Firebird 2 Quick Start Guide

IBPhoenix Editors
Firebird Project members

8 April 2008, document version 3.7 — covers Firebird 2.0–2.0.4 and 2.1

Table of Contents

About this guide

The *Firebird Quick Start Guide* is an introduction for the complete newcomer to a few essentials for getting off to a quick start with a Firebird binary kit. The guide first saw the light as Chapter 1 of the *Using Firebird* manual, sold on CD by IBPhoenix. Later it was published separately on the Internet. In June 2004, IBPhoenix donated it to the Firebird Project. Since then it is maintained, and regularly updated, by members of the Firebird documentation project.

Important

Before you read on, verify that this guide matches your Firebird version. This guide covers versions 2.0–2.0.4 and 2.1. For all other Firebird versions, get the corresponding Quick Start Guide at http://www.firebirdsql.org/?op=doc.

Some warnings before you start

- Firebird 2.0.2 was recalled due to a regression; if you use it, upgrade to 2.0.3 or higher ASAP and make sure to read your new version's Release Notes.
- If you want to rely on Linux forced writes to work correctly, upgrade to at least 2.0.4.

What is in the kit?

All of the kits contain all of the components needed to install the Firebird server:

- The Firebird server executable.
- One or more client libraries.
- The command-line tools.
- The standard user-defined function libraries.
- A sample database.
- The C header files (not needed by beginners).
- Release notes – ESSENTIAL READING!

Classic or Superserver?

Firebird comes in two flavours, called *architectures*: Classic Server and Superserver. Which one should you install? That depends on your situation. A short overview of the most important differences follows.

Table 1. Firebird 2 Classic Server vs. Superserver

	Classic Server	Superserver
Processes	Creates a separate process for every client connection, each with its own cache. Less resource use if the number of connections is low.	A single process serves all connections, using threads to handle requests. Shared cache space. More efficient if the number of simultaneous connections grows.
Local connections	Permits fast, direct I/O to database files for local connections on Linux. The client process must have filesystem-level access rights to the database for this to work.	On Linux, all local connections are made via the network layer, using `localhost` (often implicitly). Only the server process needs access rights to the database file.
	On Windows, both architectures now support safe and reliable local connections, with only the server process requiring access rights to the database file.	
Multiprocessor	SMP (symmetrical multi-processor) support. Better performance in case of a small number of connections that do not influence each other.	No SMP support. On multi-processor Windows machines, performance can even drop dramatically as the OS switches the process between CPUs. To prevent this, set the *`CpuAffinityMask`* parameter in the configuration file `firebird.conf`.
Guardian	When run as a Windows *application* (as opposed to a service) you can't use the Firebird Guardian. Note that running Firebird as an application is the only option on Windows 9x–ME.	Can be used with the Guardian on Windows, whether run as an application or as a service.

As you can see, neither of the architectures is better in all respects. This is hardly surprising: we wouldn't maintain two separate architectures if one of them was an all-fronts loser.

If you're still not sure what to choose (maybe you find all this tech talk a little overwhelming), use this rule of thumb:

- On Windows, choose Superserver.
- On Linux, just pick one or the other. In most circumstances, chances are that you won't notice a performance difference.

Note that you can always switch to the other architecture later; your applications and databases will keep functioning like before.

For Linux, Superserver download packages start with `FirebirdSS`, Classic packages with `FirebirdCS`. For Windows, there is a combined installation package; you choose the architecture during the installation process.

Embedded Server for Windows

On Windows platforms only, Firebird offers a third flavor: *Embedded Server*, a client and server rolled into one DLL for ease of deployment. While very practical, it lacks most of Firebird's usual security features. For more information on Firebird Embedded Server, consult the *Clients and Servers* chapter in *Using Firebird*:

http://www.firebirdsql.org/manual/ufb-cs-embedded.html (HTML)

http://www.firebirdsql.org/pdfmanual/Using-Firebird_(wip).pdf (PDF)

The Embedded Server comes in a separate download package.

Default disk locations

Linux

The following table shows the default component locations of a Firebird installation on Linux. Some of the locations may be different on other Unix-like systems.

Table 2. Firebird 2 component locations on Linux

Component	**File Name**	**Default Location**
Installation directory (referred to hereafter as `<InstallDir>`)	—	`/opt/firebird`
Release Notes and other documentation	Various files	`<InstallDir>/doc`
Firebird server	`fbserver` (SS) or `fb_inet_server` (CS)	`<InstallDir>/bin`
Command-line tools	`isql`, `gbak`, `nbackup`, `gsec`, `gfix`, `gstat`, etc.	`<InstallDir>/bin`
Sample database	`employee.fdb`	`<InstallDir>/examples/empbuild`
UDF libraries	`ib_udf.so`, `fbudf.so`	`<InstallDir>/UDF`
Additional server-side libraries	`libicu*.so`, `libib_util.so`	`<InstallDir>/bin`
Client libraries	`libfbclient.so.2.`*`m.n`* (network client) `libfbembed.so.2.`*`m.n`* (local client with embedded engine, Classic only) The usual symlinks (`*.so.2`, `*.so`) are created. Legacy `libgds.*` symlinks are also installed.	`/usr/lib` (actually, the real stuff is in `<InstallDir>/lib`, but you should use the links in `/usr/lib`)

Windows

In the table below, `<ProgramDir>` refers to the Windows programs folder. This is usually "`C:\Program Files`" but may also be a different path, e.g. "`D:\Programmi`". Likewise, `<SystemDir>` refers to the Windows system directory. Be sure to read the notes below the table, especially if you're running Firebird on a 64-bit Windows system.

Table 3. Firebird 2 component locations on Windows

Component	File Name	Default Location
Installation directory (referred to hereafter as `<InstallDir>`)	—	`<ProgramDir>\Firebird\Firebird_2_0`
Release Notes and other documentation	Various files	`<InstallDir>\doc`
Firebird server	`fbserver.exe` (SS) or `fb_inet_server.exe` (CS)	`<InstallDir>\bin`
Command-line tools	`isql.exe`, `gbak.exe`, `nbackup.exe`, `gsec.exe`, `gfix.exe`, `gstat.exe`, etc.	`<InstallDir>\bin`
Sample database	`employee.fdb`	`<InstallDir>\examples\empbuild`
User-defined function (UDF) libraries	`ib_udf.dll`, `fbudf.dll`	`<InstallDir>\UDF`
Additional server-side libraries	`icu*.dll`, `ib_util.dll`	`<InstallDir>\bin`
Client libraries	`fbclient.dll` (with an optional `gds32.dll`, to support legacy apps)	`<InstallDir>\bin` (with an optional copy in `<SystemDir>` – see note below table)

The Windows system directory

The exact path to the Windows System directory depends on your Windows version. Typical locations on 32-bit systems are:

- for Windows 95/98/ME: `C:\Windows\System`
- for Windows NT/2000: `C:\WINNT\System32`
- for Windows XP: `C:\Windows\System32`

For 64-bit systems, read the next note.

Important notice for 64-bit Windows users

On 64-bit Windows systems, the “`Program Files`” directory is reserved for 64-bit programs. If you try to install a 32-bit application into that folder, it will be auto-redirected to a directory which – in English versions – is called “`Program Files (x86)`”. In other language versions the name may be different.

In the same vein, the `System32` directory is reserved for 64-bit libraries. 32-bit libraries go into `SysWOW64`. That's right: 64-bit libraries are in `System32`, 32-bit libraries in `SysWOW64`.

If you're not aware of this, you may have a hard time locating your 32-bit Firebird components on a 64-bit Windows system.

(Incidentally, WOW stands for *Windows on Windows*. Now you can also work out what LOL means.)

Installing Firebird

The instructions given below for the installation of Firebird on Windows and Linux should be sufficient for the vast majority of cases. However, if you experience problems or if you have special needs not covered here, be sure to read the *INSTALLATION NOTES* chapter in the Release Notes. This is especially important if you are upgrading from a previous version or if there are remnants of an old (and maybe long gone) InterBase or Firebird installation floating around your system (DLLs, Registry entries, environment variables...)

Installing the Firebird server

Installation drives

Firebird server – and any databases you create or connect to – must reside on a hard drive that is physically connected to the host machine. You cannot locate components of the server, or any database, on a mapped drive, a filesystem share or a network filesystem.

Note

You can mount a read-only database on a CD-ROM drive but you cannot run Firebird server from one.

Installation script or program

Although it is possible to install Firebird by a filesystem copying method – such as “untarring” a snapshot build or decompressing a structured `.zip` archive – it is strongly recommended that you use the distributed release kit (`.exe` for Windows, `.rpm` for Linux), especially if this is the first time you install Firebird. The Windows installation executable, the Linux rpm program and the `install.sh` script in the official `.tar.gz` for various Posix platforms all perform some essential setup tasks. Provided you follow the installation instructions correctly, there should be nothing for you to do upon completion but log in and go!

Installing on Windows

The Firebird installer lets you choose between Superserver and Classic Server installation. Both are fully mature and stable and there is no reason to categorically prefer one to the other. Of course you may have your own specific considerations.

If you install Firebird under Windows 95/98/ME, **un**check the option to install the Control Panel applet. It doesn't work on these platforms. You'll find a link to a usable applet further down. (Note: the option to install the applet is only available for Superserver.)

On Windows server platforms – NT, 2000, 2003 and XP – Firebird will run as a system service by default, but during the installation you can also choose to let it run as an application. Non-server Windows systems – 95, 98 and ME – don't support services; running as an application is the only option there.

Use the Guardian?

The Firebird Guardian is a utility that monitors the server process and tries to restart it if it terminates abnormally. The Guardian does not work with Firebird Classic Server on Windows if run as an application. This is due to a known bug, which will be fixed later. Currently the Firebird 2 installer doesn't give you the option to include the Guardian *at all* with a Classic Server, even if you install it as a service.

The Guardian works correctly with Superserver, whether run as an application or as a service.

If you run Firebird *as a service* on Windows 2000, 2003 or XP, the Guardian is a convenience rather than a necessity, since these operating systems have the facility to watch and restart services. It is recommended that you keep the Guardian option on (if possible) in all other situations.

> **Warning**
>
> If you install Firebird 2.0.3 (and probably earlier 2.0 versions too) on Windows *without* the Guardian, the installer doesn't correctly detect an already running server. This leads to errors when it tries to overwrite existing DLLs and executables. So, in the above case, make sure to uninstall any existing Firebid server before attempting to install the new one. This bug has been fixed in versions 2.0.4 and 2.1.

Installing on Linux and other Unix-like platforms

In all cases, read the Release Notes that came with your Firebird package (chapter *Installation Notes*, section *Posix Platforms*). There may be significant variations from release to release of any Posix operating system, especially the open source ones. Where possible, the build engineers for each Firebird version have attempted to document any known issues.

If you have a Linux distribution that supports rpm installs, consult the appropriate platform documentation for instructions about using RPM Package Manager. In most distributions you will have the choice of performing the install from a command shell or through a GUI interface.

For Linux distributions that cannot process rpm programs, and for Unix flavours for which no `.rpm` kit is provided, use the `.tar.gz` kit. You will find detailed instructions in the Release Notes.

Shell scripts have been provided. In some cases, the Release Notes may instruct you to edit the scripts and make some manual adjustments.

Installing multiple servers

Firebird 2 allows the operation of multiple servers on a single machine. It can also run concurrently with Firebird 1.x or InterBase servers. Setting this up is not a beginner's task though. If you need to run multiple servers, consult the *Installation Notes* chapter of the Release Notes, and have the Firebird 1.5 Release Notes handy too – you will be directed to them at a certain point during your reading of the Installation Notes.

Testing your installation

If everything works as designed, the Firebird server process will be running on your server machine upon completion of the installation. It will also start up automatically whenever you restart your computer.

Before testing the Firebird server itself, it is advisable to verify if the server machine is reachable from the client at all. At this point, it is assumed that you will use the recommended TCP/IP network protocol for your Firebird client/server connections.

Notes

- If you have installed a Classic Server on Linux/Unix or any Firebird server on Windows, it is possible to connect directly to the local server, without using a network layer. If you intend to use Firebird for this type of connection **only**, you can skip the “Pinging the server” section below.
- For information about using the NetBEUI protocol in an all-Windows environment, refer to the *Network Configuration* chapter in the *Using Firebird* manual sold by IBPhoenix, or consult the InterBase 6 Operations Guide (http://www.ibphoenix.com/downloads/60OpGuide.zip).
- Firebird does not support IPX/SPX networks.

Pinging the server

The **ping** command – available on most systems – is a quick and easy way to see if you can connect to a server machine via the network. For example, if your server's IP address in the domain that is visible to your client is `192.13.14.1`, go to a command shell on the client machine and type the command

ping 192.13.14.1

substituting this example IP address with the IP address that your server is broadcasting. If you are on a managed network and you don't know the server's IP address, ask your system administrator. Of course you can also ping the server by its name, if you know it:

ping vercingetorix

If you are connecting to the server from a local client – that is, a client running on the same machine as the server – you can ping the virtual TCP/IP loopback server:

ping localhost –or– **ping 127.0.0.1**

If you have a simple network of two machines linked by a crossover cable, you can set up your server with any IP address you like except 127.0.0.1 (which is reserved for a local loopback server) and, of course, the IP address which you are using for your client machine. If you know the “native” IP addresses of your network cards, and they are different, you can simply use those.

Once you have verified that the server machine is reachable from the client, you can go on to the next step.

Checking that the Firebird server is running

After installation, Firebird server should be running:

On Linux or other Unix-like systems:
As a service.

On Windows server systems (NT, 2000, 2003, XP):
As a service or as an application. Service is default and highly recommended.

On Windows non-server systems (95, 98, ME):
As an application.

The following sections show you how to test the server in each of these situations.

Server check: Linux and other Unices

Use the **top** command in a command shell to inspect the running processes interactively. If a Firebird Superserver is running, you should see a process named `fbguard`. This is the Guardian process. Further, there will be one main and zero or more child processes named `fbserver`.

The following screen shows the output of top, restricted by grep to show only lines containing the characters `fb`:

```
frodo:/inkomend/firebird # top -b -n1 | grep fb
 2587 firebird  24   0  1232 1232 1028 S  0.0  0.3   0:00.00 fbguard
 2588 firebird  15   0  4124 4120 2092 S  0.0  0.9   0:00.04 fbserver
 2589 firebird  15   0  4124 4120 2092 S  0.0  0.9   0:00.00 fbserver
 2604 firebird  15   0  4124 4120 2092 S  0.0  0.9   0:00.00 fbserver
 2605 firebird  15   0  4124 4120 2092 S  0.0  0.9   0:00.02 fbserver
 2606 firebird  15   0  4124 4120 2092 S  0.0  0.9   0:00.00 fbserver
 2607 firebird  15   0  4124 4120 2092 S  0.0  0.9   0:00.00 fbserver
```

As an alternative to top, you can use **ps -ax** or **ps -aux** and pipe the output to grep.

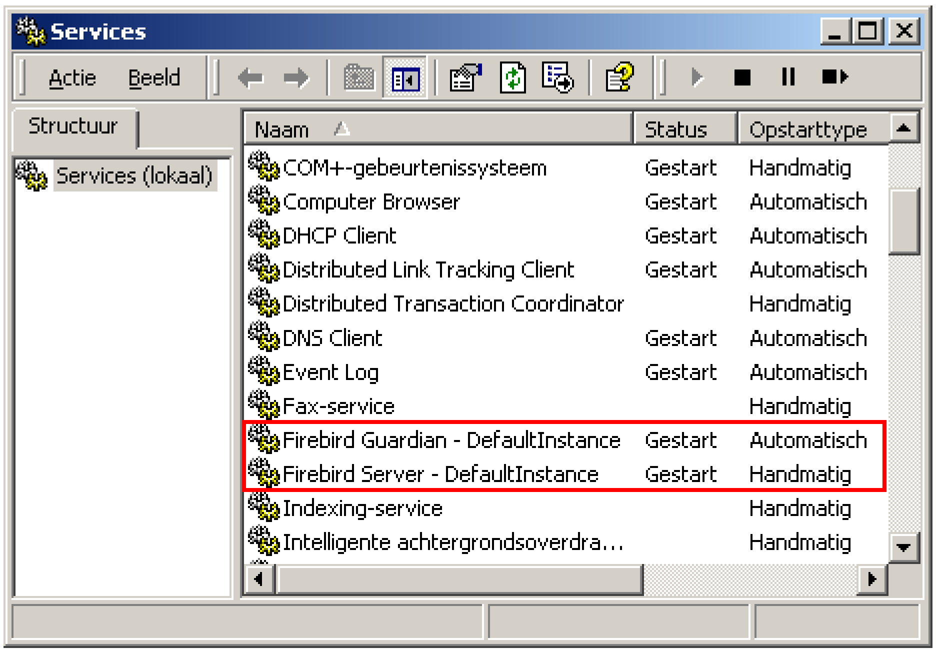
For Classic Server versions, the process name is `fb_inet_server`. There will be one instance of this process running for each network connection. Note that if there are no active connections, or if there are only direct local connections, you won't find `fb_inet_server` in the process list. `fb_lock_mgr` should be present though as soon as any kind of Classic connection has been established.

Other ways to test a Firebird server immediately after installation include connecting to a database, creating a database, and launching the gsec utility. All these operations are described later on in this guide.

Server check: Windows, running as service

Open Control Panel -> Services (NT) or Control Panel -> Administrative Tools -> Services (2000, XP).

This illustration shows the Services applet display on Windows 2000. The appearance may vary from one Windows server edition to another. Also, service names may vary with the Firebird version.



Naam	Status	Opstarttype
COM+-gebeurtenissysteem	Gestart	Handmatig
Computer Browser	Gestart	Automatisch
DHCP Client	Gestart	Automatisch
Distributed Link Tracking Client	Gestart	Automatisch
Distributed Transaction Coordinator		Handmatig
DNS Client	Gestart	Automatisch
Event Log	Gestart	Automatisch
Fax-service		Handmatig
Firebird Guardian - DefaultInstance	Gestart	Automatisch
Firebird Server - DefaultInstance	Gestart	Handmatig
Indexing-service		Handmatig
Intelligente achtergrondsoverdra...		Handmatig

You should at least find the Firebird server in the services listing. The Guardian may or may not be running, depending on the choices you made during installation.

Server check: Windows, running as application

If Firebird is up and running as an application, it is represented by an icon in the system tray:

- A green and grey server symbol if controlled by the Guardian;
- A round yellow and black graphic if running standalone.

A flashing icon indicates that the server is in the process of starting up (or at least trying to do so). A red icon, or an icon with an overlying red stop sign, indicates that startup has failed.

One way to make 100% sure if the server is running or not is to press Ctrl-Alt-Del and look for the `fbserver` or `fb_inet_server` process (and possibly `fbguard`) in the task list.

On some occasions, you may need to start the Guardian or server once explicitly via the Start menu even if you opted for “Start Firebird now” at the end of the installation process. Sometimes a reboot is necessary.

If you're desperately trying to start Firebird and nothing seems to work, ask yourself if you've installed Firebird 2 Classic server with the Guardian option enabled (the installation program doesn't offer this possibility anymore, but there are other ways). As said before, the combination Classic + Guardian currently doesn't work if Firebird runs as an application. Uninstall Firebird if necessary and reinstall Classic *without* Guardian, or Superserver with or without Guardian.

You can shut the server down via the menu that appears if you right-click on the tray icon. Notice that this also makes the icon disappear; you can restart Firebird via the Start menu.

Note

Windows Classic Server launches a new process for every connection, so the number of `fb_inet_server` processes will always equal the number of client connections plus one. Shutdown via the tray icon menu only terminates the first process (the *listener*). Other processes, if present, will continue to function normally, each terminating when the client disconnects from the database. Of course, once the listener has been shut down, new connections can't be made.

In the case of Superserver you can also use a Control Panel applet to check and alter the Firebird server status. Some available applets will be presented a little later in this guide.

Performing a client-only install

Each remote client machine needs to have the client library – `libfbclient.so` on Posix clients, `fbclient.dll` on Windows clients – that matches the release version of the Firebird server.

Firebird versions from 1.5 onward can install symlinks or copies named after the 1.0 libs (with the “old” InterBase names), to maintain compatibility with third-party products which need these files.

Some extra pieces are also needed for the client-only install.

Windows

At present, no separate installation program is available to install only the client pieces on a Windows machine. If you are in the common situation of running Windows clients to a Linux or other Unix-like Firebird server (or another Windows machine), you need to download the full Windows installation kit that corresponds to the version of Firebird server you install on your server machine.

Fortunately, once you have the kit, the Windows client-only install is easy to do. Start up the installation program just as though you were going to install the server, but select one of the client-only options from the installation menu.

Linux and some other Posix clients

A small-footprint client install program for Linux clients is not available either. Additionally, some Posix flavours – even within the Linux constellation – have somewhat idiosyncratic requirements for filesystem locations. For these reasons, not all *x distributions for Firebird even contain a client-only install option.

For most Linux flavours, the following procedure is suggested for a Firebird client-only install. Log in as `root` for this.

1. Look for `libfbclient.so.2.`*`m.n`* (*`m.n`* being the minor plus patch version number) in `/opt/firebird/lib` on the machine where the Firebird server is installed. Copy it to `/usr/lib` on the client.

2. Create chained symlinks using the following commands:

   ln -s /usr/lib/libfbclient.so.2.*m.n* /usr/lib/libfbclient.so.2

   ln -s /usr/lib/libfbclient.so.2 /usr/lib/libfbclient.so

   ...replacing *`2.m.n`* with your version number, e.g. `2.0.0` or `2.1.0`

   If you're running applications that expect the legacy libraries to be present, also create the following symlinks:

   ln -s /usr/lib/libfbclient.so /usr/lib/libgds.so.0

   ln -s /usr/lib/libfbclient.so /usr/lib/libgds.so

3. Copy `firebird.msg` to the client machine, preferably into the `/opt/firebird` directory. If you place it somewhere else, create a system-wide permanent `FIREBIRD` environment variable pointing to the right directory, so that the API routines can locate the messages.

4. Optionally copy some of the Firebird command-line tools – e.g. isql – to the client machine. *Note:* always copy the tools from a Superserver kit, regardless of the architecture of the server(s) you're planning to connect to. Tools from Classic distributions terminate immediately if they can't find the `libfbembed` library (which is useless for network connections) upon program start.

Instead of copying the files from a server, you can also pull them out of a Firebird `tar.gz` kit. Everything you need is located in the `/opt/firebird` tree within the `buildroot.tar.gz` archive that's packed inside the kit.

Server configuration and management

There are several things you should be aware of – and take care of – before you start using your freshly installed Firebird server. This part of the manual introduces you to some useful tools and shows you how to protect your server and databases.

User management: gsec

Firebird comes with a command-line user management tool called *gsec*. Although its functions can also be performed by a number of third-party GUI utilities, you should at least have a basic knowledge of gsec, since this is the official tool and it's present in every Firebird server installation. In the next sections you will use gsec to execute two tasks: changing the SYSDBA password and adding a Firebird user. First though, some points of attention:

Permission to run gsec

With some Firebird installations, you can only run gsec if you are logged into the operating system as Superuser (`root` on Linux) or as the user the Firebird server process runs under. On Windows server platforms, you typically need to be in the Power User group or higher to run gsec successfully.

Trouble running gsec

If you have enough privileges but invoking gsec results in a message like “`cannot attach to password database - unable to open database`”:

- You may be running Firebird on Windows and for some reason the local protocol isn't working. One rather common cause for this is running Windows Vista, 2003 or XP with terminal services enabled. To enable the local protocol, open `firebird.conf`, uncomment the *`IpcName`* parameter and set it to `Global\FIREBIRD`. Then restart the server.

  > **Note**
  >
  > In Firebird 2.0.1 and up, `Global\FIREBIRD` is already the default on TS-enabled Windows systems.

- If the above doesn't apply to you, you can at least circumvent the problem by “tricking” gsec into using TCP/IP. Add the following parameter to the command line, adjusting the path if necessary:

```
-database "localhost:C:\Program Files\Firebird\Firebird_2_0\security2.fdb"
```

The file `security2.fdb` is the *security database*, where Firebird keeps its user account details. It is located in your Firebird installation directory.

- Maybe your security database is a renamed `security.fdb` from Firebird 1.5. Of course this can't be the case immediately after installation. Someone (you?) must have put it there, in order to keep the existing accounts available. Consult the Release Notes for instructions on how to upgrade old security databases.

If the error message starts with "`Cannot attach to services manager`", the server may not be running at all. In that case, go back to *Testing your installation* and fix the problem.

Calling gsec on Linux

On **nix systems, if you call gsec from its own directory, you should type **./gsec** instead of just **gsec**. The current directory is usually not part of the search path, so plain **gsec** may either fail or launch a "wrong" gsec.

Changing the SYSDBA password

One Firebird account is created automatically as part of the installation process: SYSDBA. This account has all the privileges on the server and cannot be deleted. Depending on version, OS, and architecture, the installation program will either

- install the SYSDBA user with the password `masterkey` (actually, `masterke`: characters after the eighth are ignored), or
- ask you to enter a password during installation, or
- generate a random password and store that in the file `SYSDBA.password` within your Firebird installation directory.

If the password is `masterkey` and your server is exposed to the Internet *at all* – or even to a local network, unless you trust every user with the SYSDBA password – you should change it immediately using the gsec command-line utility. Go to a command shell, *cd* to the Firebird `bin` subdirectory and issue the following command to change the password to (as an example) `icuryy4me`:

```
gsec -user sysdba -pass masterkey -mo sysdba -pw icuryy4me
```

Notice that you specify "sysdba" twice in the command:

- With the *`-user`* parameter you identify yourself as SYSDBA. You also provide SYSDBA's current password in the *`-pass`* parameter.
- The *`-mo[dify]`* parameter tells gsec that you want to modify an account – which happens to be SYSDBA again. Lastly, *`-pw`* specifies the type of modification: the password.

If all has gone well, the new password `icuryy4me` is now encrypted and stored, and `masterkey` is no longer valid. Please be aware that unlike Firebird user names, passwords are case-sensitive.

Adding Firebird user accounts

Firebird allows the creation of many different user accounts. Each of them can own databases and also have various types of access to databases and database objects it doesn't own.

Using gsec, you can add a user account as follows from the command line in the Firebird `bin` subdirectory:

```
gsec -user sysdba -pass masterkey -add billyboy -pw sekrit66
```

Provided that you've supplied the correct password for SYSDBA, a user account called `billyboy` will now have been created with password `sekrit66`. Remember that passwords are case-sensitive.

> **Note**
>
> Since Firebird 2, users can change their own passwords. Previous versions required SYSDBA to do this.

Security

Firebird 2 offers a number of security options, designed to make unauthorised access as difficult as possible. Be warned however that some configurable security features default to the old, “insecure” behaviour inherited from InterBase and Firebird 1.0, in order not to break existing applications.

It pays to familiarise yourself with Firebird's security-related configuration parameters. You can significantly enhance your system's security if you raise the protection level wherever possible. This is not only a matter of setting parameters, by the way: other measures involve tuning filesystem access permissions, an intelligent user accounts policy, etc.

Below are some guidelines for protecting your Firebird server and databases.

Run Firebird as non-system user
: On Unix-like systems, Firebird already runs as user `firebird` by default, not as `root`. On Windows server platforms, you can also run the Firebird service under a designated user account (e.g. `Firebird`). The default practice – running the service as the `LocalSystem` user – poses a security risk if your system is connected to the Internet. Consult `README.instsvc` in the `doc` subdir to learn more about this.

Change SYSDBA's password
: As discussed before, if your Firebird server is reachable from the network and the system password is `masterkey`, change it.

Don't create user databases as SYSDBA
: SYSDBA is a very powerful account, with full (destructive) access rights to all your Firebird databases. Its password should be known to a few trusted database administrators only. Therefore, you shouldn't use this super-account to create and populate regular databases. Instead, generate normal user accounts, and provide their account names and passwords to your users as needed. You can do this with gsec as shown above, or with any third-party Firebird administration tool.

Protect databases on the filesystem level
: Anybody who has filesystem-level read access to a database file can copy it, install it on a system under his or her own control, and extract all data from it – including possibly sensitive information. Anybody who has filesystem-level write access to a database file can corrupt it or totally destroy it.

: As a rule, only the Firebird server process should have access to the database files. Users don't need, and should not have, access to the files – not even read-only. They query databases via the server, and the server makes sure that users only get the allowed type of access (if at all) to any objects within the database.

Disable Classic local mode on Linux

An exception to the above rule is the so-called local or embedded access mode of Firebird Classic Server on Linux. This mode *requires* that users have proper access rights to the database file itself. They must also have read access to the security database `security2.fdb`. If this worries you, reserve filesystem access to the security database (and other databases, while you're at it) for the server process only. Users are then obliged to connect via the network layer. However, the `libfbembed.*` libraries should not be removed from your system, because the Firebird command-line tools refuse to run if they are not present.

(Another exception is the Windows Embedded Server, but that's outside the scope of this manual.)

Use database aliases

Database aliases shield the client from physical database locations. Using aliases, a client can e.g. connect to “`frodo:zappa`” without having to know that the real location is `frodo:/var/firebird/music/underground/mothers_of_invention.fdb`. Aliases also allow you to relocate databases while the clients keep using their existing connection strings.

Aliases are listed in the file `aliases.conf`, in this format on Windows machines:

```
poker = E:\Games\Data\PokerBase.fdb
blackjack.fdb = C:\Firebird\Databases\cardgames\blkjk_2.fdb
```

And on Linux:

```
books = /home/bookworm/database/books.fdb
zappa = /var/firebird/music/underground/mothers_of_invention.fdb
```

Giving the alias an `.fdb` (or any other) extension is fully optional. Of course if you do include it, you must also specify it when you use the alias to connect to the database.

Restrict database access

The *`DatabaseAccess`* parameter in `firebird.conf` can be set to `Restrict` to limit access to explicitly listed filesystem trees, or even to `None` to allow access to aliased databases only. Default is `All`, i.e. no restrictions.

Note that this is not the same thing as the filesystem-level access protection discussed earlier: when *`DatabaseAccess`* is anything other than `All`, the server will refuse to open any databases outside the defined scope even if it has sufficient rights on the database files.

There are more security parameters, but the ones not mentioned here are already set to an adequate protection level by default. You can read about them in the 1.5 and 2.0 Release Notes and in the comments in `firebird.conf` itself.

Windows Control Panel applets

Several control panel applets are available for use with Firebird. Whilst such applets are not essential, they do provide a convenient way to start and stop the server and check its current status.

Firebird Server Manager

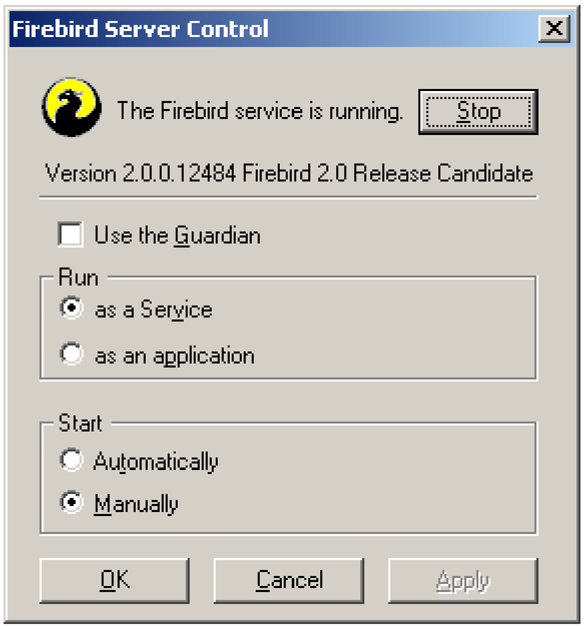
The Firebird Server Manager applet is included in the Firebird distribution. The option to install this applet is only available for Superserver.

Note

The applet is also usable for Classic server, provided that it (the server, that is) runs as a service, not as an application. Since the installation dialogue won't give you the option to include the applet with a Classic server, you must, if you really want it:

- Install Superserver first;
- Copy the applet `Firebird2Control.cpl` from the Windows system folder to a safe place;
- Uninstall Superserver;
- Install Classic;
- Copy the applet back to the system directory.

This is a screenshot of the activated applet. Notice that the title bar says “Firebird Server Control”, although it is listed in the Control Panel as *Firebird 2.0 Server Manager*.


Firebird Server Control

The Firebird service is running.

Version 2.0.0.12484 Firebird 2.0 Release Candidate

☐ Use the Guardian

Run: [✓ as a Service] as an application

Start: Automatically [✓ Manually]

Unfortunately, the bundled applet only works on Windows NT, 2000/2003 and XP.

Firebird Control Center

If you want an applet that also works on Windows 9x or ME, visit this webpage:

> http://www.achim-kalwa.de/fbcc.phtml

...and download the Firebird Control Center fbcc-0.2.7.exe. Please note that, unlike the applet included with Firebird, the Firebird Control Center will *not* work with Classic servers at all.

The Control Center doesn't look anything like the Firebird applet shown in the screenshot, but offers the same functionality, and then some. Attention: if you run Firebird *as a service* and *without the Guardian*, the Start/ Stop button will be labeled “Start” all the time, even when the server is already running. It functions as it should though. In all other configurations the button will say “Start” or “Stop” according to the situation.

Administration tools

The Firebird kit does not come with a GUI admin tool. It does have a set of command-line tools – executable programs which are located in the `bin` subdirectory of your Firebird installation. One of them, gsec, has already been introduced to you.

The range of excellent GUI tools available for use with a Windows client machine is too numerous to describe here. A few GUI tools written in Borland Kylix, for use on Linux client machines, are also in various stages of completion.

Inspect the Downloads > Contributed > Administration Tools page at http://www.ibphoenix.com for all of the options.

> **Note**
>
> Remember: you can use a Windows client to access a Linux server and vice-versa.

Working with databases

In this part of the manual you will learn:

- how to connect to an existing database,
- how to create a database,
- and some things you should know about Firebird SQL.

In as much as remote connections are involved, we will use the recommended TCP/IP protocol.

Connection strings

If you want to connect to a database or create one you have to supply, amongst other things, a *connection string* to the client application (or, if you are a programmer, to the routines you are calling). A connection string uniquely identifies the location of the database on your computer, local network, or even the Internet.

Local connection strings

An explicit local connection string consists of the path + filename specification in the native format of the filesystem used on the server machine, for example

- on a Linux or other Unix-like server:

```
/opt/firebird/examples/empbuild/employee.fdb
```

- on a Windows server:

  ```
  C:\Biology\Data\Primates\Apes\populations.fdb
  ```

Many clients also allow relative path strings (e.g. "`..\examples\empbuild\employee.fdb`") but you should use them with caution, as it's not always obvious how they will be expanded. Getting an error message is annoying enough, but applying changes to another database than you thought you were connected to may be disastrous.

Instead of a file path, the local connection string may also be a *database alias* that is defined in `aliases.conf`, as mentioned earlier. The format of the alias depends only on how it's defined in the aliases file, not on the server filesystem. Examples are:

- `zappa`
- `blackjack.fdb`
- `poker`

> **Tip**
>
> If your local connections fail, it may be because the local protocol isn't working properly on your machine. If you're running Windows Vista, 2003 or XP with terminal services enabled, this can often be fixed by setting *`IpcName`* to `Global\FIREBIRD` in the configuration file `firebird.conf` (don't forget to uncomment the parameter and restart the server). In Firebird 2.0.1, `Global\FIREBIRD` is already the default on TS-enabled Windows systems.
>
> If setting *`IpcName`* doesn't help and you don't get the local protocol enabled, you can always work around the problem by putting "`localhost:`" before your database paths or aliases, thus turning them into TCP/IP connection strings (discussed below).

TCP/IP connection strings

A TCP/IP connection string consists of:

1. a server name or IP address
2. a colon ("`:`")
3. either the absolute path + filename on the server machine, or an alias defined on the server machine.

Examples:

- On Linux/Unix:

  ```
  pongo:/opt/firebird/examples/empbuild/employee.fdb

  bongo:fury

  112.179.0.1:/var/Firebird/databases/butterflies.fdb

  localhost:blackjack.fdb
  ```

- On Windows:

  ```
  siamang:C:\Biology\Data\Primates\Apes\populations.fdb
  ```

```
sofa:D:\Misc\Friends\Rich\Lenders.fdb

127.0.0.1:Borrowers
```

Notice how the aliased connection strings don't give any clue about the server OS. And they don't have to, either: you talk to a Linux Firebird server just like you talk to a Windows Firebird server. In fact, specifying an explicit database path is one of the rare occasions where you have to be aware of the difference.

Third-party programs

Please note that some third-party client programs may have different requirements for the composition of connection strings. Refer to their documentation or online help to find out.

Connecting to an existing database

A sample database named `employee.fdb` is located in the `examples/empbuild` subdirectory of your Firebird installation. You can use this database to “try your wings”.

If you move or copy the sample database, be sure to place it on a hard disk that is physically attached to your server machine. Shares, mapped drives or (on Unix) mounted SMB (Samba) filesystems will not work. The same rule applies to any databases that you create or use.

Connecting to a Firebird database requires the user to authenticate with a user name and a valid password. In order to work with objects inside the database – such as tables, views, etc. – you also need explicit permissions on those objects, unless you own them (you own an object if you have created it) or if you're connected as SYSDBA. In the example database `employee.fdb`, sufficient permissions have been granted to PUBLIC (i.e. anybody who cares to connect) to enable you to view and modify data to your heart's content.

For simplicity here, we will look at authenticating as SYSDBA using the password `masterkey`. Also, to keep the lines in the examples from running off the right edge, we will work with local databases and use relative paths. Of course everything you'll learn in these sections can also be applied to remote databases, simply by supplying a full TCP/IP connection string.

Connecting with isql

Firebird ships with a text-mode client named *isql* (Interactive SQL utility). You can use it in several ways to connect to a database. One of them, shown below, is to start it in interactive mode. Go to the `bin` subdirectory of your Firebird installation and type **isql** (Windows) or **./isql** (Linux) at the command prompt.

[In the following examples, # means “hit **Enter**”]

```
C:\Program Files\Firebird\Firebird_2_0\bin>isql#
Use CONNECT or CREATE DATABASE to specify a database
SQL>CONNECT ..\examples\empbuild\employee.fdb user SYSDBA password masterkey;#
```

Important

- In isql, every SQL statement must end with a semicolon. If you hit **Enter** and the line doesn't end with a semicolon, isql assumes that the statement continues on the next line and the prompt will change from `SQL>` to `CON>`. This enables you to split long statements over multiple lines. If you hit **Enter** after your statement and you've forgotten the semicolon, just type it after the `CON>` prompt on the next line and press **Enter** again.
- If you run Classic Server on Linux, a fast, direct local connection is attempted if the database path does not start with a hostname. This may fail if your Linux login doesn't have sufficient access rights to the database file. In that case, connect to `localhost:<path>`. Then the server process (with Firebird 2 usually running as user `firebird`) will open the file. On the other hand, network-style connections may fail if a user created the database in Classic local mode and the server doesn't have enough access rights.

Note

You can optionally enclose the path, the user name and/or the password in single (`'`) or double (`"`) quotes. If the path contains spaces, quoting is mandatory.

At this point, isql will inform you that you are connected:

```
Database:  ..\examples\empbuild\employee.fdb, User: sysdba
SQL>
```

You can now continue to play about with the `employee.fdb` database. With isql you can query data, get information about the metadata, create database objects, run data definition scripts and much more.

To get back to the command prompt, type:

```
SQL>QUIT;#
```

You can also type `EXIT` instead of `QUIT`, the difference being that `EXIT` will first commit any open transactions, making your modifications permanent.

Connecting with a GUI client

GUI client tools usually take charge of composing the CONNECT string for you, using server, path (or alias), user name and password information that you type into prompting fields. Use the elements as described in the preceding topic.

Notes

- It is quite common for such tools to expect the entire server + path/alias as a single connection string – just like isql does.
- Remember that file names and commands on Linux and other “Unix-ish” platforms are case-sensitive.

Creating a database using isql

There is more than one way to create a database with isql. Here, we will look at one simple way to create a database interactively – although, for your serious database definition work, you should create and maintain your metadata objects using data definition scripts.

Starting isql

To create a database interactively using the isql command shell, get to a command prompt in Firebird's `bin` subdirectory and type **isql** (Windows) or **./isql** (Linux):

```
C:\Program Files\Firebird\Firebird_2_0\bin>isql#
Use CONNECT or CREATE DATABASE to specify a database
```

The CREATE DATABASE statement

Now you can create your new database interactively. Let's suppose that you want to create a database named `test.fdb` and store it in a directory named `data` on your `D` drive:

```
SQL>CREATE DATABASE 'D:\data\test.fdb' page_size 8192#
CON>user 'SYSDBA' password 'masterkey';#
```

Important

- In the CREATE DATABASE statement it is *mandatory* to place quote characters (single or double) around path, username and password. This is different from the CONNECT statement.
- If you run Classic Server on Linux and you don't start the database path with a hostname, creation of the database file is attempted with your Linux login as the owner. This may or may not be what you want (think of access rights if you want others to be able to connect). If you prepend `localhost:` to the path, the server process (with Firebird 2 usually running as user `firebird`) will create and own the file.

The database will be created and, after a few moments, the SQL prompt will reappear. You are now connected to the new database and can proceed to create some test objects in it.

But to verify that there really is a database there, let's first type in this query:

```
SQL>SELECT * FROM RDB$RELATIONS;#
```

Although you haven't created any tables yet, the screen will fill up with a large amount of data! This query selects all of the rows in the system table RDB$RELATIONS, where Firebird stores the metadata for tables. An “empty” database is not really empty: it contains a number of system tables and other objects. The system tables will grow as you add more user objects to your database.

To get back to the command prompt type `QUIT` or `EXIT`, as explained in the section on connecting.

Firebird SQL

Every database management system has its own idiosyncrasies in the ways it implements SQL. Firebird adheres to the SQL standard more rigorously than most other RDBMSes. Developers migrating from products that are less standards-compliant often wrongly suppose that Firebird is quirky, whereas many of its apparent quirks are not quirky at all.

Division of an integer by an integer

Firebird accords with the SQL standard by truncating the result (quotient) of an integer/integer calculation to the next lower integer. This can have bizarre results unless you are aware of it.

For example, this calculation is correct in SQL:

```
1 / 3 = 0
```

If you are upgrading from an RDBMS which resolves integer/integer division to a float quotient, you will need to alter any affected expressions to use a float or scaled numeric type for either dividend, divisor, or both.

For example, the calculation above could be modified thus in order to produce a non-zero result:

```
1.000 / 3 = 0.333
```

Things to know about strings

String delimiter symbol

Strings in Firebird are delimited by a pair of single quote (apostrophe) symbols: `'I am a string'` (ASCII code 39, *not* 96). If you used earlier versions of Firebird's relative, InterBase®, you might recall that double and single quotes were interchangeable as string delimiters. Double quotes cannot be used as string delimiters in Firebird SQL statements.

Apostrophes in strings

If you need to use an apostrophe inside a Firebird string, you can “escape” the apostrophe character by preceding it with another apostrophe.

For example, this string will give an error:

```
'Joe's Emporium'
```

because the parser encounters the apostrophe and interprets the string as `'Joe'` followed by some unknown keywords. To make it a legal string, double the apostrophe character:

```
'Joe''s Emporium'
```

Notice that this is TWO single quotes, not one double-quote.

Concatenation of strings

The concatenation symbol in SQL is two “pipe” symbols (ASCII 124, in a pair with no space between). In SQL, the “+” symbol is an arithmetic operator and it will cause an error if you attempt to use it for concatenating strings. The following expression prefixes a character column value with the string “`Reported by: `”:

```
    'Reported by: ' || LastName
```

Firebird will raise an error if the result of a string concatenation exceeds the maximum (var)char size of 32 Kb. If only the *potential* result – based on variable or field size – is too long you'll get a warning, but the operation will be completed successfully. (In pre-2.0 Firebird, this too would cause an error and halt execution.)

See also the section below, Expressions involving NULL, about concatenating in expressions involving `NULL`.

Double-quoted identifiers

Before the SQL-92 standard, it was not legal to have object names (identifiers) in a database that duplicated keywords in the language, were case-sensitive or contained spaces. SQL-92 introduced a single new standard to make any of them legal, provided that the identifiers were defined within pairs of double-quote symbols (ASCII 34) and were always referred to using double-quote delimiters.

The purpose of this “gift” was to make it easier to migrate metadata from non-standard RDBMSes to standards-compliant ones. The down-side is that, if you choose to define an identifier in double quotes, its case-sensitivity and the enforced double-quoting will remain mandatory.

Firebird does permit a slight relaxation under a very limited set of conditions. If the identifier which was defined in double-quotes:

1. was defined as all upper-case,
2. is not a keyword, and
3. does not contain any spaces,

...then it can be used in SQL unquoted and case-insensitively. (But as soon as you put double-quotes around it, you must match the case again!)

Warning

Don't get too smart with this! For instance, if you have tables "TESTTABLE" and "TestTable", both defined within double-quotes, and you issue the command:

```
SQL>select * from TestTable;
```

...you will get the records from "TESTTABLE", not "TestTable"!

Unless you have a compelling reason to define quoted identifiers, it is usually recommended that you avoid them. Firebird happily accepts a mix of quoted and unquoted identifiers – so there is no problem including that keyword which you inherited from a legacy database, if you need to.

Warning

Some database admin tools enforce double-quoting of *all* identifiers by default. Try to choose a tool which makes double-quoting optional.

Expressions involving `NULL`

In SQL, `NULL` is not a value. It is a condition, or *state*, of a data item, in which its value is unknown. Because it is unknown, `NULL` cannot behave like a value. When you try to perform arithmetic on `NULL`, or involve it with values in other expressions, the result of the operation will almost always be `NULL`. It is not zero or blank or an “empty string” and it does not behave like any of these values.

Below are some examples of the types of surprises you will get if you try to perform calculations and comparisons with `NULL`.

The following expressions all return `NULL`:

- `1 + 2 + 3 + NULL`
- `not (NULL)`
- `'Home ' || 'sweet ' || NULL`

You might have expected 6 from the first expression and “`Home sweet `” from the third, but as we just said, `NULL` is not like the number 0 or an empty string – it's far more destructive!

The following expression:

- `FirstName || ' ' || LastName`

will return `NULL` if either `FirstName` or `LastName` is `NULL`. Otherwise it will nicely concatenate the two names with a space in between – even if any one of the variables is an empty string.

Tip

Think of `NULL` as `UNKNOWN` and these strange results suddenly start to make sense! If the value of `Number` is unknown, the outcome of '`1 + 2 + 3 + Number`' is also unknown (and therefore `NULL`). If the content of `MyString` is unknown, then so is '`MyString || YourString`' (even if `YourString` is non-`NULL`). Etcetera.

Now let's examine some PSQL (Procedural SQL) examples with `if`-constructs:

-
  ```
  if (a = b) then
    MyVariable = 'Equal';
  else
    MyVariable = 'Not equal';
  ```

  After executing this code, `MyVariable` will be `'Not equal'` if both `a` and `b` are `NULL`. The reason is that `'a = b'` yields `NULL` if at least one of them is `NULL`. If the test expression of an “`if`” statement is `NULL`, it behaves like `false`: the '`then`' block is skipped, and the '`else`' block executed.

  Warning

  Although the expression may *behave* like `false` in this case, it's still `NULL`. If you try to invert it using `not()`, what you get is another `NULL` – not “`true`”.

-
  ```
  if (a <> b) then
    MyVariable = 'Not equal';
  else
    MyVariable = 'Equal';
  ```

  Here, `MyVariable` will be `'Equal'` if `a` is `NULL` and `b` isn't, or vice versa. The explanation is analogous to that of the previous example.

The DISTINCT keyword comes to the rescue!

Firebird 2 implements a new use of the DISTINCT keyword allowing you to perform (in)equality tests that take NULL into account. The semantics are as follows:

- Two expressions are DISTINCT if they have different values or if one is NULL and the other isn't;
- They are NOT DISTINCT if they have the same value or if both are NULL.

Notice that if neither operand is NULL, DISTINCT works exactly like the “<>” operator, and NOT DISTINCT like the “=” operator.

DISTINCT and NOT DISTINCT always return true or false, never NULL.

Using DISTINCT, you can rewrite the first PSQL example as follows:

```
if (a is not distinct from b) then
  MyVariable = 'Equal';
else
  MyVariable = 'Not equal';
```

And the second as:

```
if (a is distinct from b) then
  MyVariable = 'Not equal';
else
  MyVariable = 'Equal';
```

These versions will give you the results that a normal human being (untouched by SQL standards) would expect, whether there are NULLs involved or not.

More about NULLs

A lot more information about NULL behaviour can be found in the *Firebird Null Guide*, at these locations:

http://www.firebirdsql.org/manual/nullguide.html (HTML)

http://www.firebirdsql.org/pdfmanual/Firebird-Null-Guide.pdf (PDF)

An updated and greatly extended version of the Null Guide is available since January 2007.

Preventing data loss

Backup

Firebird comes with two utilities for backing up and restoring your databases: *gbak* and *nbackup*. Both can be found in the bin subdirectory of your Firebird installation. Firebird databases can be backed up whilst users are connected to the system and going about their normal work. The backup will be taken from a snapshot of the database at the time the backup began.

Regular backups and occasional restores should be a scheduled part of your database management activity.

Warning

Except in nbackup's lock mode, do not use external proprietary backup utilities or file-copying tools such as WinZip, tar, copy, xcopy, etc., on a database which is running. Not only will the backup be unreliable, but the disk-level blocking used by these tools can corrupt a running database.

Important

Study the warnings in the next section about database activity during restores!

More information about gbak can be found in *The Firebird Book*, the *Using Firebird* guide (a not-so-recent version is available through IBPhoenix, an updated version is currently in a state of growth on the Firebird site), or in the InterBase 6.0 manuals combined with the Firebird 1.5 and 2.0 Release Notes. See the links to these resources in How to get help.

The nbackup manual is here (HTML and PDF version, same content):

http://www.firebirdsql.org/manual/nbackup.html

http://www.firebirdsql.org/pdfmanual/Firebird-nbackup.pdf

How to corrupt a database

The following sections constitute a summary of things *not* to do if you want to keep your Firebird databases in good health.

Modifying metadata tables yourself

Firebird stores and maintains all of the metadata for its own and your user-defined objects in special tables, called *system tables*, right in the database itself. The identifiers for these system tables, their columns and several other types of system objects begin with the characters `RDB$`.

Because these are ordinary database objects, they can be queried and manipulated just like your user-defined objects. However, just because you can does not say you should. The Firebird engine implements a high-level subset of SQL (DDL) for the purpose of defining and operating on metadata objects, typically through CREATE, ALTER and DROP statements.

It cannot be recommended too strongly that you use DDL – not direct SQL operations on the system tables – whenever you need to alter or remove metadata. Defer the “hot fix” stuff until your skills in SQL and your knowledge of the Firebird engine become very advanced. A wrecked database is neither pretty to behold nor cheap to repair.

Disabling forced writes

Firebird is installed with forced writes (synchronous writes) enabled by default. Changed and new data are written to disk immediately upon posting.

It is possible to configure a database to use asynchronous data writes – whereby modified or new data are held in the memory cache for periodic flushing to disk by the operating system's I/O subsystem. The common term for this configuration is *forced writes off* (or *disabled*). It is sometimes resorted to in order to improve performance during large batch operations.

Disabling forced writes on Windows

The big warning here is: do *not* disable forced writes on a Windows server. It has been observed that the Windows server platforms do not flush the write cache until the Firebird service is shut down. Apart from power interruptions, there is just too much that can go wrong on a Windows server. If it should hang, the I/O system goes out of reach and your users' work will be lost in the process of rebooting.

Note

Windows 9x and ME do not support deferred data writes

Disabling forced writes on Linux

Linux servers are safer for running an operation with forced writes disabled temporarily. Still, do not leave it disabled once your large batch task is completed, unless you have a very robust fall-back power system.

Warning

It was recently discovered that **forced writes did not work at all** under Linux. This is due to a bug in the `fcntl()` function on Linux and it affects all Firebird versions up to and including 2.0.3. The only known workaround is to mount the partition in question with the `sync` option — or upgrade to Firebird 2.0.4 or higher.

Other Unices don't seem to suffer from this bug. To make sure, test if you system's `fcntl()` can successfully set the `O_SYNC` flag. Set the flag on and off and *read it back both times* to make sure the change was actually written.

Restoring a backup to a running database

One of the restore options in the gbak utility (`gbak -rep[lace_database]`) allows you to restore a gbak file over the top of an existing database. It is possible for this style of restore to proceed without warning while users are logged in to the database. Database corruption is almost certain to be the result.

Note

Notice that the shortest form of this command is `gbak -rep`, not `gbak -r` as it used to be in previous Firebird versions. What happened to `gbak -r`? It is now short for `gbak -recreate_database`, which functions the same as `gbak -c[reate]` and throws an error if the specified database already exists. You can force overwriting of the existing database by adding the `o[verwrite]` flag though. This flag is only supported with `gbak -r`, not with `gbak -c`.

These changes have been made because many users thought that the `-r` switch meant *restore* instead of replace – and only found out otherwise when it was too late.

Warning

Be aware that you will need to design your admin tools and procedures to prevent any possibility for any user (including SYSDBA) to restore to your active database if any users are logged in.

If is practicable to do so, it is recommended to restore to spare disk space using the `gbak -c[reate]` option and test the restored database using isql or your preferred admin tool. If the restored database is good, shut down the server. Make a filesystem copy of the old database and then copy the restored database file (or files) over their existing counterparts.

Allowing users to log in during a restore

If you do not block access to users while performing a restore using `gbak -rep[lace_database]` then users may be able to log in and attempt to do operations on data. Corrupted structures will result.

How to get help

The community of willing helpers around Firebird goes a long way back, to many years before the source code for its ancestor, InterBase® 6, was made open source. Collectively, the Firebird community does have all the answers! It even includes some people who have been involved with it since it was a design on a drawing board in a bathroom in Boston.

- Visit the official Firebird Project site at http://www.firebirdsql.org and join the user support lists, in particular `firebird-support`. Look at http://www.firebirdsql.org/?op=lists for instructions.
- Use the Firebird documentation index at http://www.firebirdsql.org/?op=doc.
- Visit the Firebird knowledge site at http://www.ibphoenix.com to look up a vast collection of information about developing with and using Firebird. IBPhoenix also sells a Developer CD with the Firebird binaries and lots of documentation.
- Order the official Firebird Book at http://www.ibphoenix.com/main.nfs?a=ibphoenix&s=1093098777:149734&page=ibp_firebird_book, for more than 1100 pages jam-packed with Firebird information.
- As a last resort – since our documentation is still incomplete – you can consult the InterBase 6.0 beta manuals (the files whose names start with `60` at http://www.ibphoenix.com/downloads/) in combination with the Firebird 1.5 and 2.0 Release Notes.

> **Note**
>
> The IBPhoenix publications *Using Firebird* and *The Firebird Reference Guide*, though still on the Developer CD, are no longer actively maintained. However, most of the material contained in those documents is currently being brought up to date and added, bit by bit, to the official project documentation.

The Firebird Project

The developers, designers and testers who gave you Firebird and several of the drivers are members of the Firebird open source project at SourceForge, that amazing virtual community that is home to thousands of open source software teams. The Firebird project's address there is http://sourceforge.net/projects/firebird. At that site are the source code tree, the download packages and a number of technical files related to the development and testing of the codebases.

The Firebird Project developers and testers use an email list forum – `firebird-devel@lists.sourceforge.net` – as their “virtual laboratory” for communicating with one another about their work on enhancements, bug-fixing and producing new versions of Firebird.

Anyone who is interested in watching their progress can join this forum. However, user support questions are a distraction which they do not welcome. Please do not try to post your user support questions there! These belong in the `firebird-support` group.

Happy Firebirding!

Appendix A: Document History

The exact file history is recorded in the `manual` module in our CVS tree; see http://sourceforge.net/cvs/?group_id=9028

Revision History

0.0	2002	IBP	Published as Chapter One of *Using Firebird*.
1.0	2003	IBP	Published separately as a free Quick Start Guide.
1.x	June 2004	IBP	Donated to Firebird Project by IBPhoenix.
2.0	27 Aug 2004	PV	Upgraded to Firebird 1.5 Added Classic vs. Superserver section. Reorganised and corrected Disk Locations Table. Added (new) screenshots. Added section on security. Updated and completed information on Control Panel applets. Added more examples to “Expressions involving `NULL`”. Various other corrections and additions.
2.1	20 Feb 2005	PV	Enhanced GSEC section. Added more info to CONNECT and CREATE DATABASE sections. Added version number and document history.
2.1.1	1 Mar 2005	PV	Changed `gbak r[estore]` to `r[eplace]` in two places.
2.1.2	8 Apr 2005	PV	Reordered Firebird SQL subsections. Added links to Firebird Null Guide.
2.2	2 Dec 2005	PV	Removed "Using the books by IBPhoenix" as it doesn't make sense in the QSG. Promoted "How to get help" to 1st-level section and removed "Where to next" shell. Removed link to UFB and RefGuide; added a note instead explaining their current status. Updated/corrected classic-super comparison table. Moved a number of sections on installing, working with databases, and (un)safety into newly created top-level sections.
2.2.1	22 Dec 2005	PV	Corrected statement on SS thread usage in Classic-vs-Superserver table. Fixed broken link.
3.0	21 May 2006	PV	Creation of 2.0 Quick Start Guide, still equal to previous revision except for some version numbers, XML ids etc.
3.2	10 Aug 2006	PV	Promoted “Firebird Project members” to co-authors in articleinfo.

			Updated references to website (`firebird.sourceforge.net` -> `www.firebirdsql.org`). Removed “maturity” and “Service Manager” rows from Classic-vs-Super table; these things are no longer different in Firebird 2. Also changed the row on local connections: CS and SS now both allow safe, reliable local connections on Windows. Added row on Guardian. Prepended a column with feature names. Removed any and all remarks about Classic not having a (full) Service Manager. Removed 2nd paragraph of “Default disk locations” section. Removed notes stating that Classic/Win connections will fail without a host name. Updated location table and inserted rows for documentation. Edited the Installation sections; added sections on Guardian and installing multiple servers. Removed “if-you-do-not-find-the-release-notes” tip. Heavily edited and extended the “Testing your installation” sections. The “Other things you need” section is now gone and its contents distributed across other sections. Added a section on gsec (consisting partly of existing material). Greatly enhanced and extended the *Security* section, and moved it to another location. Extended and improved the “Windows Control Panel applets” section. Edited “Working with databases”. Added a special section on connection strings. Added information on access to database objects, the `EXIT` statement, and local vs. remote connections. Made some paths in the examples relative, to keep the lines short. Extended paragraph on metadata. Weakened the claim that Firebird is more SQL-compliant than any other RDBMS. Changed the “Expressions involving `NULL`” section. Added a subsection on DISTINCT. Changed “More about `NULL`s” subsection somewhat. Renamed “Safety measures” to “Preventing data loss”. The Security subsection has been moved elsewhere. Extended *Backup* section to include nbackup information. Added links to other documentation. In the “How to corrupt...” part, changed gbak -r syntax to -rep and added explanatory note. Added the “IB6 plus rlsnotes” as last-resort option to *How to get help*. Also mentioned firebird-support explicitly. Corrected more version numbers, paths, and stuff. Many sections have been reshuffled, moved up or down the hierarchy, etc. Many smaller modifications are not listed here. Added “Happy Firebirding!” to conclude the last section.
3.3	15 Oct 2006	PV	Default disk locations table: added isql to command line tools; added row for additional server-side libs. Added introductory paragraph to “Installing Firebird”. Changed first sentence of “Installing on Linux...” Changed and extended “Server check: Linux and other Unices”. Corrected and extended the section on Linux client-only installs.

			Security section: moved last paragraph of the “Protect databases...” listitem into a new item on Classic local mode. Connection strings: improved and extended introductory paragraph; added a subsection on third party program requirements. Changed 3rd and 4th paragraph of “Connecting to an existing database”. Used relative paths in connection examples. Updated/corrected note on the use of quote characters. Edited first “Important” item in “The CREATE DATABASE statement”. Updated the warning about concatenation of long strings. Extended the note in “Restoring a backup to a running database”. Updated last sentence of first paragraph in “The Firebird Project”.
3.4	25 Jan 2007	PV	*About this guide*: Changed note about versions and replaced HTML and PDF links with single link to new doc index page. *Classic or Superserver?*: Replaced note on Embedded Server with a proper subsection, containing more info and links to UFB. *Default disk locations*: Created two subsections (for Linux and Windows); also split table in two and removed first column. Introduced placeholders `<ProgramDir>` and `<SystemDir>`. Changed text around tables, changed existing note, and added note for Win64 users. *Security*: Removed statement that 1.5 Release Notes are included with 2.x packages. *More about `NULL`s*: Replaced note about the Null Guide being updated with a para announcing the availability of the new version. *Backup*: Updated information on UFB. *How to get help*: Updated documentation links and changed text here and there.
3.5	14 Mar 2007	PV	*About this guide* and *Important notice for 64-bit Windows users*: Minor rewordings. *User management: gsec* and *Connection strings*: Added information on enabling local protocol with *`IpcName=Global\FIREBIRD`*. *Security :: Use database aliases*: Changed type from `<database>` to `<literal>` to improve output.
3.6	21 Sep 2007	PV	*About this guide*: Mentioned 2.0.3. Warned against 2.0.2. *Expressions involving `NULL`*: Space added to expected concatenation result: “`Home sweet `”.
3.7	8 Apr 2008	PV	*About this guide*: Added 2.0.4 and 2.1 to covered versions. Mentioned forced writes bug. *Installing the Firebird server :: Use the Guardian?*: Added warning about Win installer not detecting existing server. *How to corrupt a database?*: Gave subsections `id` attributes. *Disabling forced writes on Windows*: Created new parent section *Disabling forced writes*, with the Windows and Linux cases as subsections. Warned against Linux forced writes bug. *License notice*: Copyright end year now 2008.

Appendix B: License notice

Alphabetical index