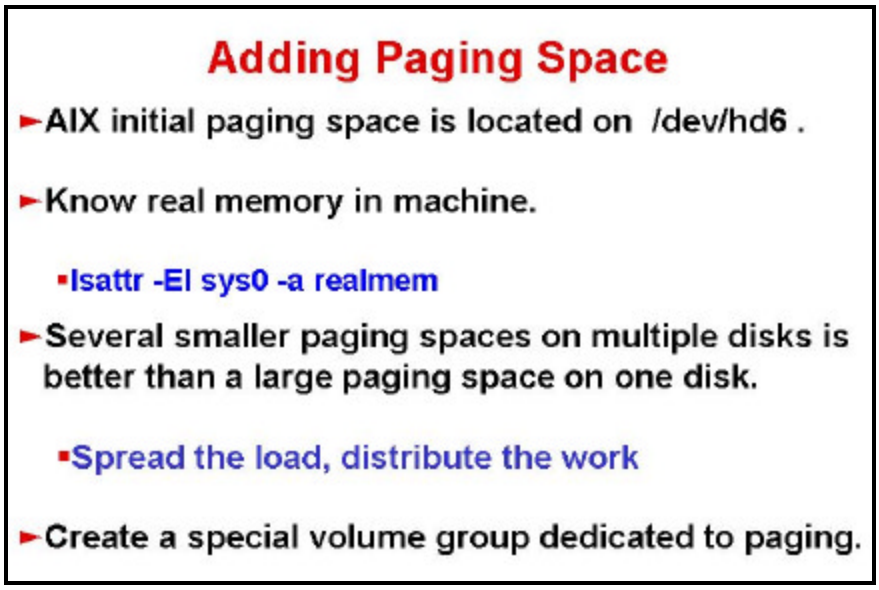
1. Create a logical volume of type 'JFSLOG' that has two logical partitions.
2. Format the JFSLOG logical volume by using the logform command.
3. Using the fast path **smitty jfs**, create a file system that will be added on the logical volume created and formatted in steps 1 and 2.
4. If the file system is mounted, you must unmount it. Do NOT modify the /etc/filesystems file when the file system is mounted. Modifying JFSLOG entry of a mounted file system will corrupt the file system.
5. Modify the JFSLOG entry for the file system in /etc/filesystems.
6. Mount the file system.



Paging Space

A paging space is a secondary storage area for over-committed memory and is a means of improving system performance somewhat—when used well. However, paging space should not be considered a substitute for real memory. It is very slow as compared to memory. This is because excessive paging (going back and forth from memory to disk, a.k.a. 'thrashing') also consumes system resources and can significantly degrade performance.

To understand how well or poorly your paging spaces are serving your system performance goals, use the **lsps -a** command. This will monitor paging space by showing the percentage of the paging space that is being used. As you review these monitors, keep in mind that running low on paging space is not recommended. So, if the percentage is a large number, you should do something to resolve this volume of paging.



Adding Paging Space Example

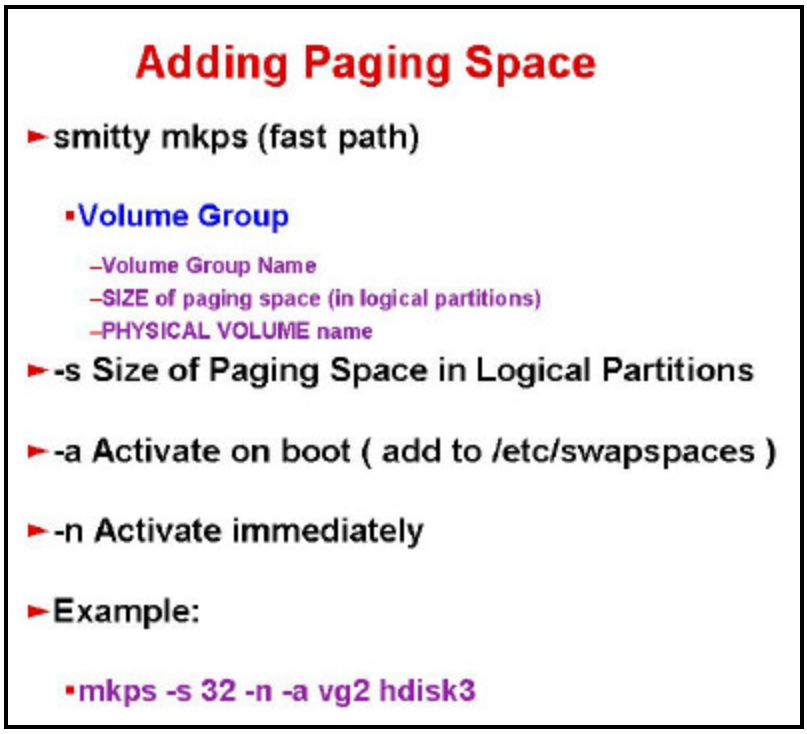
Now that you understand the value of paging spaces, let's talk about how you might add paging space—perhaps because the paging space is dangerously high. Here's what you need to know.

AIX initial paging space is located on the /dev/hd6 logical volume.

You should also know how much real memory your server has. You can use the bootinfo command discussed earlier in this course. Or, you can determine this by using the List Attributes (lsattr) command, shown in blue on this chart.

The optimal setup from a performance perspective for paging spaces is that they are created to reside as smaller spaces spread across multiple disks. Do not set up one large paging space on a single disk drive. You should keep in mind the concept of spreading the load and distributing the work.

It is also recommended that you create a special volume group that is dedicated to paging.



Adding Paging Space Example

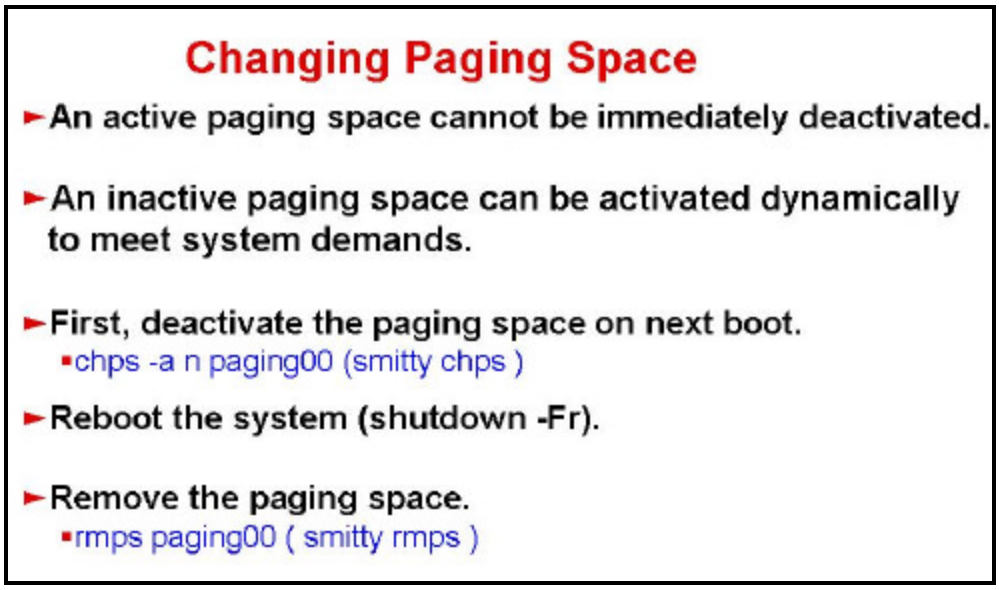
Let's look at how you would set up paging spaces on your server. Once again, you can use a fast path **smitty mkps**. Select **Volume Group** from the panel that is displayed. Then, specify the following as you are prompted for them...

- **Volume Group Name**—Hit F4 to get a list from which to choose.
- **SIZE of paging space**—This number must be specified in increments of whole logical partitions.
- **PHYSICAL VOLUME name**— Again, the F4 button will provide a selection list.

The Make Paging Space (**mkps**) command can be used directly. mkps takes several flags and options...

- The **-s** flag is followed by a number that represents the number of logical partitions that will, together, represent the size of the paging space.
- The **-a** flag activates the paging space on subsequent boots. It accomplishes this by adding the new paging space to the /etc/swapspaces file.
- The **-n** flag activates the paging space immediately.

The mkps command example shown at the bottom of this chart would allocate 32 logical partitions to be activated immediately and also activated on boot for devices vg2 and hdisk3.



Changing Paging Space

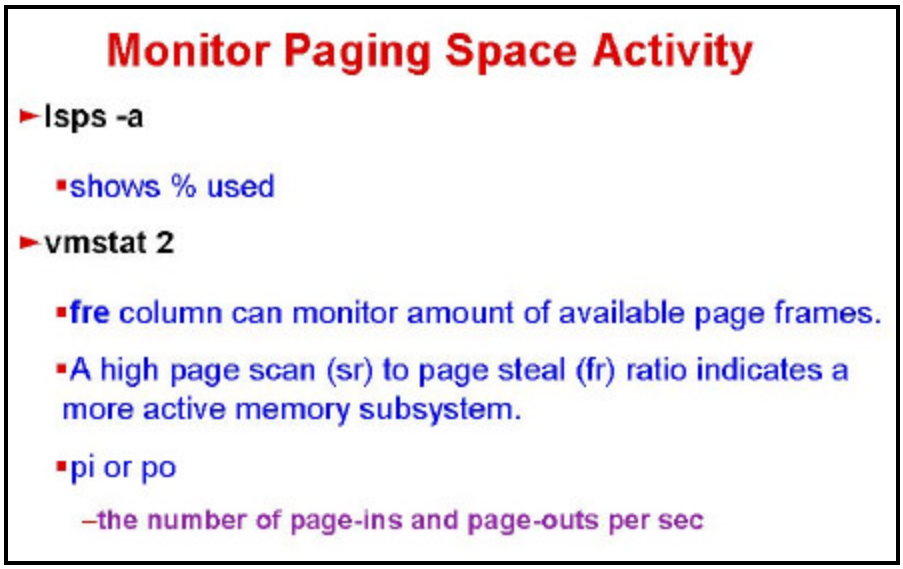
You can also change paging spaces as long as they are not active. If they are active, they are being used and cannot be deactivated. Instead, you must wait to deactivate that paging space on the next boot.

The Change Paging Space (chps) command would be used to deactivate the paging space(s). Notice the chps command example shown on this chart. The –a flag causes the command to make changes to the specified paging space at the next system start. The 'n' defines the change that needs to be performed to cause the paging space to be inactive (at boot). Of course, you could use the fast path smitty chps to accomplish this same instruction.

After you have issued the chps command, you would initiate a shutdown and immediately reboot via the **shutdown –Fr** command. After the reboot has completed, the paging space will have been deactivated so that you can change or remove it.

You can remove a paging space by using the Remove Paging Space (rmps) command (or the fast path **smitty rmps**). Look at the rmps command example shown on this chart. As you can see, it is a simple command that requires only one parameter—the paging space name.

An inactive paging space can be activated dynamically to meet system demands. But to belabor the point... an active paging space cannot be dynamically deactivated until the next system boot.

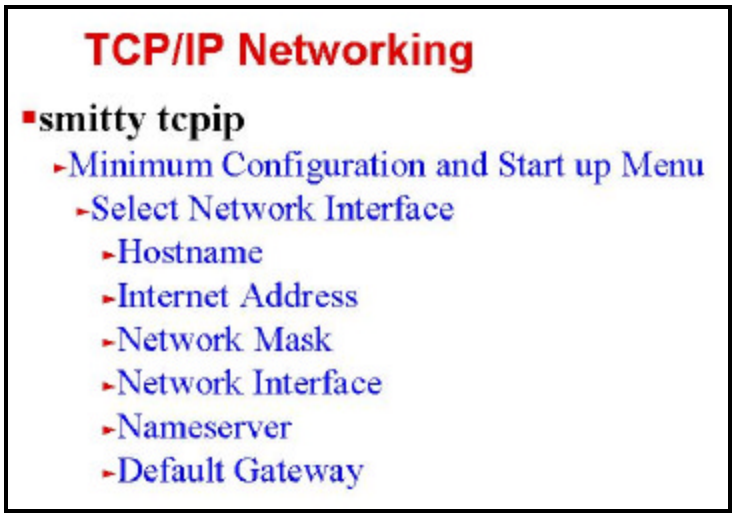


Monitor Paging Space Activity

As already mentioned, the List Paging Space (lsps) command can be used to observe activity within the paging spaces. You will remember that the **lsps –a** command shows the percentage of the paging space that is being used—the higher the percentage, the worse your performance will probably become.

The Virtual Memory Statistics (vmstat) command reports on virtual memory statistics. This screen-based report has some valuable information...

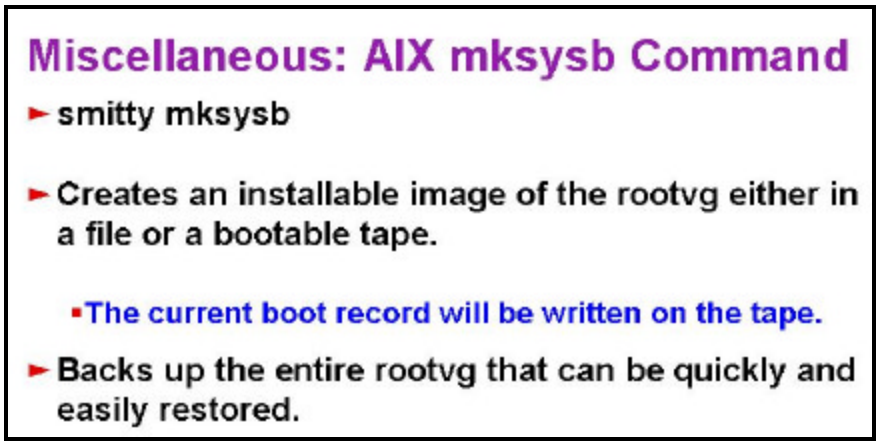
- The **fre** value reports the size of the page space that is free and available.
- The **sr** value shows the number of pages scanned by a page-replacement algorithm; while the **fr** value shows the number of actual page steals. A high sr-to-fr ratio indicates a more active memory subsystem.
- The **pi** value represents the number of pages that are paged in (per second) from the paging space. The **po** value similarly represents the number of pages that are paged out (per second) to the paging space. These two numbers give you a very good initial idea of how active (busy) your page-in's and page-out's are.



TCP/IP Networking

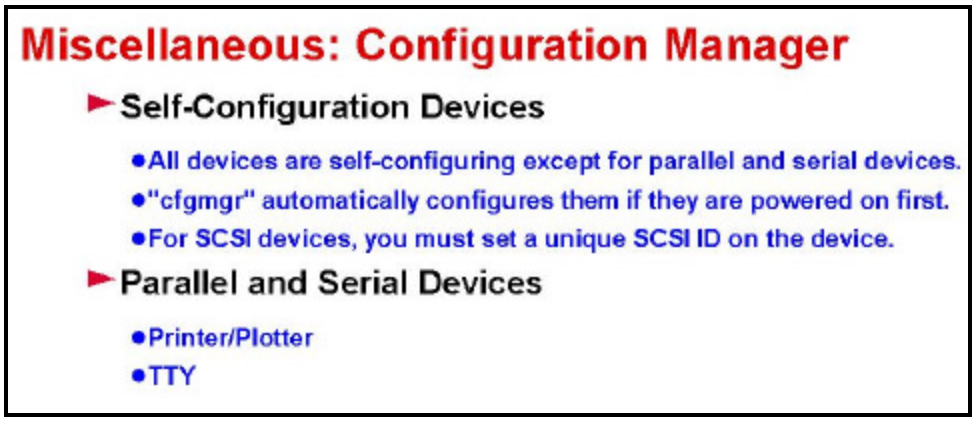
The fast path **smitty tcpip** can be used to configure your AIX server so that it can join a TCP/IP network. After this configuration, the pSeries server will be available, via secure methods, to remote Web-based users who are appropriately authorized. When invoking the fast path, the panel you will see is the **Minimum Configuration and Start up Menu**, from which you will click on **Select Network Interface**. You will then have the opportunity to enter the following values—all of which (except for the network interface) will have been provided to the administrator by the company's network manager...

- **Hostname**—This required value sets the name of the host. If using a domain naming system, the domain and any subdomains must be specified.
- **Internet Address**—This required value sets the Internet address of the host. Specify the address in dotted decimal notation. Each network interface on the host should have a unique Internet address. The following is the standard format for setting the Internet address: 136.10.31.2
- **Network Mask**—This optional value specifies the mask that the gateway should use in determining the appropriate subnetwork for routing. The subnet mask is a set of four bytes, as in the Internet address. The subnet mask consists of high bits (1s) corresponding to the bit positions of the network and subnetwork address, and low bits (0s) corresponding to the bit positions of the host address.
- **Network Interface**—This value is required. It specifies a particular network interface (e.g., en0 or en1). In particular, **en0** is used for the Internet.
- **Nameserver**—This optional value specifies the Internet address of the name server that the host uses for name resolution, if applicable. The address should be entered in dotted decimal notation, for example: 136.1.0.1
- **Default Gateway**—This optional value sets the gateway address for a static route. Specify the address in dotted decimal notation. The following is the standard format for setting the gateway address for a static route: 192.9.52.0



Miscellaneous: AIX mksysb Command

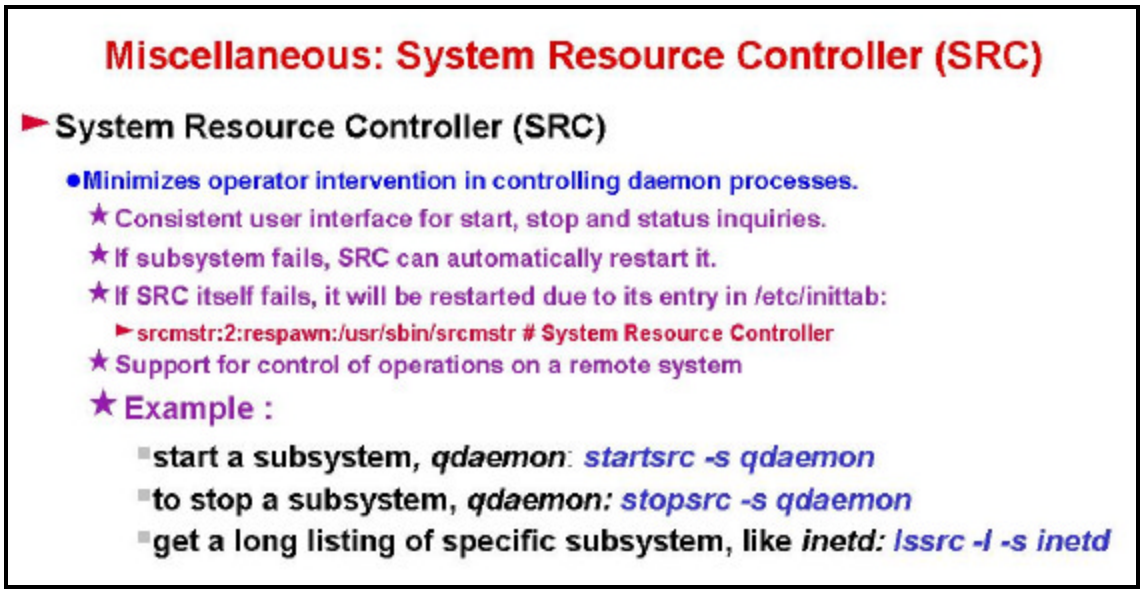
The fast path **smitty mksysb** creates an installable image of the boot that resides in the rootvg. The resulting image is either in a file or a bootable tape. It is a bit different than the bosboot command—bosboot creates a boot image that gets the system up and running, while mksysb creates a backup of the file systems that are contained in the entire rootvg so that rootvg can be quickly and easily restored.



Miscellaneous: Configuration Manager

Any devices you want to configure after the AIX configuration itself has already been accomplished is easy to do since almost all of them are self-configuring—except for parallel and serial devices. The **cfgmgr** command is used to automatically configure these self-configuring devices as long as they are powered on prior to the cfgmgr run. For SCSI devices, you will need to set a unique SCSI ID on the device in order for it to be self-configuring.

Parallel and serial devices, such as printers, plotters, and TTY devices, can be added and configured via **smitty devices** panels.



Miscellaneous: System Resource Controller (SRC)

The SRC minimizes the operator intervention that might otherwise be required to control daemon processes.

It provides a consistent user interface for start, stop, and status inquiries.

If a subsystem should fail for any reason, the SRC can automatically restart it successfully.

The SRC provides support for operational control on a remote system.

The SRC is started during system initialization with a record for the /usr/sbin/srcmstr daemon in the /etc/inittab file. Because of its entry in the /etc/inittab file, if the SRC itself were to fail, it would also be restarted automatically. You can see an example of this entry shown in red on this chart.

For example, if the qdaemon (which enables users to use the system printer) is not working properly, you can type in **startsrc -s qdaemon** to start the qdaemon. Similarly, you can enter **stopsrc -s qdaemon** to stop the qdaemon.

The lssrc command provides flags and parameters for specifying the subsystem by name or PID, for listing all subsystems, for requesting a short or long status report, and for requesting the status of SRC resources—either locally or on remote hosts. To obtain a long listing of a specific subsystem, such as inetd, you would type in **lssrc -1 –s inetd**.

Now you can administer an AIX server environment environment with ease!

Summary

We have covered quite an array of diverse AIX administration topics in this last hour. Hopefully, you like what you see and will go to the links section of this course to do more in-depth research on what is available for the sophisticated, automated, and efficient administration of your AIX server environment.

The white papers and Redbooks are a good place to start. You will find the information available on the pSeries Information Center invaluable.

Good luck as you move forward with your AIX administration skills.

Links

Here are some related links to the topics discussed in this course...

IBM Redbooks
ibm.com/redbooks

- System Management Interface Tool (SMIT)

IBM Publications
www.elink.ibmlink.ibm.com/public/applications/publications/cgibin/pbi.cgi?CTY=US

AIX Technical Reference Library
ibm.com/servers/aix/library

pSeries and AIX Information Center
http://publib16.boulder.ibm.com/pseries/en_US/infocenter/base/index.htm

AIX Commands You Should Not Leave Home Without
ibm.com*/developerworks/eserver/articles/dutta_cmds.html*

AIX Updates V2: How to work the puzzle
ibm.com*/developerworks/eserver/articles/dutta_work.html*

AIX 5L V5.2: What's in it for you?
ibm.com*/developerworks/eserver/articles/dutta_AIX5L.html*

AIX Administrator How-to's (AIX pSeries Information Center)
http://publib16.boulder.ibm.com/pseries/en_US/infocenter/base/HowToAdmin.htm#CATEGORY_0

White paper: "JFS Overview" by Steve Best
ibm.com*/developerworks/library/l-jfs.html*

Trademarks and Disclaimers

The following are trademarks of the International Business Machines Corporation in the United States and/or other countries: IBM, eServer, AIX, AIX 5L, pSeries, POWER, and HACMP.

All other products may be trademarks or registered trademarks of their respective companies.

This publication was produced in the United States. IBM may not offer the products, services or features discussed in this document in other countries, and the information may be subject to change without notice. Consult your local IBM business contact for information on the product or services available in your area.

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