Sybase Adaptive Server Enterprise / SQL Server HOWTO

Kian Spongsveen

<sybase(at)kian.org>

2003−09−21

Revision History

Revision 1.0	2003−09−21	Revised by: ks
Initial public release		
Revision 0.9	2003−09−05	Revised by: ks
Initial draft release		

Describes installation and configuration of Sybase Adaptive Server Enterprise (formerly known as SQL Server) relational database server on the Linux platform, together with basic usage.

Table of Contents

1. Introduction

Sybase ASE / SQL Server is an industry−strength high−performance database solution used by some of the largest corporations worldwide. It is also, in my personal opinion, one of the easiest database servers to learn and to work with. It has a wide range of extremely advanced options, probably the most advanced replication in the market, and a set of various add−on products that will even support opaque access of data from competing products. There is a Linux version available for download which is free even for production use. Sharing a common heritage with Microsoft SQL Server (Sybase sold the code to Microsoft, but the T−SQL dialect remains almost the same in both products), Sybase ASE is the easiest way of getting "MS SQL on Linux". Having said that, you may also understand by now that Sybase ASE could deserve the label "the best−kept secret in the market". Welcome to a free helping of a commercial database server that basically runs most of Wall Street and stock exchanges around the world − now on your home PC or business server.

1.1. Copyright and License

1.2. Contact information

Kian Spongsveen is the originator and currently the maintainer of this HOWTO. Please send feedback, suggestions for improvement, questions to me at `<sybase@kian.org>`. English is not my native language, so please send me helpful corrections to grammar and spelling as well as any technical issues you find in this document.

1.3. What is Not Covered

There are some parts I have skipped since this is a HOWTO for installation, configuration and very basic usage. Some points that could be of interest are:

- SySAM − a utility needed to install software license certificates for certain advanced options that you have to pay extra for in version 12.5.0.x and up.
- Full−Text Search − an add−on for searching TEXT columns
- Data replication − Two products are available for free with the server (SQL Remote in 11.9.2.x, ASE Replicator in ASE 12.5.0.1 and up). There is also Replication Server, a separate product also available for Linux.

1.4. History

Sybase started as a company in 1984, making a relational database management system (RDBMS) called SQL Server. At version 4.9.2, Microsoft licensed the code and released their SQL Server 6.0. Sybase version−inflated their next version to "System X". The product has been enhanced and improved over the years, it was first released on Linux with a pilot of version 11.0.3.3 which was already a mature version on it's way out at the time of release in 1999. The product was renamed from SQL Server to Adaptive Server Enterprise (ASE) when version 11.5 was released, the name is a bit of a mouthful and some users simply refer to it as "ace". Later, version 11.9.2 was released both as a free developer version and a fully supported production version on Linux. Currently Linux is considered a tier−1 platform and the current version as of this writing, 12.5.0.x, as well as the beta release of the next version, 12.5.1, are available for the Linux OS.

A brief note about the version numbering. Sybase patches are called EBFs, or Emergency Bug Fixes, and each has a unique number (product number) for that particular set of fixes on each particular architecture. Since this made comparing the various bug fixes across OS platforms difficult, the notation of ESD (Electronic Software Distribution) was introduced. So for a major release (Maintenance Release) 12.5.0.0 there were some EBFs numbered ESD#1, ESD#2 and so on, until a minor release with significant enhancements (Interim Release) numbered 12.5.0.1 and then further EBFs are based on this. If a Linux machine and a Solaris machine both run 12.5.0.3 ESD #1, you know they will have exactly the same bugfixes corrected even if the EBF product number of course differs.

1.5. Availability

I have briefly mentioned the available versions. You can download the free versions from http://linux.sybase.com, and if you need a production server with support etc. you can order a boxed CD from your local Sybase office or http://eshop.sybase.com.

- SQL Server 11.0.3.3 ESD #6 is no longer supported or maintained. The reason it is still made available is that it is free for production use on Linux. Read the license file shipped with the product for details. The 11.0.3 versions are the earliest that were Y2K−certified.
- ASE 11.9.2.x was the first release on Linux where you could get either a free developer license or buy support. It is no longer actively maintained or supported. I describe it in this document since it was the first production use supported release for Linux and hence became very popular.
- ASE 12.5.0.x is the current release on all platforms, including Linux.
- ASE 12.5.1 is the next release, currently in beta. There is a beta for Linux available.

The functionality varies in the releases, certain significant enhancements have been made in the current versions. However, major corporations based their infrastructure on SQL Server 11.0.x until very recently (and some of them may have old installations still very active) so don't think of it as outdated. Some of the new enhancements may be of interest to you, others will only sooth your pain if you were annoyed by the lack of them over a period of time first.

2. Installation

The different versions of ASE for Linux that are available have slightly different installation and configuration procedures. The installation documents that come with the product are quite detailed so I will instead try to focus on some alternatives in order to avoid common problems. In particular, I will describe how to install without RPM since the RPM utility is a functionality unique to Linux and most Sybase DBAs with background from other types of UNIX prefer to avoid this format and use the traditional installation instead. However, I will also describe how you can play it safer when using RPM.

2.1. Super-quick RPM installation for the impatient

If you are just doing a test, simply install the `sybase-common` and `sybase-ase` RPMs using **rpm -ihv** as root. The installation should then create a sybase user and start the configuration scripts automatically. For version 11.0.3.3, only the `sybase-ase` package is needed.

```
bash$ su - root
Password:
bash# rpm -ihv sybase-common-12.5-1.rpm sybase-ase-12.5-1.rpm
```

I strongly recommend that you read through any text files shipped with the software (`README` for instance), and also the documents "Release Bulletin" and "Installation Guide" available on the Sybase documentation website (also known as SyBooks). These will contain lots of important information, and I have attempted to avoid duplication of those efforts as much as possible.

2.2. Pre-install Steps For All Installations

1. The first thing to check is whether your Linux installation is able to run the version of ASE you have selected. The RPM utility will attempt to check this during installation, but you can check the dependencies in advance with **rpm -qRp <package>**. RPM will check against the RPM database of installed products on your machine, if these have lower version numbers than the requirements you will get an error message and it is first necessary to update these packages to be able to install. Note that you will also get an error if the package name is not found in the RPM database. This can happen if you installed software without using RPM (e.g. compiled from source, not SRPMs) or the package has a different name from what the ASE packages list as dependencies (e.g. RedHat places the Linux kernel in a package named "kernel", SuSE uses k_deflt for the default kernel and other names for other supplied kernels such as k_smp). You can then check the kernel version with **uname -r** and the glibc version will most likely be what is reported by **ldd --version**, but also do **ls -l /lib/** and check the actual filenames that the symlinks `/lib/libc.so`, `/lib/libpthread.so` and `/lib/librt.so` link to.

   Certified OS levels

   The distribution Sybase has chosen to certify on is RedHat. For installation of 11.0.3.3 or 11.9.2 you can probably use any distribution, for the newer enterprise features of 12.5.0.x you will have to check exactly which distribution is certified and only use that particular Linux distribution and release level. Don't expect anything else than the certified OS level to work. The installation document lists the minimum required kernel and glibc versions.

   Glibc-specific issues

There are some known issues with glibc that cause serious problems with ASE. Basically, you will see this as a Segmentation Fault when starting the server (the error log will say something about process infected with signal 11 and then dump out a stack trace.). These problems are seen above version 12.5.0.1 on any uncertified platform, including currently RedHat 9.0. You must use a certified Linux distribution and version (RedHat 7.2, RedHat Advanced Server 2.1) and make sure the proper fixes are in the i686 glibc libraries (update them from RedHat Network). In particular, at the time of this writing the current ASE version (12.5.0.3) does not work on the current RedHat version (9.0) – nor on the beta of the next Enterprise Server (3.0) due to glibc issues! RedHat is working on this, and the release of Enterprise Server should be in order. See RedHat bugs 90002 and 102995 as well as the research done by Sybase under CR 326398. Hopefully this will eventually trickle up to the FSF/GNU distribution point for glibc and then to the various other distributions.

2. Check the available disk space. You will initially need space for installing the binaries and for installing the system databases. This is listed in the Installation Guide, but you will need additional space for the databases you create and you will have to make some estimates of how much you need for this.

   Table 1. Default Installation Directory

Version	Minimum RAM Dedicated to ASE	Application Disk Space	System Database Devices
11.0.3.3	Ü15 MB	110 MB	17 MB master 12 MB sysprocsdev
11.9.2.x	Ü32 MB	140 MB	25 MB master 45 MB sysprocsdev
12.5.0.x	Ü50 MB	215 MB	30 MB master 100 MB sysprocsdev

3. After checking that ASE can be installed, create the sybase user group and then the sybase user as a member of it. This is an ordinary user that will be used mainly for starting the database server, so the user has to own all the ASE–related files. I suggest you keep the home directory of this user together with the home directories of other normal users, instead of using the installation directory of the ASE software as a home directory. Use **adduser**, **useradd** or whatever GUI utility your distribution provides you with.

   ```
   bash$ su - root
   bash# groupadd sybase
   bash# useradd -g sybase -d /home/sybase -c "Sybase ASE DBA account" -p Hard2Guess sybase
   ```

   The installation scripts will not attempt to create the user if you have done so yourself first.

Once these steps are done, we are ready to unload the software onto the base directory that ASE will be installed in. By default, the RPM packages are configured to place this under `/opt/`*`<sybase-directory>`* where <sybase–directory> is:

Table 2. Default Installation Directory

Version	Directory
11.0.3.3	/opt/sybase/
11.9.2.x	/opt/sybase–11.9.2/
12.5.0.x	/opt/sybase–12.5/

You can choose a different location if you like, either with the *--relocate* parameter to **rpm**, or by unpacking to a different directory if you are converting to .tgz file first. Again, I recommend keeping a normal home directory for the sybase user, a directory for the ASE software and a third location for the database storage.

2.3. Unloading the Software to Disk, Part 1: Making Some Considered Decisions

After preparing the environment, it is time to write the software to disk. You can either install with RPM or convert the RPM packages to another format first. These options will be covered here, and why you would want to choose one or the other.

2.3.1. The issues with RPM and how to avoid them

So, why is RPM bad? Well, as long as you install the latest ASE patches on the latest working RedHat version with the latest patches you may not have a lot of problems. However, you may also find that the RPM installations will cause the following problems depending on exactly which version of ASE (or EBFs) you are installing on which RedHat version (or other distribution).

Known Problems with RPM and Their Workarounds

Problem: RPM installation of ASE or applying EBFs will truncate the `/etc/ld.so.conf` file.

Symptom: After RPM installation of ASE if will truncate your `/etc/ld.so.conf` file to 0 bytes, making your OS installation forget where shared libraries are found. This may affect X, sshd and several others.

Caused by: the post–install scripts. You can view them with **rpm –qp –scripts <*package*>**. They try to use a varaible named `$RPM_INSTALL_PREFIX` which is apparently not set by newer versions of the RPM utility.

Kludge 1: first copy all the files that will be destroyed by the installation. `/etc/ld.so.conf`, `.profile` and `.cshrc` for the sybase user should be copied to a safe place before installing. Then install as per the Sybase instructions and copy the files back when you are done. When the correct `/etc/ld.so.conf` is in place, run **ldconfig** as root.

Kludge 2: First set the `$RPM_INSTALL_PREFIX` to the top–level directory where the ASE files are to be installed (what will become `$SYBASE`).

Kludge 3: Make RPM install without running the scripts; **rpm –ihv ––noscripts**

Problem: .profile and .cshrc overwritten

Symptom: RPM installation will overwrite the `.profile` and `.cshrc` files in the $SYBASE directory. If this is the home directory of the sybase user and you have personalized these to include preferred settings you will have to rewrite them afterwards.

Caused by: The files are located in the $SYBASE/install/ directory in ASE 12.5. The script $SYBASE/install/sybinstall.sh which is called during RPM installation will copy these to $SYBASE.

Kludge 1: Backup these files before installation, copy back after installation is done.

Kludge 2: Make RPM install without the scripts; **rpm --ihv --noscripts**

Kludge 3: Create a home directory ($HOME) for the sybase user separate from the installation directory ($SYBASE).

Problem: RPM claims that installed libraries or even the kernel are not of the approved version or not installed.

Problem: RPM installation reports that certain versions of OS rpms such as kernel or glibc must be installed, but you verify they are OK.

Caused by: RPM installations may fail if it is unable to determine whether you have the proper versions of kernel and glibc installed. For instance, SuSE calls its default kernel rpm package "k_deflt", other kernels are named after their main functionality (smp, athlon etc.). The Sybase RPMs search for "kernel" since that is what RedHat uses.

*Kludge:*Make absolutely sure you do actually have the required level of these packages. Then force-install with **rpm -ihv --nodeps --force**.

2.4. Unloading the Software to Disk, Part 2: Getting the Job Done

Unless the above problem descriptions scared you, you may want to use RPM with a few extra modifications. Instead of letting RPM try to be helpful but end up causing trouble, we will disable the friendly attempts and have to do some manual corrections ourselves afterwards.

2.4.1. The Manual RPM Installation

Just to be on the safe side, we first copy out the files that are at risk; /etc/ld.so.conf, <sybase user home>/.profile and <sybase user home>/.cshrc. The switches I have chosen to add are:

--relocate

--nodeps

--noscripts

Now, mount the CD or download the files with FTP and **cd** to this directory. As root, install the files with:

```
rpm -ihv --relocate /opt/sybase-12.5=/opt/sybase --nodeps --noscripts sybase-common* sybase-ase*
```

As part of the server configuration we now need to make the changes we disabled when we stopped the RPM scrips from excuting using the --noscripts.

2.4.2. Using the RPM Alternatives

A couple of common utilities for installing without RPM are rpm2tgz and alien. You can use any of these to first convert into a tgz package and then unpack the files without running any of the RPM scripts. The distribuiton of the RPM utility also ships with a program called rpm2cpio which will convert to a cpio archive.

- alien: http://www.kitenet.net/programs/alien/

- rpm2tgz: This is apparently the name of a utility shipped with Slackware and TurboLinux. The TurboLinux files indicate that it is simply a wrapper for rpm2cpio and cpio2tgz, the latter of these will actually unpack the cpio archive into a temporary directory and repack it as tgz.
- rpm2cpio: This is part of the RedHat Package Manager (RPM) distribution so it should be installed if you have rpm installed. You will need cpio together with it.

The alien utility can either unpack the rpm using **alien −g**, or you can create a tgz archive with **alien −t**.

```
bash$ su -
Password:
bash# alien -t sybase-ase*.rpm sybas-common*.rpm
sybase-common-12.5.tgz generated
sybase-ase-12.5.tgz generated
```

The rpm2tgz utility does the same thing, converts the rpm into a tgz archive.

```
rpm2tgz sybase-ase*.rpm
rpm2tgz sybase-common*.rpm
```

The distribution of RPM contains the rpm2cpio utility, it reads an rpm from a file or STDIN and outputs a cpio archive on STDOUT.

```
rpm2cpio sybase-ase-12.5-3.i386.rpm > sybase-ase-12.5-3.i386.cpio
rpm2cpio sybase-common-12.5-3.i386.rpm > sybase-common-12.5-3.i386.cpio
```

Once you have created a .tgz archive from the RPM, you can unpack it. Since the path in the archive is relative to /, you need to do this as root.

```
bash$ su -
Password:
bash# tar -xvzf sybase-ase*.tgz
bash# tar -xvzf sybase-common*.tgz
```

For a cpio archive, unpack it with the cpio command:

```
cpio --extract --make-directories --verbose
```

3. Configuring an ASE Server

So far we have copied the software into the file system, now we will do the actual setup of a database server. First we need to make sure the sybase user is set up properly, then we can start the configuration utilities and enter the correct settings there.

3.1. Pre−Configuration steps

1. To begin with, read the `LICENSE` file if it exists. Once you agree with this, continue down this list.
2. There are some useful steps done by the installation scripts in the RPM packages that you may want to do by hand if you installed with extra RPM switches or used alien. One is that directories in ASE 12.5 get shortcuts created with shorter names. ASE−12_5 gets a symlink named ASE, OCS−12_5 has a symlink named OCS. You can set these up yourself to save some typing later.

```
bash$ cd /opt/sybase
bash$ ln −s SYBASE_ASE ASE
bash$ ln −s SYBASE_OCS OCS
bash$ ln −s shared−1_0 shared
bash$ ln −s SYSAM−1_0 SYSAM
```

3. Change the installation to be owned by the sybase user.

```
bash$ su − root
bash# chown −R sybase.sybase /opt/sybase
bash# chmod −R ug+w /opt/sybase
```

   This is more or less what the `sybinstall.sh` script does for you. In SQL Server 11.0.3.3 there is a script which attempts to set all permissions right, `$SYBASE/install/setperm_all`
4. Make sure you allow ASE to allocate the necessary amount of shared memory. Your current maximum is what you find from **ipcs −lm** (in KiloBytes) or **cat /proc/sys/kernel/shmmax** (in bytes) or **sysctl kernel.shmmax** (in bytes). This can be set with

```
sysctl −w kernel.shmmax=[number of bytes]
```

   or by directly

```
echo [number of bytes] > /proc/sys/kernel/shmmax
```

   as root. Make sure this is permanently changed on every reboot by either inserting a line in `/etc/sysctl.conf` or place the echo command in a rc startup file (such as rc.local) as the Sybase documentation says. Depending upon the version, here is the memory requirements for ASE:

   Table 3. ASE Memory Requirements

Version	Minimum Memory	shmmax	"total memory"
11.0.3.3	Ü15 MB	15360000 bytes	7500 2K pages
11.9.2.x	32 MB	33554432 bytes	16384 2K pages
12.5.0.x	50 MB	52428800 bytes	25600 2K pages

5. Make sure the sybase user has all relevant environment variables set correctly.

   Table 4. Sybase environment variables

Version	Variable	Set to
11.0.3.3 and 11.9.2.x		
	$LANG	Must not be set!
	$LC_ALL	"default"

	$SYBASE	Top−level of installation, default is /opt/sybase/
	$PATH	Add /opt/sybase/bin to the start of the existing $PATH
	$LD_LIBRARY_PATH	Add /opt/sybase/lib to the start of the existing $LD_LIBRARY_PATH
	$SYBPLATFORM	"linux"
12.5.0.x	All of the above plus	
	$SYBASE_ASE	Subdirectory (not full path) to the ASE part of the installation
	$SYBASE_OCS	Subdirectory (not full path) to the Open Client / Open Server part of the installation

Create a script with all environment variables that you can source in, or modify startup files such as `.cshrc` or `.profile`. Here are the needed settings:

sh/bash/ksh type shells

```
unset LANG
export LC_ALL="default"
export SYBASE="/opt/sybase/"
export PATH="$SYBASE/bin:$SYBASE/$SYBASE_ASE/bin:$SYBASE/$SYBASE_OCS/bin:
$SYBASE/$SYBASE_SYSAM/bin:$PATH"
export LD_LIBRARY_PATH="$SYBASE/lib:$SYBASE/$SYBASE_ASE/lib:$SYBASE/$SYBASE_OCS/lib:
$LD_LIBRARY_PATH"
export SYBPLATFORM="linux"
export SYBASE_ASE="ASE"
export SYBASE_OCS="OCS"
export SYBASE_FTS="FTS"
export SYBASE_SYSAM="SYSAM"
```

csh/tcsh type shells

```
unsetenv LANG
setenv LC_ALL default
setenv SYBASE /opt/sybase/
setenv PATH $SYBASE/bin:$SYBASE/$SYBASE_ASE/bin:$SYBASE/$SYBASE_OCS/bin:
$SYBASE/$SYBASE_SYSAM/bin:$PATH
setenv LD_LIBRARY_PATH=$SYBASE/lib:$SYBASE/$SYBASE_ASE/lib:$SYBASE/$SYBASE_OCS/lib:
$LD_LIBRARY_PATH
setenv SYBPLATFORM=linux
setenv SYBASE_ASE ASE
setenv SYBASE_OCS OCS
setenv SYBASE_FTS FTS
setenv SYBASE_SYSAM SYSAM
```

6. Prepare the data area.

3.2. Preparing the Data Area

Since the file system for the software is now prepared, we can decide where to locate the data storage used to hold the data inserted into the databases. The first decision is whether to use raw devices or file system. Traditionally, raw devices were used. You had to partition a disk and set a certain label to have it recognized as raw. Linux didn't have this previously and the Sybase virtual devices were instead written to files in a file system.

So, what is the difference between these approaches and when should you choose each? Unfortunately, it is one of the areas where the answer is a very clear and unambiguos "it depends". Raw devices were introduced first with kernel patches and then included in the 2.4 series. You use the command **raw** to create bindings between partitions and the `/dev/raw`*n* devices. This means that you will need to have a free partition on a disk. Changing this later can also be difficult. The advantages of raw devices are that you know exactly where on disk they are located − you can place them on the fastest cylinders if you like. Writes to the raw devices will be unbuffered, they will go directly to disk. In case of a failure (such as power outage, disk controller failure or process crash) the finished writes are guaranteed to be on disk instead of only having been written to a buffer waiting for the disk to be ready. Also, the ASE server will to asynchronous I/O on these devices. It can queue a series of I/O instead of having to do a single I/O, wait for it to finish and then do the next operation. While the I/O result is pending, the ASE server will perform other work. With a virtual device written to a file on a file system (ext2, ext3 and Reiserfs are all supported), the writes will normally be done to the file system buffers. The actual file may be located anywhere on the partition and may not be contiguous. Note that there is no need for a journalling file system since these will only log the changes to the inode, not the contents of the data blocks. The Sybase ASE transaction log will do exactly this kind of "journalling" of the data modifications for you. The buffering means that reading data not held in ASE cache may take place from the file system buffer instead of having to read it from disk, thereby speeding up these. However, writes will probably take longer time to finish since they have to pass through the file system driver layer. In order to have safe writes to file system devices you can configure ASE to open the device using the O_DSYNC flag, which will cause writes to be flushed to disc immediately when writing.

In ASE 11.0.3.3 ESD #6 there is support for raw devices. This will need certain patches shipped in the GPL directory, one for raw device access, one for KAIO. There are instructions for how to apply these to the kernels and distributions that were current at the time.

In 11.9.2.x it was decided not to include any experimental raw device support before the Linux kernel officially included and supported this. In order to guarantee safe writes, the O_DSYNC flag is always enabled by default. You can globally disable this with traceflag 1625; add −T1625 as a parameter to the RUN_SERVER file.

In ASE 12.5.0.x there was a change starting in 12.5.0.2 where the server is now compiled to use the kernel 2.4 functionality. That means you can create raw devices on OS level and configure ASE to use these without any special patches. If you prefer file system devices, you can set the O_DSYNC flag to be used with the *`dsync=true`* parameter when creating the device with **disk init**, or the **sp_deviceattr** stored procedure to modify it later.

3.3. Starting the Configuration

To configure a SQL Server 11.0.3.3 you log in as the sybase user and start the **sybinit** utility. You can run this in a console, make changes and hit **Ctrl**−A for "accept" and **Ctrl**−X to go back to the previous menu.

In ASE 11.9.2.x and 12.5.0.x, you normally use the **srvbuild** command, but you will need an X server to display it. For those not using X, an option is to use the shell script "sybinit4ever", available on the web from SyPron, see the 3rd party utilities section later. This will give you a classic sybinit menu interface to the configuration process. An alternative from Sybase is the **srvbuildres** utility which uses a resource file containing all configuration options which you can edit with your settings, these options are described in an appendix of the Installation Guide.

The exact entries for configuring a server are described in more detail in the /opt/sybase/doc/howto/howto−ase−quickstart.html file from the ASE 11.0.3.3 documentation RPM, and in the "Configuration Guide for UNIX" for the newer versions. The information you will need for all versions are:

- IP address of the interface which ASE will listen on. Use the network IP of the machine or 127.0.0.1 if you are only going to accept local connections.
- TCP port which ASE will listen on. Choose any port not in use. Port 2025 was used before, 4100 is the current default. Use 1433 if you are trying to trick applications into believing this is an MS SQL Server.
- Path to the master device (raw device or file).
- Path to the sybsystemprocs device (raw device or file).
- Character set and sort order for your data (you want to get this right the first time, changing it later involves copying all data out to files and load them back in). ISO_1 character set should be sufficient for most purposes. Unicode is only supported in the 12.5.0.x releases.
- Logical server name − this is just a placeholder to identify an ASE instance. In the examples I have named the server SYBASE.

Once you have decided on these settings, start the configuration program. If you use **srvbuild**, you will need to set your $DISPLAY variable correctly. Type in the requested settings and let the configuration utility build the server. Some parts of this, such as creating the database devices and loading the system stored procedures, will take time. Once the process has finished, you server should have been built and be up and running, ready for requests.

3.4. Finishing Off

If you want automatic startup and shutdown, consider the rc.sybase script a good start. You can copy this to where all your other startup scripts are and create the correct start and stop links from your runlevel directories.

4. Basic Usage

The intention behind this section is to allow you to get started with the Sybase−specific parts quickly. You will still need to know a little bit about relational databases, creating tables, designing the database and so on. You will see how to start and stop the server, execute commands and a little bit about backups.

4.1. Starting and Stopping the Server

Starting the server is done with the **startserver** utility. During installation a script file was created named `RUN_` and then the name of the server. Since the file name will vary with your server name, it is generally referred to as the `RUN_SERVER` file. To start your server, make sure all environment variables (in particular `$SYBASE`) are set correctly. Check with **showserver** that the server is not already running − there is no risk of starting the server twice, but you may get some confusing error messages about the master device already being in use. Assuming the server is not started, execute the following: **startserver −f RUN_SYBASE**Assuming that you named your server "SYBASE" at install time, this should make various messages starting with a timestamp scroll over your screen and hopefully end with messages about the default character set and sort order. If not, read the messages carefully for any errors. If they have scrolled out of your screen buffer you can read the error log file instead, located in the same directory as the `RUN_SERVER` file.

4.2. Connecting to the Server

Once the server is up and running, you can attempt to connect to it. Start with the simplest tool, which is also the traditional utility for Sybase administration and use; **isql**. It takes many parameters, as listed in the Utility Guide, but we only need a few. Make your first connection like this: **isql −Usa −P −S*server***Replace the *−S* parameter with your logical server name that you chose during installation.

Remember to change the blank sa password

The *−U* parameter is the login name, we are here logging in as "*sa*" the Sybase equivalent of "*root*". The password specified with the *−P* parameter is blank, which is the default password for this user. It should be fairly obvious that you don't want to leave your most powerful login with a blank password. Note that the ASE logins are independent from the OS logins and passwords.

The result should be a simple prompt:

```
1>
```

If the connection did not succeed, make sure the server is running and that the port is reported by **netstat −na** to have a LISTENING state.

You can now start typing T−SQL commands and use the keyword "**go**" as a terminator.

```
isql −Usa −P −SSERVER
1> select @@version
2> go
 −−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−
 Adaptive Server Enterprise/12.5.0.3/EBF 10980 ESD#1/P/Linux Intel/Linux 2.4.18−18.7.xsmp i686/rel12503/1919/32−bit/OPT/Mon Mar 24 20:49:12 2003

(1 row affected)
```

4.3. Your First Useful Commands

The first thing to do is to set a password for "sa" and then create a new login with less privileges that we can use to play around with. By default you are placed in the `master` database once you are logged in. We will make our new login default to an example database instead to avoid creating objects in the master database by mistake. So, we first set a password for sa by calling the stored procedure **sp_password** with old and new passwords as parameters:

```
1> exec sp_password NULL, "Secr3t"
2> go
Password correctly set.
(return status = 0)
```

Now sa has a new password, changed from the old null default. We add a new login with **sp_addlogin**:

```
1> exec sp_addlogin "sybtest", "SomePass"
2> go
Password correctly set.
Account unlocked.
New login created.
(return status = 0)
```

We can log out by telling isql to disconnect using the **exit** command.

```
1> exit
```

Since this is not a T−SQL command but a directive to isql we don't need to terminate with "go".

The **isql** command can also be used non−interactively to apply scripts of T−SQL to the server. One such script that is shipped with the server is `installpubs2`, located in the scripts directory. This is a very simple example database for a bookshop or a publishing house holding data about books, authors, publishers and so on. It is used in Sybase manuals and training courses and also in some SQL books. Microsoft SQL Server contains a similar database in addition to the Northwind example database they have added. In order to create the database, use the *−i* parameter to read the script in. Have a look at the file first so you understand the basics of what it is doing; it will create a database named pubs2 and several tables populated with data. It is time to execute the script. We'll do this as sa who will also become the owner (dbo − database owner) of the database. We redirect the output to a file we call `errors.out`. The *−e* parameter tells isql to also echo the T−SQL commands to the same file, giving more output but making it easier to match any errors to the commands causing them.

```
bash$ cd $SYBASE/$SYBASE_ASE/scripts
bash$ isql −Usa −PSecr3t −SSYBASE −iinstallpubs2 −e −oerrors.out
```

In order to allow our new login full privileges to this sample database we change ownership of the database to the new login. Here's how we give the database away with **sp_changedbowner**:

```
isql −Usa −PSecr3t −SSYBASE
1> use pubs2
2> go
1> exec sp_changedbowner sybtest
2> go
DBCC execution completed. If DBCC printed error messages, contact a user with
System Administrator (SA) role.
Database owner changed.
(return status = 0)
```

We can now log in interactively as our new user and check what has been installed.

```
1> exit
bash$ isql -Usybtest -PSomePass -SSYBASE
1> use pubs2
2> go
1> sp_help
2> go
[Lots of output deleted - the command displays all objects in the current database]
```

(Note that we don't actually have to use "exec" to execute a stored procedure, the server will assume any non−keyword is a procedure.)

```
1> quit
```

One last command as the sa login in order to make life more convenient when we continue to use our new login − we make the new pubs2 database the default database.

```
1> exit
bash$ isql -Usa -PSecr3t -SSYBASE
1> sp_modifylogin sybtest, "defdb", "pubs2"
2> go
Default database changed.
(return status = 0)
```

4.4. Stopping the Server

In order to stop the server in a controlled fashion, log in as sa and issue the **shutdown** command.

```
bash$ isql -Usa -PSecr3t -SSYBASE
1> shutdown
2> go
Server SHUTDOWN by request.
The SQL Server is terminating this process.
CT-LIBRARY error:
        ct_results(): network packet layer: internal net library error: Net-Library operation terminated due to disconnect
```

You will immediately be disconnected and a message is printed by isql to warn you of this fact. You can check the error log for a message about the server being shutdown and you can verify that the process is no longer running with **showserver**.

4.5. Maintenance

One of the most important aspects of being a database administrator may be the backup. The I/O load of a relational database means little rest for the hard drives and once a drive fails the database is in need of serious disaster recovery. Even a mistyped command may result in the need to revert to a previous backup generation. For this purpose, a separate server application called the Backup Server is used. It is by default named the same as your server with an extension of BCK. Start it with **startserver −f RUN_SYBASE_BCK**. Certain commands typed into the isql propmt will be forwarded from the `dataserver` process to the `backupserver` process, which will then proceed with the actual backup (in Sybase terminology, this is a database dump) while processing in the database continues unaffected. You should schedule database dumps (usually via cron) to run at low activity hours. A typical full database bacup is simply done like this:

```
isql -Usa -PSecr3t -SSYBASE
1> dump database pubs2 to "/mnt/backup/pubs2.bkp"
2> go
Backup Server session id is:  8.  Use this value when executing the
'sp_volchanged' system stored procedure after fulfilling any volume change
request from the Backup Server.
```

```
Backup Server: 4.41.1.1: Creating new disk file /mnt/backup/pubs2.bkp.
Backup Server: 6.28.1.1: Dumpfile name 'pubs2011710275E  ' section number 1
mounted on disk file '/mnt/backup/pubs2.bkp'
Backup Server: 4.58.1.1: Database pubs2: 396 kilobytes DUMPed.
Backup Server: 4.58.1.1: Database pubs2: 602 kilobytes DUMPed.
Backup Server: 3.43.1.1: Dump phase number 1 completed.
Backup Server: 3.43.1.1: Dump phase number 2 completed.
Backup Server: 3.43.1.1: Dump phase number 3 completed.
Backup Server: 4.58.1.1: Database pubs2: 610 kilobytes DUMPed.
Backup Server: 3.42.1.1: DUMP is complete (database pubs2).
1>
```

You can restore this back into your database using the load database command.

As time passes while users are doing modifications in the database, adding, deleting or changing data, all operations are being written to the transaction log. This keeps track of changes so they can be undone by an implicit or explicit rollback, or for the undo/redo phases of revocery at startup. This transaction log should normally be placed on a device of its own for several reasons, but a small test database can be created on a single mixed log and data device.

Apart from the performance benefits of spreading I/O, one reason for keeping the log and data separate is for recovery purposes. You can at regular intervals, depending upon your recovery needs, dump this log of changes to the database. Together with the full database dump, this transaction log dump now constitute an incremental backup. Should a restore become necessary, you can load the database dump, then load all subsequent transaction log dumps. There is even an "until_time" option to the load command enabling you to specify the exact time you want to restore until, abandoning any mistakes done after that time. Dumping the log is done with a similar syntax:

```
isql -Usa -PSecr3t -SSYBASE
1> dump transaction database to "/mnt/backup/dbtrandump2003_08_27_T23_32.bkp"
2> go
```

Note that we could not do this with pubs2 as it was not created with a separate log fragment.

Unless you keep dumping the transaction log, it will just keep growing until it fills up it's space and starts reporting error 1105. Users will be suspended and appear to be hanging while the situation remains unresolved. Dump the transaction log to file or tape, or simply truncate it if you don't use incremental backups.

```
isql -Usa -PSecr3t -SSYBASE
1> dump transaction pubs2 with truncate_only
2> go
1>
```

Other maintenance commands you should read up on are

- **DBCC**, the DataBase Consistency Checker which will verify that the physical integrity of the data structures on the ASE devices are OK.
- **update statistics**, which will make sure that ASE has a correct view of how your data is distributed in your tables, enabling it to make the best decisions of how to retrieve the data in the shortest possible time.

4.6. Other Sybase Utilities

There are some other external utilities that are useful to know. The Java based applications need the JDBC driver installed, this is included with newer servers under the product name jConnect. There is a separate CD that comes with ASE 11.9.2.x and 12.5.0.x called "PC Client CD". This is a set of Windows−based tools for administration, pop the CD into your Windows box and the installer will start up so you can select components from the CD.

* bcp or Bulk Copy is a command−line utility with a plethora of parameters that imports table data from flat files and exports data out to files.
* Sybase Central is a GUI tool for database adminstration. It used to be a native Win32 application, but in ASE 12.5.0.x it is now Java based and can be installed on Linux. Note that you also need to install and register the plug−in for administering ASE since Sybase Central is only a framework which is used for many Sybase products by registering their respective plugins.
* Jisql is a Java based GUI version of isql with some neat features such as command history and table and column name lookup.

5. ASE Architecture: Observing the Server

What follows is a brief description of the architecture of ASE and how you can peek into this a bit closer. In order to get an overview of what ASE is doing and how it is set up we'll use a combination of internal ASE commands and OS commands.

5.1. Processes

Each ASE server running on the machine will have at least one OS process, the `dataserver` binary, and may have many such running. A single server instance consists of at least one `dataserver` process. Under Sybase terminology, such `dataserver` processes that are cooperating and communicating with each other in shared memory are known as engines. For production use, one CPU on the server machine is often reserved as dedicated to host each Sybase engine. The engine may then be configured to hog this CPU, even when there is no active work it will idle loop polling for new incoming client connections to avoid context switches. Of course, this behaviour is entirely configurable and running one or more server instances on a single−CPU machine is not a problem − depending of course upon the load on these servers. As long as there is sufficient memory for each instance and they are started on different TCP ports, there is no problem having several instances on one machine − even of different versions.

A simple **ps** will show you the `dataserver` processes (by default you only have one), Sybase has provided a utility named **showserver** that will just show you the Sybase−related processes that are active. The **sp_sysmon** stored procedure will monitor ASE for a given time interval, then dump out several pages of global performance data. The Engine section shows how active the server really is, regardless of the CPU usage shown on OS level.

5.2. Physical Storage

The ASE server does I/O to the raw devices or files, these are represented internally as virtual devices. A database can reside on one or spread out on many of these virtual devices, and a virtual device can hold many databases if you want. You should locate the OS−level device files on fast disks and make sure they are not removed or messed with by other applications or sysadmins on a cleanup crusade. The path to the virtual devices are stored in the master..sysdevices table, you can list these with the **sp_helpdevice** stored procedure.

5.3. Network

The server listens for incoming connections on one or several TCP ports. You identify the server by the logical Sybase server name when you connect. This logical name is listed in the interfaces file, used by both ASE server and clients such as isql. When the ASE server is started, it finds it's name in the RUN_SERVER file, looks this up in the interfaces file, finds the master entry and starts a listener on the IP / port found there. When you start isql it also looks for the logical server name in the interfaces file, but looks for the query line instead. Normally this is the same IP and port, but it gives you the option of starting the server on several different IPs and ports and configure clients in different parts of the network to utilize different pathways to the server. JDBC does not use the interfaces file, but instead lets you use the IP and port as part of the URL.

You can observe the open port and established connections with **netstat** or **lsof −i**. It is also possible to trace the communication using **tcpdump** or **Ethereal**, these utilities have support for the Tabular Data Stream (TDS) protocol used in Sybase client−server connections.

Once a client has connected it will be visible inside ASE as a task, an internal process. These are not seen as separate OS processes, but can be listed with the **sp_who** stored procedure.

5.4. Memory

You can configure how much memory you want ASE to use down to a certain needed minimum and up to whatever your OS and your ASE version combination will allow you. Except for doing careful analysis and clever design and SQL writing, using more of the available memory is what makes databases speed up without changing hardware. By default, most of the memory you allow allocated to ASE is used for caching data to avoid disk I/O as much as possible. Another area of memory is used to cache stored procedures in a compiled form, enabling these to be readily re−used without having to read from disk as frequently. Smaller parts are reserved for various administrative memory structures needed by the server for keeping track of each user connection, each database and so on.

On OS level you can see this normally contiguous memory chunk with **ipcs −m**. Inside ASE you can use **sp_configure** to read and modify configuration parameters such as total memory. There are several ways of determining the efficiency of memory usage, this art is explained in the Performance and Tuning Guide.

5.5. Boot Process

You start the server using the startserver utility. This will call the RUN_* file that you specify on the command line. If you open this RUN_SERVER file in a text editor you will find it simply calls the `dataserver` executable with several parameters listed in the file. These are documented in the utility guide. The server reads its configuration file (specified by one of the parameters) and allocates the amount of shared memory stated in that file (NOTE: this is configured in 2KB pages, not bytes), then does it's own internal distribution of this memory for various purposes. Once the memory is available, the process of initializing (you can think of it as "mounting") the virtual devices used to store databases is started. When these are verified available and OK, the databases have to go through recovery. This means reading the write−ahead transaction log and comparing any changes recorded there to the actual data stored. Transactions are redone and undone as needed to get the databases to a clean and correct state. Once the process is done the databases will be online. For the system databases the same process is applied, except for the scratchpad database tempdb which will be totally overwritten with the model template database and any remaining space zeroed out. Finally the TCP port is opened and the server is ready to accept incoming client connections.

A. Languages for connecting to ASE

You can connect to ASE and perform SQL commands in all common computer languages.

- C / C++

  Sybase provides client development libraries under the name of Open Client Developer's Kit which is included in their SDK. There is also an Embedded SQL/C precompiler which enables you to add SQL code to your C programs

  If you prefer to use free / open source products, there is the FreeTDS project at http://www.freetds.org/. This is a re–implementation of Sybase's Tabular Data Stream (TDS), used for the client–server communication.

  Note that Microsoft has modified their SQL Server TDS dialect from their Sybase legacy and you are no longer guaranteed that Microsoft SQL Server clients can communicate with a Sybase server or vice versa. MS SQL Server 6.5 came with a patch to enable Sybase TDS compatibility, later versions (7.0, 2000) do not support the Sybase communication protocol.
- Java / JDBC

  Sybase ships the jConnect JDBC driver with ASE for Linux in ASE 12.5.0.x and up. There are both JDBC 1.0 and 2.0 implementations, labeled as jConnect 4.x and 5.x respectively. These are Type 4 JDBC drivers, which means they are 100% Java implementations of the TDS protocol.

  The FreeTDS project mentioned above also ships a JDBC driver.
- ODBC

  For Windows clients, the ODBC drivers are shipped on the "PC Client CD". This used to be a ODBC layer on top of the Open Client runtime libaries, but are being modified to speak TDS directly instead.

  There is also an ODBC driver available from FreeTDS. Unfortunately, it appears there is no unixODBC (http://www.unixodbc.org/) driver available yet.
- Perl

  You can use any of DBD::Sybase, DBD::ODBC, DBD::FreeTDS or Sybperl with the FreeTDS libraries. These modules are available from CPAN. You can find lots of details from Michael Peppler's site, http://www.mbay.net/~mpeppler/.
- Python

  A Python module for connecting to ASE (and documentation for it) can be downloaded from http://www.object–craft.com.au/projects/sybase/
- PHP

  PHP includes Sybase connectivity, based on the Open Client libraries. The PHP manual shows examples on connecting to an ASE server, sending queries and handling the result sets. See http://www.php.net/.
- Tcl

  An extension for connecting to ASE from tcl is available from http://www.nyx.net/~tpoindex/tcl.html

B. Technical Resources

- Sybase official sites
    - The main entry to all Sybase Linux information is at http://www.sybase.com/linux.
    - Direct links to the ASE−related parts are at http://www.sybase.com/ase_linux.
    - From the main http://www.sybase.com website, there are plenty of useful links, you can also go directly to the support site, http://www.sybase.com/support/. You may need to generate a login to get access to all the unrestricted information − registered support users get further access to their own account details in order to directly log and trace their issues on the web.
    - White papers etc are located at http://www.sybase.com/support/techdocs/.
    - Manuals are at http://www.sybase.com/manuals/, also known as "SyBooks". Choose "Adaptive Server Enterprise" as the product or go directly to http://sybooks.sybase.com/as.html. Of particular interest is perhaps the "Transact−SQL User's Guide", the "Administrator's Guide" and "Performance and Tuning Guide".
    - CodeXchange is a relatively new section of the Sybase site, intended for sharing code and tools. http://www.codexchange.sybase.com
- Unofficial sites
    - The International Sybase Users Group is at http://www.isug.com/, and they host the unofficial FAQ at http://www.isug.com/Sybase_FAQ/.
    - Lots of stuff you will need to know sooner or later, including several printed reference books, have been published by Rob Verschoor and can either be downloaded or ordered at http://www.sypron.nl. Plenty of good links are maintained at http://www.sypron.nl/syblinks.html.
    - The author of SybPerl, Michael Peppler, has written an ASE on Linux FAQ which is at http://www.mbay.net/~mpeppler/linux.html. The same pages of course host SybPerl and various other utilities and information.
    - Ed Barlow has written a set of very useful stored procedures for ASE administration, http://www.edbarlow.com.
- Third Party Utilities
    - Many people prefer a tool known as "sqsh" to replace the standard isql utility provided by Sybase. You can download this from http://www.sqsh.org.
    - If you connect from Java over JDBC using Sybase's jConnect driver, you may want to use iSQL−Viewer as a GUI replacement for isql or jisql. Note that result sets from Sybase stored procedures are not visible in the current release version (2.1.5), download the core classes from CVS until the next version is released. The author has corrected the problem and the versions checked into CVS handle multiple result sets returned by a query. Available at http://isql.sourceforge.net.
- News Groups
    - There is a public USENET discussion group named comp.databases.sybase, but you can also find lots of official Sybase news groups at the news server forums.sybase.com.
- Mailing lists
    - You can sign up for information about new white papers, patches, announcements etc. when you register at http://www.sybase.com. This will only send you updates on the topics you sign up for, Sybase is very strict on avoiding spam and unwanted advertising.
    - There is an ASE Linux mailing list hosted at http://mathforum.org/epigone/ase−linux−list.

- There is also the SYBASE−L mailing list, see info from the URL http://www.lsoft.com/scripts/wl.exe?SL1=SYBASE−L&H=LISTSERV.UCSB.EDU.

C. GNU General Public License

C.1. Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software−−to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

C.1.1. GNU GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.

b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

a) Accompany it with the complete corresponding machine–readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine−readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty−free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER

PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

C.1.2. How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

```
one line to give the program's name and a brief idea of what it does.
Copyright (C) yyyy name of author

    This program is free software; you can redistribute it and/or modify
    it under the terms of the GNU General Public License as published by
    the Free Software Foundation; either version 2 of the License, or
    (at your option) any later version.

    This program is distributed in the hope that it will be useful,
    but WITHOUT ANY WARRANTY; without even the implied warranty of
    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
    GNU General Public License for more details.

    You should have received a copy of the GNU General Public License
    along with this program; if not, write to the Free Software
    Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA  02111-1307  USA
```

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

```
Gnomovision version 69, Copyright (C) year name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it
under certain conditions; type `show c' for details.
```

The hypothetical commands `show w' and `show c' should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than `show w' and `show c'; they could even be mouse–clicks or menu items––whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

```
Yoyodyne, Inc., hereby disclaims all copyright interest in the program
`Gnomovision' (which makes passes at compilers) written by James Hacker.

signature of Ty Coon, 1 April 1989
Ty Coon, President of Vice
```

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Library General Public License instead of this License.