Ingres II HOWTO

Table of Contents

Table of Contents

Ingres II HOWTO

Pal Domokos, pal.domokos@usa.net

v1.01, 23 December 1999

This document helps install the Ingres II Relational Database Management System on Linux.

Next Previous Contents Next Previous Contents

1. Introduction

1.1 Version History

- v1.0 – 7 November 1999 – Pal Domokos – Original version.
- v1.01 – 23 December 1999 – Pal Domokos – Minor fixes.

1.2 Copyright

1.3 Aim of the HOWTO

This HOWTO aims to help install Ingres II on (Intel) Linux. As always, help is useful for those who need it and can utilize it as well.

If you are an Ingres pro familiar with Linux then you don't really need to read this HOWTO. Skim through it though if you have time.

If you have no previous background in relational database management (experience with at least one real RDBMS, not some dBase−like file management system), you don't know UNIX and have just started using Linux, this HOWTO won't make an easy reading for you. Even then, I don't want to persuade you *not* to install Ingres. Don't give up easy!

If you are not a novice in database management and have some working knowledge of Linux, this HOWTO is for you! We are not going to discuss the basics of relational database management or SQL, neither are we going to elaborate on how to edit text files in Linux. You can find as much information as you want on these topics in numerous places. This HOWTO is not an Ingres guide, either: the Ingres manuals serve that purpose.

The objective of this HOWTO is that the reader can prepare for, then implement the installation of Ingres on Linux, through simple and understandable steps. It also gives starting points for basic Ingres system administration and application development.

I can only hope that the HOWTO reaches its goal. Anyway, I ask you, the reader, to email me your feelings and criticisms concerning this document. I will try to incorporate your suggestions in the next version.

1.4 Disclaimer

To put it briefly: there is no warranty about the validity of any statement contained in this document. Read and use at your own risk.

Furthermore, I am not an employee of Computer Associates International and I have no official links with CA. Computer Associates International bears no responsibility for the contents of this HOWTO.

Next Previous ContentsNextPreviousContents

---

2. The Ingres Software Development Kit

In this section the Ingres SDK is introduced and you come to know how to get it.

2.1 University Ingres and Commercial Ingres

Let us start with an important fact: there are two different types of Ingres. The original one, which was designed and developed by a research group led by Michael Stonebraker at University of California, Berkeley, was real open source software. It was free to use and distribute, source code included. In fact, it is still free software, although its development stopped in 1989. Its last version (version 8.9) made it into some Linux distributions as well. If you are interested in it, you can download it from, say

ftp://ftp.suse.com/pub/suse/i386/current/suse/ap1/ingres.rpm

ftp://ftp.suse.com/pub/suse/i386/current/suse/ap1/ingrtool.rpm

In 1979, with the foundation of Relational Technology, Inc., the career of Commercial Ingres started. Since 1995 it has been distributed by Computer Associates. Its latest version is called Ingres II 2.0. This HOWTO deals with the installation of this version of Ingres.

2.2 The Software Development Kit

Ingres, being commercial software, is not free to use. However, CA, like other RDBMS vendors, offers a free version of it (the Software Development Kit) to everyone who is interested in getting to know Ingres. The SDK has two variants, one for Windows NT and one for Linux. These variants aren't quite the same as far as

components are concerned. Obviously, we are engaged in installing the SDK for Linux here. This contains the following elements:

- Intelligent DBMS: the database engine
- Internet Commerce Enabled (ICE): this component makes database access through the Web possible
- Enhanced Security: the tool supporting mandatory access control
- C2 Security Auditing: the possibility of C2 level auditing
- Terminal Monitors: forms−based and command line SQL interfaces
- Querying and Reporting Tools: forms−based querying, report−writing and report−running tools plus a forms editor
- Querying and Reporting Runtime: like the previous one, but without the forms editor
- Vision Pro: integrated, forms−based development environment with a code generator
- Embedded SQL Precompilers: precompilers for embedding SQL statements in 3GL applications. Supported languages are C, C++, COBOL, Fortran

The SDK does not contain the following components:

- Net: this component makes possible for Ingres utilities and user applications to have access to databases residing on different machines
- Enterprise Access: communication with different database management systems and other, non−relational data sources (used to be called Gateways)
- Star: to handle distributed databases
- Replicator: support for replication functions
- Visual DBA: GUI database management tool
- OpenRoad: object−oriented, GUI development tool

You can order a free copy of the Ingres SDK CD at

http://www.cai.com/registration/cd_ingres.htm

Remember that you are not allowed to install the SDK in a business environment. It is for evaluating Ingres and prototyping applications only.

The freshest beta version of the SDK is always at

http://www.cai.com/products/betas/ingres_linux/ingres_linux.htm

If you want to give the beta version a try, check first if it is newer than your CD. You can do it by comparing the `ReadMe` file on the CD to its counterpart on the page above. This on−line `ReadMe` contains the expiration date of the beta as well.

The full version of Ingres for Linux isn't out yet. According to CA, it is due to arrive by the end of 1999.

Let me note that the Linux version of CA's Unicenter TNG Framework also includes Ingres as its embedded database management system. For this reason, knowing Ingres may come in handy when using Unicenter, too. You can order a free Unicenter TNG Framework CD from

http://www.cai.com/registration/tng_framework_linux/index.htm: for RedHat

http://www.cai.com/registration/tng_framework_linux/suse_linux.htm: for SuSE

2.3 The SDK CD

The SDK CD contains both the Windows NT and the Linux versions of the SDK. You can find the Linux files in the following directories:

- `/doc`: the manuals in PDF format. They are also available at http://www.cai.com/products/ingres/documentation_set.htm
- `/int_lnx`: this directory contains `ingres.tar`, the tarball to be installed. You can find an almost identical version of the `ReadMe` file here, too, under the name of `readme.txt`. Don't forget to read this file! `ingres.tar` can be installed directly from the CD or you can copy it to hard disk first.

NextPreviousContentsNextPreviousContents

---

3. System Requirements

In this section you will see what hardware and software requirements must be met before you can install Ingres. The ingres user, would−be owner of the installation, makes a debut, too.

3.1 Hardware

According to the `ReadMe`, the minimal hardware capable of running Ingres is:

- 486x33 processor, Pentium recommended
- 16 Mb RAM, with 32 Mb swap space (64 Mb RAM recommended)
- 150 Mb disk space if you install the whole package (200 Mb recommended. The whole space doesn't need to be in one file system: we discuss the possibilities in section **Preparing for the Installation**)

3.2 Software

Based again partly on the `ReadMe`, the following software elements must be present to install and run Ingres:

- kernel 2.0.34 or higher
- glibc 2.07 or higher (on glibc 2.1 see subsection **Miscellaneous/Forms−Based Development Tools**)
- libcrypt.so − because of export restrictions in the US, this library isn't included in every Linux distribution. If you live outside America, you can download the library from your distribution's supplementary site on another continent.
- uncompress − certain Linux distributions (such as Caldera's OpenLinux 2.2) don't contain the ncompress package. If you don't have it, get it from another distribution.

You can find a list of supported Linux distributions and versions here:

http://support.cai.com/techbases/ingres/linuxversions.html

3.3 The ingres User and II_SYSTEM

We need an account called ingres to install and run Ingres. He will own the installed software and only he can perform system management tasks such as starting and stopping Ingres. You shouldn't do any other work as ingres.

The ingres user may belong to any group. In the following example, we will create a separate group for him which will be called ingres, too.

The verified (therefore, recommended) shell for the ingres user is `bash`. All examples in this paper apply to this shell. If you use some other shell (which is probably just as fine), take into account the differences in syntax.

The binaries, shared libraries, configuration files and other files which make up the Ingres software, will be located in a tree structure after installation. You will set the root of this tree via the shell variable II_SYSTEM (to be exact, the root will be `$II_SYSTEM/ingres`) in the environment of the ingres user.

For the whole SDK, 60−odd Mb free space is needed under `$II_SYSTEM/ingres`.

If this is the first time you install Ingres, I suggest you keep the whole installation (the Ingres software, databases, backups, temporary areas, etc.) in one place so that you can find every component easily. If you have at least 150−200 Mb free space under `$II_SYSTEM/ingres` and you don't plan to create large databases (at least, not for some time), your system will work without problems. Should you at any later time run out of space, you will always have the possibility to relocate some of your databases to another partition.

In the following, I will assume that II_SYSTEM is set to `/opt`.

Logging in as root, execute the tasks mentioned above:

```
# mkdir /opt/ingres
# chmod 755 /opt/ingres
# groupadd −g 200 ingres
# useradd −g ingres −d /opt/ingres −s /bin/bash ingres
# chown ingres:ingres /opt/ingres
# passwd ingres
```

You can choose any Group ID that hasn't been allocated yet. It is practical to use a number greater than 100.

We set the home directory of ingres to `/opt/ingres` (`$II_SYSTEM/ingres`). This is not mandatory but convenient. We also granted ownership of the directory to ingres (that is compulsory).

Finally, put the following lines in the `.profile` file of ingres (or one of the other command files that start automatically at login):

```
umask 022
export II_SYSTEM=/opt
export PATH=$II_SYSTEM/ingres/bin:$II_SYSTEM/ingres/utility:$PATH
export LD_LIBRARY_PATH=/lib:/usr/lib:$II_SYSTEM/ingres/lib
export ING_EDIT=/usr/bin/vi
if [ -n "$DISPLAY" ]
then
   export TERM_INGRES=vt100fx
else
  export TERM_INGRES=vt100f
fi
```

ING_EDIT sets the editor that can be called from Ingres utilities or application programs. Naturally, you can use any editor, not just `vi`. You must, however, specify the whole access path to the program. (If you stick to `vi`, check if it is under `/usr/bin`: it can be somewhere else in your system.) Take care: if the EDITOR shell variable is also set, it overrides the value of ING_EDIT.

Setting TERM_INGRES is necessary for the terminal to work properly. Forms−based Ingres utilities, such as the install program itself, and also applications created with traditional Ingres development tools (ABF, Vision) make heavy use of function keys. The `.profile` above sets TERM_INGRES according to the terminal type (VT100−like or X).

These settings must be included in the `.profile` of every user that will be granted access to the Ingres installation.

NextPreviousContentsNextPreviousContents

4. Preparing for the Installation

This is the longest section and so it should be: after careful planning the installation itself should be an easy task.

4.1 Ingres Environment Variables

You will use Ingres environment variables to determine where to put further elements of the Ingres installation. These variables, unlike II_SYSTEM, aren't shell variables but rather parameters of Ingres stored in a file. Some of them can be changed at any time after installation, but altering the value of others requires a whole new install. Later you will see which of them are of this "stable" nature.

During installation, you can choose between setting these variables manually or letting the installer program set them to their default values (Express Install).

In the following, we will take the Ingres environment variables one by one and see what each of them is good for. It will help if you put their planned values on paper. You can find an Installation Worksheet in the **Getting Started Guide** which you can print out and use for this purpose.

4.2 II_LOG_FILE and II_DUAL_LOG

Ingres uses an installation−wide transaction log file to record all changes made to any database (recording changes is necessary for rolling back transactions, should it be required, and also for recovering databases after a system crash). The transaction log resides in `II_LOG_FILE/ingres/log`, where II_LOG_FILE is an Ingres environment variable. The name of the log file is `ingres_log`.

Express Install creates a log file of the minimal possible size, 16 Mb. Such a log file may not be large enough even in a development system. If you have space and choose manual install (in which case you can specify the size of the log), set it to something much larger.

Both the location of the log file and its size can be changed at any time after installation. The method of doing this is described in the **System Reference Guide**.

You also have to decide if you want dual logging (mirroring the transaction log). If the log gets corrupted for any reason, Ingres stops and you have to recover your databases from backup. Therefore, in a live system, it is almost compulsory either to have some type of RAID protection of the log or to have it mirrored by Ingres. If you use dual logging, the copy of the log file can be found under `II_DUAL_LOG/ingres/log`. Its name is `dual_log`.

In a development environment, mirroring the log is not always necessary.

4.3 Database Locations

Files constituting an Ingres database are put in different directories determined by means of Ingres locations. Every location points to the root of a directory tree. A location can be used for storing different types of files (see below). Databases can also share locations.

Let us see the five location types:

- Data location: place for data files of a database. In Ingres every system table and every user table, and also every index goes in a separate file.
- Checkpoint location: checkpoint is the term Ingres uses for a database backup. By default, backups are created in the checkpoint location of the database.
- Dump location: on−line backups are possible in Ingres, that is, the database may be in use while the backup program is running. For this reason, the database may change while it is being checkpointed. Ingres, so that it can restore the database to the state it was in at the *beginning* of the backup, saves the before images of those data blocks (pages) that change during the backup process. Ingres places these pages in the dump location.
- Journal location: from time to time, Ingres writes the records of committed transactions from the log file to journal files (journaling may be set on or off at the database or even at the table level). The frequency of this activity is set as a function of the amount of information that is written to the transaction log. Journaling protect the installation against media failures: if the disk containing the database crashes, you can restore the last (just before the failure occurred) committed state of the database using a backup (checkpoint) of the database and the journals created after that checkpoint was taken. If you lose the log disk, you can restore the last committed state the database was in at the time the last journal file was refreshed. Naturally, the journal location determines where the journal

files for a database reside.
- Work location: Ingres, mainly for sorting large volumes of data, needs temporary work space on disk. It creates temporary files in the work location.

A database can have more than one data or work location (this is called extending the database). However, every database must have a primary data location. The system tables reside on the primary location, together with the control file of the database (this latter is called `aaaaaaaa.cnf`. The control file stores certain basic information about the database. You can see this information with the `infodb` command after you have completed the installation.) When creating a table or index, if you don't specify the location(s) to put it in, it will be placed in the primary data location.

If a database has more than one work location, Ingres, by default, uses all of them for each sort.

When backing up the database you don't necessarily need to use the checkpoint location of the database. The `ckpdb` command allows you to specify an arbitrary place for the backup, this way you can checkpoint a database directly to tape as well.

As I said before, every Ingres location points to a Linux directory. The opposite is not true: more than one location can point to the same directory. Let us suppose that our database, **test**, has the following locations:

- DATALOC1: data location −−> `/opt`
- CKPLOC: checkpoint location −−> `/opt`
- DMPLOC: dump location −−> `/opt`
- JRNLLOC: journal location −−> `/opt`
- WORKLOC1: work location −−> `/opt`

Every location points to the `/opt` directory. Then, elements of the database will be in these directories:

- data files: `/opt/ingres/data/default/test`
- checkpoint files: `/opt/ingres/ckp/default/test`
- dump files: `/opt/ingres/dmp/default/test`
- journal files: `/opt/ingres/jnl/default/test`
- temporary files: `/opt/ingres/work/default/test`

Let us suppose now, that we extend the database to the following locations:

- DATALOC2: data location−−> `/opt`
- DATALOC3: data location−−> `/disk2`
- WORKLOC2: work location −−> `/disk2`

The database is effectively extended to the directories:

- data files: `/disk2/ingres/data/default/test`
- temporary files: `/disk2/ingres/work/default/test`

Location DATALOC2 points to `/opt`, just like DATALOC1. Tables to be created in location DATALOC2 will go to `/opt/ingres/data/default/test`, the same directory where tables created in location DATALOC1 reside.

As a location can be used for storing different types of files, we could have created just one location for DATALOC1, CKPLOC, DMPLOC, JRNLLOC, and WORKLOC1.

You can also see from the example above why different databases can use the same location: the name of the database becomes part of the directory tree, hence files of different databases never mix.

4.4 The iidbdb database

Every Ingres installation has a master database called **iidbdb**. Ingres stores information about users, locations and user databases in this database. **iidbdb** is created by the installer.

During installation you have to set the locations for **iidbdb**. These locations are stored in the following Ingres environment variables:

- II_DATABASE: data location
- II_CHECKPOINT: checkpoint location
- II_DUMP: dump location
- II_JOURNAL: journal location
- II_WORK: work location

These variables determine the default locations for every user database as well, if you don't override them when creating those databases.

Changing the value of any of the five variables requires a complete re−install of Ingres.

Let us see them one by one.

4.5 II_DATABASE

II_DATABASE determines the data location of **iidbdb**. Its default value is $II_SYSTEM (in case of a manual install you can enter a different value for II_DATABASE, while Express Install inevitably sets it to $II_SYSTEM).

The size of **iidbdb** after the installation is somewhat more than 5 Mb. It can only grow significantly if you create hundreds of Ingres users, databases or locations.

4.6 II_CHECKPOINT

II_CHECKPOINT contains the value for the checkpoint location of **iidbdb**. By default, it is also set to $II_SYSTEM.

The size of a checkpoint is just about the same as the size of the database itself (at least until you modify the template file of the checkpoint program: it is possible, as you will see in subsection **Basic System and Database Administration/Backup and Recovery**). The installer takes the first checkpoint of **iidbdb**.

If you plan to place checkpoints of user databases under II_CHECKPOINT then you have to provide more space here.

A further factor that must be taken into account is how long you want to keep backups. When starting the checkpoint program, you can request the deletion of older backups if you don't have too much free space.

4.7 II_DUMP

II_DUMP determines the dump location of the **iidbdb** database. By default, its value equals to that of II_CHECKPOINT.

By the end of the installation process, II_DUMP will contain a very small amount of data. If you always create checkpoints off−line then you won't need much space here.

4.8 II_JOURNAL

II_JOURNAL contains the value for the journal location of the **iidbdb** database. Its default value is the same as II_CHECKPOINT's.

The first checkpoint, taken by the installer causes the first, small journal file to appear here. If you don't use different journal locations for user databases then the necessary amount of free space under II_JOURNAL depends on three factors:

- Do you want Ingres to journal at all. If you take checkpoints of your databases regularly and don't mind losing the changes you made to them since the latest checkpoint should anything nasty happen to these databases or the transaction log, you may switch off journaling. Naturally, running a live system without journaling is usually not acceptable.
- If journaling is switched on, then the growth rate of the journal area is determined by the volume of changes made to the databases. Frequent, large updates require quite a bit of space under II_JOURNAL.
- The third factor is, how long you wish to keep old journal files. If, when taking a checkpoint, you instruct `ckpdb` to delete the old checkpoints, then previous journal files will be removed as well.

4.9 II_WORK

II_WORK determines the work location of the **iidbdb** database. It also defaults to II_CHECKPOINT.

The problem of sizing the work location only arises if II_WORK serves as a work location of user databases as well. It is next to impossible to estimate the temporary disk space that will be needed; however, having the size of the largest table multiplied by three should work as a starting point.

Remember that a database can have more than one work location. If the original location turns out too small, you can always extend the database to further work locations.

4.10 Other Ingres Environment Variables

Besides the Ingres environment variables that determine locations there are a couple more of them you have to set during installation (or have Express Install set them to their default values). These are:

- II_INSTALLATION: a two−character code, identifying the installation. If you create multiple Ingres installations on the same machine, each of them must have its own unique installation code. The default value for II_INSTALLATION is II. Once set, it can't be changed.
- II_NUM_OF_PROCESSORS: number of processors in the machine. By default, it is 1. If you set it to a higher value, Ingres will use spin−locks when accessing the database cache. If you don't know exactly what spin−locks are, don't bother. The point is to set II_NUM_OF_PROCESSORS to 2 if you have a multiprocessor machine. Its value can be changed at any time later.
- II_CHARSET: this variable determines the code set of all character data stored in all databases you will create in the installation. Its default value is ISO−8859−1. Perhaps it's not surprising that changing it to a different value after installation may corrupt data stored in your existing databases. As the **iidbdb** database is created by the installer program, you shouldn't choose Express Install if ISO−8859−1 doesn't suit you.
- II_TIMEZONE_NAME: name of the time zone, by default NA−PACIFIC. During manual install you can select its value from a list of valid codes. Ingres stores all date and time values in GMT and adjusts them according to II_TIMEZONE_NAME when communicating with the client. Therefore, if you set II_TIMEZONE_NAME to a different value, you will see all date−time values in the database change. For this reason, set this variable to its final value before creating the first user database.

The (manual) installer prompts you for the value of two further parameters which aren't Ingres environment variables:

- Maximum number of concurrent users in the system: this is 32 by default. Based on this number, the installer sets the value of a number of other parameters, such as the size of the database cache. These derived parameters can later be adjusted.
- SQL−92 compatible databases: by default, Ingres databases differ from the SQL−92 standard in some ways. For example, object names not protected by single or double quotes are converted to lower case rather than upper case. You can find the other differences in the **SQL Reference Guide**.

After you have made up your mind on the values of all installation parameters, you now know whether the default values for the variables that can't be changed after installation are acceptable to you. If they are, you can choose Express Install.

NextPreviousContentsNextPreviousContents

---

5. The Installation Process

In this section, at last, the actual installation of Ingres takes place.

5.1 Starting the Installation Program

In the following I will presume that you install directly from the CD which is mounted under `/cdrom`. Log in as ingres and cd to `$II_SYSTEM/ingres` if it's not your home directory. Unpack the `install` subdirectory from the tar file and start the `ingbuild` program:

```
$ cd $II_SYSTEM/ingres
$ tar xf /cdrom/ingres.tar install
$ install/ingbuild
```

On the starting screen of `ingbuild` you have to specify the path to the tar file and select the type of install: Custom or Package.

If you select Package Install you can choose between the installation of two packages: one is the whole software, the other is the DBMS server only. In the latter case none of the development tools will be installed.

I suggest you choose Custom Install. It won't be any more difficult than Package and you will see all the elements that can be installed. Furthermore, Express Install is available with Custom Install only.

After choosing Custom Install, a table on the next screen shows all components of Ingres together with their respective sizes. Because of common parts in different components the sizes added up indicate much more necessary disk space than really required.

You can see three meta−components in the table. These are:

- Full Installation
- Ingres Intelligent DBMS (the database engine, without development tools and terminal monitors)
- Ingres Stand−alone DBMS server (the database engine, without development tools but with terminal monitors)

By default every component is set to be installed. If you want to exclude some of them, you have to write "No" into column "Install?". In this case you have to exclude all meta−components that contain these elements as well.

You had previously decided if the default values for the "stable" Ingres environment variables were acceptable for your installation. If so, you can choose Express Install here. Remember that you can alter the values of II_LOG_FILE and II_NUM_OF_PROCESSORS as well as the size of the transaction log at any later time.

If this is your first Ingres install, you have the necessary space under `$II_SYSTEM/ingres` and the "stable" parameters' default values are OK, choose Express Install.

Let us see this option first.

5.2 Express Install

In the case of Express Install the installer executes the following tasks:

- It untars the components from the tar file to the `$II_SYSTEM/ingres/install/tmp` directory.
- Checks the integrity of all components.
- Puts the components into appropriate subdirectories under `$II_SYSTEM/ingres`.
- Sets the Ingres environment variables to their default values.
- Starts Ingres.
- Creates the **iidbdb** database.
- Takes a checkpoint of **iidbdb**.
- Stops Ingres.
- Sets up those components that require this (ABF, Enhanced Security, etc).

If the installation process went OK, the program tells you that every installed component is ready to use. In the table on the screen the "Install?" column shows "Ready" for every selected component.

Ingres is installed on your machine. Jump to subsection **Completing the Initial Configuration**.

5.3 Manual Install

If you choose Install rather than Express Install, the installer untars `ingres.tar`, checks the integrity of the components and puts them into their respective directories. Then it asks you if you want to setup these parameters now.

If you decide to do the setup later, the installer stops. In the table containing the components the "Install?" column shows "Not Set Up" for every selected component. You can run `ingbuild` again at any time and choose one of the options Setup All or Setup. A component can't be used until it is set up.

Let us see what happens if you choose to set up the components.

First, you have to set up the DBMS server. On the screens to follow you will see a fair amount of explanatory text about the parameters we have covered earlier.

During the setup phase, the installer prompts you for the values of the Ingres environment variables and the other parameters:

- II_INSTALLATION
- II_DATABASE
- II_CHECKPOINT: if you set it to the same value as II_DATABASE, the program warns you of the dangers of losing a database and its backup at the same time. You have to repeat II_CHECKPOINT's value for the program to accept it.
- II_JOURNAL
- II_DUMP
- II_WORK
- II_LOG_FILE: the installer reminds you Ingres' capability of mirroring the transaction log. Naturally, it only makes sense if the mirrored log file is on a different disk than the primary log file. The

installer asks you if you want to switch off dual logging. Then you have to specify the size of the log (16 Mb by default, make it bigger if you have space as I suggested earlier). After this you have to tell the installer where to put the primary log file, and, if you haven't switched dual logging off, the dual log file (II_DUAL_LOG).
- II_NUM_OF_PROCESSORS
- II_TIMEZONE_NAME
- II_CHARSET
- Maximum number of concurrent users
- SQL−92 compatible databases

At every prompt, enter the appropriate parameter's previously decided value.

The installer will also ask you about C2 audit and Enhanced Security. Accept the default values for these.

5.4 Completing the Initial Configuration

After an Express Install (but perhaps after a manual install as well), you may want to change the value of some of the Ingres environment variables. You will see how to do this here. Stay logged in as ingres.

You can view the current values of Ingres environment variables with the command:

```
$ ingprenv
```

You can change the value of any variable as in the following example:

```
$ ingsetenv II_TIMEZONE_NAME GMT2
```

We set II_TIMEZONE_NAME to GMT2 (Greenwich Mean Time + 2 hours), the time zone Hungary is placed in. You can find all possible values for II_TIMEZONE_NAME in the file `$II_SYSTEM/ingres/files/tz.crs` (look for the lists beginning with the word VALID).

You can change the value of any other Ingres environment variable in a similar way. `ingprenv` and `ingsetenv` don't require a running Ingres server.

The **System Reference Guide** contains descriptions for every Ingres environment variable. Let me mention two of those that we haven't covered yet.

II_DATE_FORMAT determines the display format of dates. By default, its value is US which provides the format dd−mmm−yy.

It should be pointed out that the setting of this variable has no effect on the way dates are stored in the database. Ingres always stores full date values, century included. Hence, you can always change the setting of II_DATE_FORMAT without the risk of corrupting data. In order to avoid Y2K problems in your applications, you should use a date format that contains the century, such as MULTINATIONAL4 (dd/mm/yyyy) or FINLAND (yyyy−mm−dd). The latter seems especially proper under Linux :−)

Another Ingres environment variable that has a good chance to be changed from its default value is II_MONEY_FORMAT. This one is responsible for how values of money type are displayed. Just like with

dates, the value of II_MONEY_FORMAT doesn't have an impact on the storage format of money columns which is always the same.

II_MONEY_FORMAT consists of two parts: the first part tells whether the currency sign precedes the amount (L = Leading or T = Trailing), the second part describes the currency itself ($, DM, Ft, etc: this part is a string of maximum 4 characters). The two parts are separated by a colon. II_MONEY_FORMAT defaults to L:$.

Only the ingres user is allowed to use `ingsetenv`, since these Ingres environment variables apply to the whole installation. However, some Ingres environment variables (including II_DATE_FORMAT and II_MONEY_FORMAT) can be overridden in the users' shell, via Linux variables of the same name. You can check the **System Reference Guide** about which other variables fall into this category.

Be careful: Ingres doesn't prevent you from changing the value of any Ingres environment variable, including II_DATABASE, II_CHECKPOINT, II_CHARSET, etc. (the "stable" parameters as we saw earlier). However, if you change one of these, prepare for the nastiest possible consequences, the mildest one of which is that Ingres won't run.

You can find information about how to set up Ingres/ICE in section **Web Access**.

5.5 Re−installation

If you want to re−install Ingres for whatever reason, remember to do the following first:

- Backup everything you need from `$II_SYSTEM/ingres` (user databases, source code of applications stored there, etc). Also backup any other databases you want to keep that are stored in different locations. You can use the `unloaddb` utility for this purpose, see the **System Reference Guide**.
- Stop Ingres.
- Remove everything under `$II_SYSTEM/ingres`. Also remove the contents of every other location where you stored any part of any database.

Databases that aren't completely removed can cause problems when you try to re−create them.

5.6 Command Line Install

You can run `ingbuild` in a non−interactive mode as well. The easiest way to do an Express Install is to start `ingbuild` in the following way:

```
install/ingbuild −express /cdrom/int_lnx/ingres.tar
```

In this case a regular Express Install takes place without having to press another key.

5.7 The Installer's Log

No matter which type of install you choose (express or manual), you can find all of `ingbuild`'s messages in `$II_SYSTEM/ingres/files/install.log`. I suggest you check this file after an Express Install to see what happened during the installation process. On the other hand, if `ingbuild` stops with an error message, check this log also.

NextPreviousContentsNextPreviousContents

6. First Steps

After you have installed and configured Ingres, it is time to check if it works properly. Supposing you are still logged in as ingres, start the Ingres system:

```
$ ingstart
```

Create a new database:

```
$ createdb test
```

Start the command line SQL interface:

```
$ sql test
```

Create a table, insert a row into it and query the table's contents:

```
create table t1 (col1 char(10));
insert into t1 values ('abcde');
select * from t1;
\g
```

If everything went OK, you should see the following:

```
$ sql test
INGRES TERMINAL MONITOR Copyright (c) 1981, 1998 Computer Associates Intl, Inc.
Ingres Linux Version II 2.0/9808 (lnx.us5/95)libc6 login
Sun Oct  3 03:43:54 1999

continue
* create table t1 (col1 char(10));
* insert into t1 values ('abcde');
* select * from t1;
* \g
Executing . . .

(1 row)
```

```
col1

abcde

(1 row)
continue
*
```

You can leave `sql` with the command \q.

NextPreviousContentsNextPreviousContents

7. Basic System and Database Administration

In this section we outline the basic tasks of the Ingres system administrator and the Ingres database administrator. You will also see what tools Ingres provides to perform these tasks. In the following I suppose you are logged in as ingres.

7.1 Starting and Stopping Ingres

You have already seen how to start Ingres:

```
$ ingstart
```

To stop Ingres, use the command:

```
$ ingstop
```

`ingstop` only stops Ingres if the are no active user sessions. If you want to stop the system regardless of user sessions, use the form:

```
$ ingstop -force
```

Take care: forcing Ingres to stop might make your databases inconsistent.

After Ingres has stopped you can check if it released all shared memory segments and semaphores:

```
$ ipcs -a
```

If you see that ingres still has some shared memory segments or semaphores, you can release them with the utility:

```
$ ipcclean
```

7.2 New Ingres Users and Locations

Since the ingres user has unlimited power of changing and possibly destroying any element of an Ingres installation, it is highly advisable that you use this account for carrying out administrative tasks only. Create one or more new Linux users and set their environment to that of ingres.

In order for any user to have access to the Ingres installation, you have to define them as Ingres users with the `accessdb` utility.

Start `accessdb`:

```
$ accessdb
```

Select the Users option:

Create

Enter the name of the user. You don't have to modify permissions (perhaps the Set Trace Flags could be set to y to allow debugging).

Save, then End and End.

You can also use `accessdb` to create new locations, change their types or extend databases to new locations. The usage of `accessdb` is covered in the **System Reference Guide** and the **Database Administrator's Guide**.

As an alternative to `accessdb`, you can maintain users and locations by running SQL commands on **iidbdb** (create user, create location, etc.). The syntax of these commands can be found in the **SQL Reference Guide**.

7.3 Creating and Destroying Databases

In section **First Steps** you created a new database. You didn't specify any options in the command

```
$ createdb test
```

Therefore the values stored in II_DATABASE, II_CHECKPOINT, etc. became locations of the **test** database. You could have specified each location explicitly:

```
$ createdb test −d<data location> −c<checkpoint location> −j<journal location>
−b<dump location> −w<work location>
```

You can remove a database in the following manner:

```
$ destroydb test
```

Be careful, because Ingres won't prompt you before destroying the database.

7.4 Collation Sequence

The collation sequence determines which of any two character strings is less than the other. In Ingres, every database can have its own sorting order. You can specify the collation sequence when creating the database:

```
createdb test −lhun
```

If you omit the −l parameter, characters in that database will be ordered as determined by the code set of the installation.

In order to use a different collation sequence you have to create a text file. The structure of this file must obey to simple rules by which you specify the absolute or relative ordering of letters and/or strings in your language. This file must then be compiled by the `aducompile` utility for Ingres to be able to use it.

The Spanish collation sequence and the collation based on the DEC Multinational Character Set are available both in source (`spanish.dsc` and `multi.dsc`) and compiled form (`spanish` and `multi`).

You specify these collation sequences in the following way:

```
createdb test −lspanish
```

or

```
createdb test −lmulti
```

The compiled definition files for a collation sequence must be in the `$II_SYSTEM/ingres/file/collation` directory. The syntax rules of the definition files can be found in the **System Reference Guide**. It may also be useful to examine the definition files for the Spanish and the DEC Multinational collations.

7.5 Backup and Recovery

The only supported way of backing up Ingres databases is via the `ckpdb` utility. With `ckpdb`, you can back up a whole database or certain tables of it. The following command backs up the **test** database:

```
$ ckpdb test
```

Checkpoints can be taken on−line.

Restoring a database can be done with the `rollforwarddb` command:

```
$ rollforwarddb test
```

By default, `rollforwarddb`, using the latest checkpoint and all journal files created since that checkpoint, restores the database to its current state. However, you can specify a point in time to restore the database to the state it was in at that time. You can go back as far as 16 checkpoints (Ingres stores data for the last 16

checkpoints in the control file of the database).

Both `ckpdb` and `rollforwarddb` can have many parameters. You can read more about these commands in the **System Reference Guide**. Besides, you should read Michael Leo's paper on Ingres backup and recovery at:

http://www.naiua.org/papers/backup99.zip

`ckpdb` and `rollforwarddb` use a template file (`$II_SYSTEM/ingres/files/cktmpl.def`). By modifying this file, you can customize the Linux commands that do the physical backup and restore of the data files. The **Database Administrator's Guide** explains the usage of this file.

7.6 Configuring Ingres

Most Ingres parameters can be set via the `cbf` utility. This is the program by which you can specify the number of DBMS servers, the sizes of different caches and a lot of other variables. The usage of `cbf` is detailed in the **System Reference Guide**.

7.7 Monitoring Ingres

You can use the `ipm` utility to monitor a running Ingres system (Visual DBA doesn't have a Linux version). With `ipm`, you can monitor user sessions, the locking and the logging subsystems. It also contains session and server management functions.

IMA (Ingres Management Architecture) makes possible to write customized, SQL−based applications for monitoring and managing Ingres installations. You can find information on IMA in the **System Reference Guide**.

7.8 Message Files

You can find the Ingres message files under `$II_SYSTEM/ingres/files`. The most important of these is `errlog.log`. Should any problems arise during the running of Ingres, this is the file to check first.

NextPreviousContentsNextPreviousContents

8. Web Access

Ingres versions running on non−Linux platforms have the Spyglass Web Server built into them. Ingres on Linux doesn't include Spyglass but the ICE (Internet Commerce Enabled) component of Ingres can be easily configured to provide HTTP access to databases. In this section we briefly examine how to connect Ingres to Apache, the most popular web server on Linux.

8.1 `ingvalidpw`

The first prerequisite for accessing Ingres databases through a web server is the presence of the `ingvalidpw` program. This program will validate the client's identifier and password.

You have to build `ingvalidpw` as root, by running the `mkvalidpw` script. Log in as root and set the environment as that of ingres, then simply type:

```
mkvalidpw
```

8.2 Configuring Apache

Your Linux distribution probably contains a pre−configured version of the Apache Web Server. If it doesn't, you can download either its binaries or source from http://www.apache.org, or one of its mirror sites. Building, installing and configuring Apache is beyond the scope of this HOWTO, we will cover only the parameters that play a role in connecting the web server to Ingres.

First, make sure that the mod_env module of Apache is compiled into the binary. If you build Apache, put the following line into `Configuration.tmpl`:

```
AddModule modules/standard/mod_env.o
```

If you have a pre−installed Apache, run it in the following way (provided the executable's name is `httpd.apache`):

```
$ httpd.apache −l
```

If you can't see `mod_env.c` among the listed modules, rebuild Apache after you have modified `Configuration.tmpl` by adding the above mentioned line to it.

The Apache server processes must run as ingres. There are two possible ways to achieve this.

The first method is to run the main process of Apache as root but start server processes as ingres. Although this is feasible, it creates so many problems that it is simply not worth the trouble.

For this reason, I suggest you configure the whole Apache system in the environment of the ingres user. In the following example, I suppose that we will put the Apache executable and all other Apache files in the

/opt/ingres/apache directory or directories under it. The following subdirectories need to be created here:

```
cgi-bin
conf
logs
```

Copy the Apache executable to /opt/ingres/apache, and put httpd.conf, srm.conf and access.conf in /opt/ingres/apache/conf.

Set the following lines in httpd.conf:

```
Port 8000 -- must be greater than 1023
User ingres
Group ingres
ServerRoot /opt/ingres/apache
ErrorLog /opt/ingres/apache/error_log
TransferLog /opt/ingres/apache/access_log
CustomLog /opt/ingres/apache/access_log common
PidFile /opt/ingres/apache/httpd.pid
ScoreBoardFile /opt/ingres/apache/apache_status
Timeout  3000 -- or whatever you like, but the default value (300) is probably too small
(it is measured in seconds)
PassEnv II_SYSTEM
PassEnv LD_LIBRARY_PATH
```

The last two lines must be added to httpd.conf. These variables will be passed from the environment of the ingres user to the environment of CGI programs started from Apache (specifically iceinst and ice, the two executables of ICE).

Parameters to be set in srm.conf:

```
DocumentRoot /opt/ingres/apache
ScriptAlias /cgi-bin/ /opt/ingres/apache/cgi-bin/
```

DocumentRoot is the default directory for every document provided by Apache, while ScriptAlias is the directory containing the CGI programs.

To be modified in access.conf:

```
<Directory /opt/ingres/apache/cgi-bin>
AllowOverride None
Options ExecCGI
</Directory>
```

After you have edited the configuration files, start Apache. Supposing again, that the Apache executable is called httpd.apache:

```
/opt/ingres/apache/httpd.apache -f /opt/ingres/apache/conf/httpd.conf
```

8.3 ICE Setup

It is time to configure ICE and its Tutorials. You can do this with a browser and the `iceinst` program. Let us suppose that, according to the previous example, our CGI directory is `/opt/ingres/apache/cgi-bin` and Apache is listening on port 8000. Let the name of our machine be ingserv1. Then you can start `iceinst` in the following manner:

```
$ iceinst -d/opt/ingres/apache/cgi-bin -u/cgi-bin -shttp://ingserv1:8000
-b/opt/netscape/netscape
```

Option -d is the full path to the CGI directory, -u is this directory's address within the site, -s is the Internet address of the server, while -b is the full path to the browser. If you omit option -b and write -remote instead, then `iceinst` won't try to start the browser. This way you can configure ICE from another machine, directing your browser to

http://ingserv1/cgi-bin/iceinst

First the program asks for the value of II_SYSTEM. Then you should visit every screen and set all parameters presented on them. Have `iceinst` install the Dynamic SQL Tutorial and the Macro Processor Tutorial as well. These show the usage of ICE via applications and a database (**icedb** by default).

It is important to create a directory under Apache's DocumentRoot where ICE can store the output it creates for clients' requests. ICE won't start until you create this directory and specify its name in `iceinst`.

After you have finished with every form, choose the Install and/or Uninstall Selected Components option. If you have set everything properly, the configuration of ICE and the installation of the tutorials take place. ICE is ready to use.

NextPreviousContents Next PreviousContents

9. Miscellaneous Topics

Further hints to the use of Ingres.

9.1 Automatic Startup and Shutdown

If you want Ingres to start automatically whenever Linux boots and stop when you shutdown or reboot the system, do the following:

Log in as root.

Check if your Linux variant has System V or BSD style `init` (`init`'s `man` page will tell that).

If your system conforms to System V, the `/etc/rc.d/init.d` directory must exist. Create a file there (call it `ingres` or any other name you wish). The file should contain at least the following:

```
#!/bin/sh

case "$1" in
 start)
  echo "Starting Ingres"
  su − ingres −c "ingstart"
  ;;

 stop)
  echo "Stopping Ingres"
  su − ingres −c "ingstop"
  ;;

 *)
  echo "Usage: ingres {start|stop}"
  exit 1
  ;;
esac

exit 0
```

Link the file as `K01ingres` to these directories:

`/etc/rc.d/rc0.d`

`/etc/rc.d/rc1.d`

`/etc/rc.d/rc2.d`

`/etc/rc.d/rc6.d`

Also link it as `S99ingres` to these directories:

`/etc/rc.d/rc3.d`

`/etc/rc.d/rc4.d`

`/etc/rc.d/rc5.d`

It is not important to call its links `K01ingres` and `S99ingres`, the point is that the name starting with K should contain a small number (so that Ingres stops early when changing to a lower runlevel) and the name starting with S should contain a large number (so that Ingres starts after everything else has started). Naturally, the file names must not clash with names of existing files.

If you have a BSD style `init`, put the following lines into `/etc/rc.d/rc.local`:

```
echo "Starting Ingres"
su − ingres −c "ingstart"
```

This will start Ingres. (As a matter of fact, you can use `/etc/rc.d/rc.local` even if you have a System V style `init`.)

To stop Ingres automatically, create a file in `/etc/shutdown.d` (call it, say, `ingres`) that contains the commands

```
echo "Stopping Ingres"
su − ingres −c "ingstop"
```

No matter which type your system is, the files you create must be executable files, owned by root.

Naturally, if your system provides a utility for configuring programs to start and stop automatically (such as `chkconfig` in RedHat), use that if you wish.

9.2 `ingmenu`

The easiest way to access an Ingres database (at least, for beginners) is via the `ingmenu` program. You can reach Ingres' forms−based utilities, by which you can create, update and query tables, create, edit and run reports and ABF or Vision applications, from `ingmenu`. Its usage is:

```
ingmenu test
```

Test is the name of the database.

9.3 Circumventing Ingres Net

As the SDK doesn't include Ingres Net, in theory it is not possible for Ingres applications to access databases on different machines. However, there exists a method, not supported by CA, by which you can come around this problem.

Let us suppose your application runs on host ingdev and the database (called **test**) you would like to update or query resides on host ingserv. Your first task is to find out the port number of the appropriate DBMS server running on ingserv. You can use `ipm` for this purpose: start `ipm` on ingserv and choose option Server List. In the list of servers select one that is of type INGRES and handles the **test** database (you have to see either test or ALL in column Connecting to Databases). You find the port number of the DBMS server in the first column. Let us suppose it is 1259.

On machine ingdev, set the shell variable II_DBMS_SERVER in the following way:

```
$ export II_DBMS_SERVER='ingserv::1259'
```

Run the command:

```
$ sql test
```

If it works, you have access to the **test** database on host ingserv.

This solution is applicable only if both machines are of the same architecture, the same operating system is running on both of them, the character set is the same in both Ingres installations and so on: I don't know the

full list of necessary conditions. Therefore, I can't guarantee that this trick will work.

On the other hand, if you restart Ingres on host ingserv, the DBMS server process will get a different TCP/IP port, therefore you probably have to automate fetching the current port number to the application server. You can use the `show` command of the `iinamu` utility for this purpose. The following line gives the port number of the DBMS server if there is only one server running:

```
$ echo show | iinamu | grep INGRES | tr -s ' ' '\t' | cut -f4
```

9.4 Forms-Based Development Tools

The installation includes a sample application, created by ABF, the traditional development tool of Ingres. You can load it with the `abfdemo` command. Unfortunately, the manuals of ABF and Vision can't be found either on the CD or on the CA site.

An even greater problem is that under glibc 2.1 applications created by ABF or Vision can't be either compiled or run directly from the database. This problem will be solved in the next Ingres version. In the meantime, you can download the RedHat glibc 2.0 compatibility packages from the following URL's:

- ftp://ftp.redhat.com/pub/redhat/redhat-6.0/i386/RedHat/RPMS/compat-binutils-5.2-2.9.1.0.23.1.i386.rpm
- ftp://ftp.redhat.com/pub/redhat/redhat-6.0/i386/RedHat/RPMS/compat-egcs-5.2-1.0.3a.1.i386.rpm
- ftp://ftp.redhat.com/pub/redhat/redhat-6.0/i386/RedHat/RPMS/compat-egcs-c++-5.2-1.0.3a.1.i386.rpm
- ftp://ftp.redhat.com/pub/redhat/redhat-6.0/i386/RedHat/RPMS/compat-glibc-5.2-2.0.7.1.i386.rpm
- ftp://ftp.redhat.com/pub/redhat/redhat-6.0/i386/RedHat/RPMS/compat-libs-5.2-1.i386.rpm

By default, these packages go to `/usr/i386-glibc20-linux`. Change the first line of `$II_SYSTEM/ingres/files/abflnk.opt` to:

```
-L/usr/i386-glibc20-linux/lib -L/lib -L/usr/lib -L/usr/local/lib -L$II_SYSTEM/ingres/lib
```

In the users' shell change LD_LIBRARY_PATH:

```
LD_LIBRARY_PATH=/lib:/usr/lib:$II_SYSTEM/ingres/lib:/usr/i386-glibc20-linux/lib
```

The compatibility packages work perfectly in Caldera OpenLinux 2.2 as well. I haven't tested them in other distributions, though.

9.5 Ingperl and Perl DBI

Previous Perl versions, version 4 included, made Ingres access possible via libraries known as ingperl. You can find information on ingperl at

http://www.contrib.andrew.cmu.edu/~lfm/ingperl.html

In Perl 5 a new, unified database interface, called Perl DBI, appeared. Its site is

http://www.symbolstone.org/technology/perl/DBI/index.html

You can download the Ingres module of DBI from that site.

9.6 Ingres links

I leave you with a few pointers to important Ingres sites:

- http://www.cai.com/ingres/: home page of the Ingres RDBMS on the CA site
- http://support.cai.com/ingressupp.html: Ingres Technical Support
- http://www.cai.com/ingres/inquire/: inquire_ingres: Ingres magazine
- http://www.naiua.org/faqs.html: Ingres FAQ's
- http://www.naiua.org/papers/: Ingres papers
- news:comp.databases.ingres: the Ingres newsgroup

Have fun!

Next PreviousContents