



Firebird 1.5 Quick Start Guide

IBPhoenix Editors

Updates, additions: Firebird Project

2 December 2005 - Document version 2.2

Table of Contents

About this guide	3
What is in the kit?	3
Classic or Superserver?	3
Default disk locations	5
Installing Firebird	6
Installing the Firebird server	6
Testing your installation	8
Other things you need	11
Performing a client-only install	13
Working with databases	14
Connecting to the sample database	14
Creating a database using isql	16
Firebird SQL	17
Safety measures	20
Security options on the server	20
Backup	21
How to corrupt a database	21
How to get help	23
The Firebird Project	23
Appendix A: Document History	25
Appendix B: License notice	26
Alphabetical index	27

About this guide

This guide is an introduction for the complete newcomer to a few essentials for getting off to a quick start with a Firebird binary kit. For the fine details of configuring and running your server and tuning your installation, please refer to Chapters 4-6 of the *Using Firebird* manual, distributed on the [IB-Phoenix CD](#).

The *Firebird Quick Start Guide* started life 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 the guide to the Firebird Project.

Important

Before you read on, verify that your Firebird version matches this guide. This guide covers versions 1.5 and higher. If you run Firebird 1.0, 1.0.2 or 1.0.3, get the appropriate version of the Quick Start Guide at <http://www.firebirdsql.org/manual/> (HTML) or <http://www.firebirdsql.org/pdfmanual/> (PDF).

What is in the kit?

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

- The Firebird server executable.
- A client library located on the server machine.
- 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 flavors, 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 1.5 Classic Server vs. Superserver

Classic Server	Superserver
On Linux: fully mature. On Windows: fully mature as from version 1.5.1.	Fully mature on both Windows and Linux.
Creates a process for each client connection, each with its own cache. Less resource use if the number of connections is low.	Single server process with a separate thread for each connection. Shared cache space. More efficient if the number of simultaneous connections grows.
Permits fast, direct I/O to database files for local connections on Linux. On Windows, you must make local connections network-style, by connecting to <code>localhost</code> .	On Linux, local connections are made network-style, via <code>localhost</code> (often implicitly). On Windows, this is optional; you can also make direct local connections, but these are not as fast as the “Classic” ones on Linux and also less safe.
1.5: Partially implemented <i>Services Manager</i> , supporting tasks like backup/restore, database shutdown etc. over the network. Other service tasks have to be performed locally using the client tools (small separate executables) that come with Firebird. 1.5.1 and up: full Services Manager.	Full <i>Services Manager</i> (on Windows and Linux) enables you to perform management tasks (backup/restore, database shutdown, user management, stats, etc.) programmatically. You can connect to the Services Manager over the network and thus perform these tasks remotely.
SMP (symmetrical multi-processor) support. Better performance in case of a small number of multiple 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 <i>CpuAffinityMask</i> parameter in the configuration file <code>firebird.conf</code> .

As you can see, neither of the architectures is better in all respects. This is hardly surprising: we wouldn't maintain two 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 (except if your apps call unsupported or malfunctioning Services Manager functions in Classic).

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.

Note

We've not been completely honest with you. Firebird has a third flavor: *Embedded Server*, but this is an entirely different beast and not intended for your typical client-server setups. Consult the Release Notes for details.

Default disk locations

The following table describes the default disk locations for the components on Windows and Linux.

In the major codebase revision starting at v.1.5, the old hard links to InterBase artifacts were removed and many of the major components were renamed. As a result, Firebird 1.5 permits a running InterBase® server to be present, provided the two servers don't listen on the same TCP/IP port. If you need this feature, read *Configuring the port service* in the Release Notes that came with your Firebird installation (look in the `doc` subdir).

Table 2. Components of the Firebird 1.5 installation

Platform	Component	File Name	Default Location
32-bit and 64-bit Windows (Windows 95, 98, ME, NT, 2000, XP, ...)	Installation directory (referred to hereafter as <InstallDir>)		C:\Program Files\Firebird\Firebird_1_5
	Firebird server	fbserver.exe (SS) or fb_inet_server.exe (CS)	<InstallDir>\bin
	Command-line tools	gbak.exe, gfix.exe, gstat.exe, etc.	<InstallDir>\bin
	Sample database	employee.fdb	<InstallDir>\examples
	User-defined function (UDF) libraries	ib_udf.dll & fbudf.dll	<InstallDir>\UDF
	Firebird client	fbclient.dll (with an optional gds32.dll, to support legacy apps)	<InstallDir>\bin (with an optional copy in the Windows System dir – see note below table)

Platform	Component	File Name	Default Location
Linux and possibly other UNIX distros	Installation directory (referred to hereafter as <InstallDir>)		/opt/firebird
	Firebird server	fbserver (SS) or fb_inet_server (CS)	<InstallDir>/bin
	Command-line tools	gbak, gfix, gstat, etc.	<InstallDir>/bin
	Sample database	employee.fdb	<InstallDir>/examples
	UDF libraries	ib_udf.so, fbudf.so	<InstallDir>/UDF
	Firebird client	libfbclient.so.-1.5.n (binary); libfbclient.so.1, libfbclient.so (symlinks) 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)

Note

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

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

Installing Firebird

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 file or decompressing a structured WinZip .zip file – it is strongly recommended that you use the distributed release kit the first time you install Firebird. The Windows executable installation script, the Linux rpm (*RPM Package Manager*, originally *RedHat Package Manager*) program and the official .tar.gz for other Posix platforms perform some essential setup tasks. Provided you follow the instructions correctly, there should be nothing for you to do upon completion but log in and go!

Windows platforms

The Firebird installer lets you choose between Superserver and Classic Server installation. As said before, you should choose Superserver unless you know the differences and have reasons to prefer Classic.

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. We'll give you a link to a usable applet later on in this guide.

On server platforms – Windows NT, 2000 and XP – the Firebird service will be running when the installation completes. Next time you boot up your server machine, it will be started automatically.

The non-server Windows platforms – Windows 95, 98 and ME – do not support services. The installation will start Firebird server as an application, protected by another application known as the Guardian. If the server application should terminate abnormally for some reason, the Guardian will attempt to restart it.

Posix platforms

In all cases, read the release notes that pertain to the version of Firebird that you are going to install. 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.

Tip

If you do not find a copy of the release notes in your kit, go back to the [Downloads page](#) of the Firebird website at <http://firebird.sourceforge.net> and download a copy from there.

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 the various UNIX flavors, 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 modify the

scripts and make some manual adjustments.

Testing your installation

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

At this point, it is assumed that you will use the recommended TCP/IP protocol for your Firebird client/server network.

Note

For information about using NetBEUI protocol in an all-Windows environment, refer to Chapter 6, *Network Configuration* in the *Using Firebird* manual.

Warning

IPX/SPX networks are not supported by Firebird.

Pinging the server

Usually, the first thing you will want to do once installation is complete is ping the server. This just gives you a reality check to ensure that your client machine is able to see the host machine in your 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 and type the command

ping 192.13.14.1

substituting this example IP address for the IP address that your server is broadcasting.

Warning

If you get a timeout message, study the *Using Firebird* manual – Chapter 6: *Network Configuration*, and Chapter 7: *Troubleshooting Connections* – for further instructions.

Note that 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

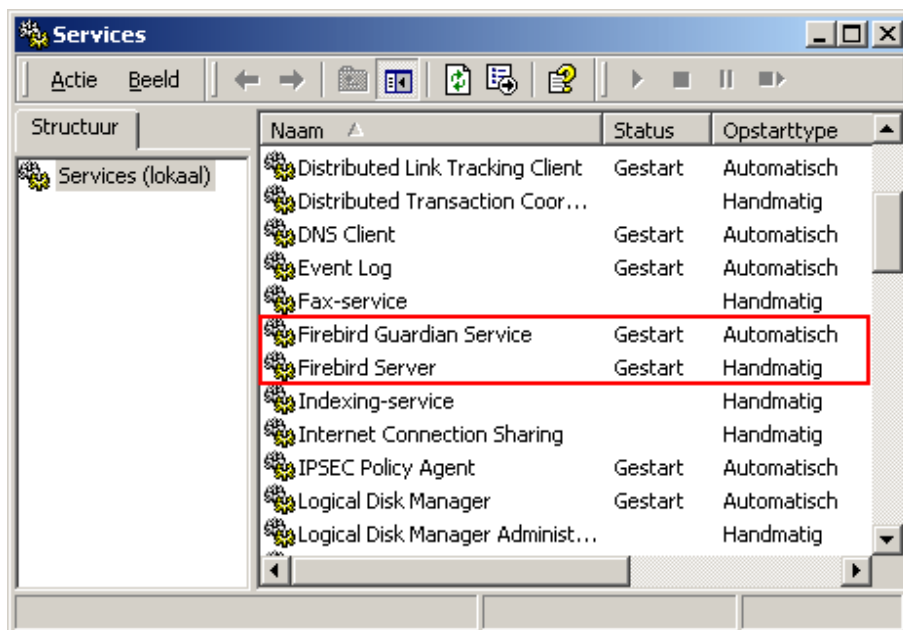
Checking that the Firebird server is running

After installation, Firebird server should be running as a service on Windows NT, 2000 or XP or on Linux.

Windows NT4, 2000 and XP

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.



If the Guardian is running (as shown in the screenshot, over) it may have a different service name because of version changes.

Note

On Windows 2000 and XP, the Guardian is a convenience rather than a necessity, since these two operating systems have the facility to watch and restart services. It is recommended that you keep Guardian active for other platforms if a SYSDBA is not available to restart the server manually in the event that it is stopped for some reason.

Windows 9x or ME

On Windows 9x or ME Firebird server should be running as an application, monitored by the Guardian. The Guardian's icon should appear in the tray with a green graphic. If the icon is flashing or showing as a red graphic, it indicates that Guardian is either attempting to start the server or has failed.

If you used an installation kit that installed but did not automatically start the Guardian and the Firebird server, you can set it up as follows:

1. Locate the executable file for the Guardian program (`fbguard.exe`) and create a shortcut for it in the Startup area of your machine's Start Menu.
2. Open the Properties dialog of the shortcut and go to the field where the command line is.
3. Edit the command line so it reads as follows:

`fbguard.exe -a` (for Superserver)

`fbguard.exe -c` (for Classic Server)
4. Save and close the Properties dialog.
5. Double-click on the shortcut to start the Guardian. The Guardian will proceed to start `fbserve-`

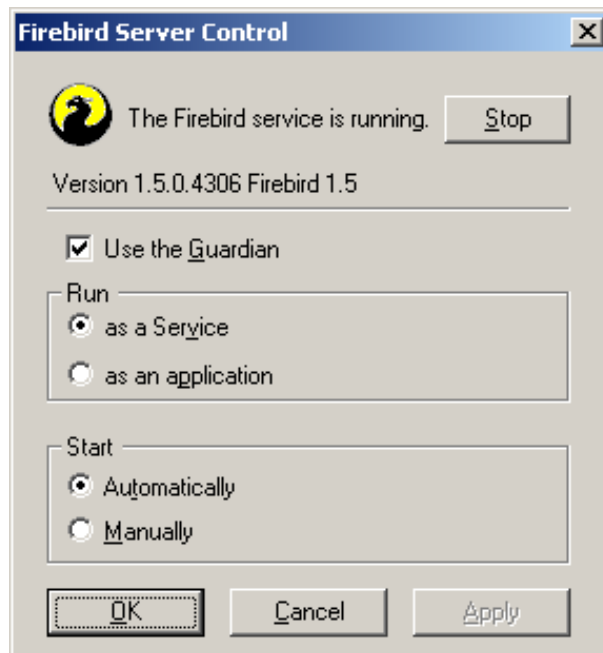
er.exe or fb_inet_server.exe.

The Guardian should start up automatically next time you boot your Windows 9x or ME machine.

Alternatively, you can use a Control Panel applet to control the starting and stopping of the Firebird server.

Windows Control Panel applets

Since version 1.0.3, a control panel applet is included in the Firebird distribution. Whilst the applet is not essential, it does provide a convenient way to start and stop the server.



Unfortunately, the bundled applet only works on Windows NT, 2000 and XP. On Windows 9x and ME, if you want a handy applet like this, visit this webpage:

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

and download the Firebird Control Center [fbcc-0.2.6.exe](#).

This applet looks different from the above screenshot, but offers the same functionality.

Posix servers

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.

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.

The following screen shows the output of top, restricted by grep to show only processes with names starting with 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`.

Other things you need

A network address for the server

- If you are on a managed network, get the server's IP address from your system administrator.
- 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.
- If you are intending to try out a single-machine installation of both client and server, you should use the local loopback server address – localhost, with the IP address 127.0.0.1

Note

It is possible to connect directly to a local Windows Superserver, without using the TCP/IP loopback. This is not a TCP/IP connection and it is not a thread-safe way to connect to a local server. For using single instances of the command-line tools (`gsec`, `gbak` etc.) it works just fine. By contrast, direct database connections - even multiple - under a Linux Classic server are completely safe.

Default user name and password

The SYSDBA user has all privileges on the server. 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 SYSDBA password and store that in `/opt/firebird/SYSDBA.password`

If your server is exposed to the Internet *at all* and the password is `masterkey`, you should change it immediately using the `gsec` command-line utility.

How to change the SYSDBA password

Firebird comes with a command-line tool called `gsec` that is used to manipulate user accounts.

Important

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.

If you have enough privileges but invoking gsec results in a message like “unavailable database - unable to open database”:

- you're either running Windows Classic Server and you didn't provide a correct `-database` argument (see below), or
- the server may not be running at all. In that case, go back to *Testing your installation* and fix the problem.

Let's say you decide to change the SYSDBA password to `icurry4me`.

1. Go to a command shell on your server and change to the directory where the command-line utilities are located. Refer to the [Firebird installation components table](#) to find this location.
2. Type the following command, *except* if you run Windows Classic Server:

gsec -user sysdba -password masterkey

In case of a Windows Classic Server, you *must* specify the full network location of the security database (unfortunately!):

gsec -user sysdba -password masterkey -database "localhost:C:\Program Files\Firebird\Firebird_1_5\security.fdb" (adapt the path if necessary)

Note

- On Linux, type `./gsec` rather than `gsec`. Otherwise there's a chance that a “wrong” gsec is launched, or that it isn't found at all.
- Paths and file names are case-sensitive on all platforms except Windows; passwords are always case-sensitive.

In either case, you should now see the shell prompt for the gsec utility:

GSEC>

3. Type this command:

modify sysdba -pw icurry4me

4. Press **Enter**. The new password `icurry4me` is now encrypted and saved and `masterkey` is no longer valid.
5. Now quit the gsec shell:

quit

Note

Because Firebird ignores all characters in a password past the eighth character, `icurry4m` will work, as will `icurry4monkeys`.

An Admin tool

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.

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 > Admin Tools](http://www.ibphoenix.com) page at <http://www.ibphoenix.com> for all of the options.

Note

You can use a Windows client to access a Linux server and vice-versa.

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 compact installation program is available to assist with installing the client pieces on a Windows client. If you are in the common situation of running Windows clients to a Linux or other Posix 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 Linux or other server machine.

Fortunately, once you have the kit, the Windows client-only install is easy to do. Run the installation program, just as though you were going to install the server – but select the **CLIENT ONLY** option from the install menu.

Linux and some other Posix clients

A small-footprint client install program for Linux clients is not available either. Additionally, some Posix flavors – 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 flavors, the following procedure is suggested for a Firebird client-only install. Log in as `root` for this.

1. Look for `libfbclient.so.1.m.n` (*m.n* being the minor plus patch version number) in `/opt/firebird/lib` on the server where 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.1.m.n /usr/lib/libfbclient.so.1
```

```
ln -s /usr/lib/libfbclient.so.1 /usr/lib/libfbclient.so
```

replacing *1.m.n* with your version number, e.g. *1.5.0* or *1.6.1*

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 the `firebird.msg` file to `/opt/firebird`
4. In the system-wide default shell profile, or using `setenv()` from a shell, create the `FIREBIRD` environment variable and point it to `/opt/firebird`, to enable the API routines to locate the messages.

Working with databases

Connecting to the sample database

In the `examples` subdirectory of your Firebird installation is a sample database named `employee.fdb`. You can use this database to “try your wings”.

Server name and path

If you move the sample database, be sure you move it to 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.

There are two elements to a TCP/IP connection string: the server name and the disk/filesystem path. Its format is as follows:

- For a Linux server:

```
servername:/filesystem-path/database-file
```

Example on a Linux or other Posix server named `serverxyz`:

```
serverxyz:/opt/interbase/examples/employee.fdb
```

- For a Windows server:

```
servername:DriveLetter:\filesystem-path\database-file
```

Windows example:

```
serverxyz:C:\Program Files\Firebird\examples\employee.fdb
```

The CONNECT statement

Connecting to a Firebird database requires the user to authenticate using a user name and a valid password. Any user other than SYSDBA, root (on Posix systems), or Administrator (on Windows systems, if Firebird runs as this user) needs also to have permissions to objects inside a database. For simplicity here, we will look at authenticating as SYSDBA using the password masterkey.

Using isql

There are several different ways to connect to a database using isql. One way is to start isql in its interactive shell. Go to the bin subdirectory of your Firebird installation and, at that prompt, type the command **isql** (on Linux: **./isql**) [# means “hit **Enter**”]:

```
C:\Program Files\Firebird\Firebird_1_5\bin>isql#  
Use CONNECT or CREATE DATABASE to specify a database  
SQL>CONNECT "C:\Program Files\Firebird\Firebird_1_5\examples\employee.fdb"#  
CON>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 spread long statements over multiple lines. If you hit **Enter** after your statement and you've forgotten the semicolon, just type it on the empty line after the CON> prompt 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 1.5 usually running as 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.
- If you run Classic Server on Windows, you *must* specify a hostname (which may be localhost) plus a full path, or the connection will fail.

Note

Although single quote symbols are the “norm” for delimiting strings in Firebird, double quote symbols were used with the database path string in the above example. This is sometimes necessary with some of the command-line utilities where the path string contains spaces. Single quotes should work for paths that do not contain spaces.

The quotes around “SYSDBA” and “masterkey” are optional, by the way. Database paths without spaces also don't need to be quoted.

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

```
DATABASE "C:\Program Files\Firebird\Firebird_1_5\examples\employee.fdb",  
User: sysdba  
SQL>
```

You can now continue to play about with the `employee.fdb` database. The characters `isql` stand for *interactive SQL [utility]*. You can use it for querying data, getting information about the metadata, creating database objects, running data definition scripts and much more.

To get back to the command prompt type

```
SQL>QUIT;#
```

For more information about `isql`, see *Using Firebird*, Chapter 10: *Interactive SQL Utility (isql)*.

Using a GUI client

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

Note

- It is quite common for such tools to expect the entire server + path as a single string
- Remember that file names and commands on Linux and other Posix command shells are case-sensitive

Creating a database using isql

There is more than one way to create a database using `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. There is a complete chapter in the *Using Firebird* manual discussing this topic.

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_1_5\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 the quotes around path string, username, and password are *mandatory*. 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 1.5 usually running as `firebird`) will create and own the file.
- If you run Classic Server on Windows, you *must* specify a hostname (which may be `localhost`) plus a full path, or the creation will fail.

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.

To verify that there really is a database there, type in this query:

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

The screen will fill up with a large amount of data! This query selects all of the rows in the system table where Firebird stores the metadata for tables. An “empty” database is not empty – it contains a database which will become populated with metadata as you begin creating objects in your database.

To get back to the command prompt type

```
SQL>QUIT;#
```

For more information about `isql`, see *Using Firebird*, Chapter 10: *Interactive SQL Utility (isql)*.

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 any other RDBMS except possibly its “cousin”, InterBase®. 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 a 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 this 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 characters “Reported by: ”:

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

Take care with concatenations. Be aware that Firebird will raise an error if your expression attempts to concatenate two or more char or varchar columns whose potential combined lengths would exceed the maximum length limit for a char or a varchar (32 Kb).

See also the note 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 RDBMSs 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.

So – here are some examples of the types of surprises you will get if you try to perform calculations and comparisons with NULL:

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

- `FirstName || ' ' || LastName`

will return `NULL` if either `FirstName` or `LastName` is `NULL`.

Tip

Think of `NULL` as `UNKNOWN` and all 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.

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)

Safety measures

Security options on the server

Firebird 1.5 has a number of new security provisions. Explore them! Many configurable features default to the old, “insecure” behaviour in order not to break existing applications, but you can significantly enhance your system's security if you raise the protection level wherever possible.

Amongst the new and/or changed security features are:

- On Posix systems, Firebird now 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 about this setup.
- *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/fire-`

`bird/music/underground/mothers_of_invention.fdb`. Aliases also allow you to relocate databases while the clients keep using their existing connection strings.

- *DatabaseAccess* parameter; 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.
- *ExternalFileAccess* parameter, for controlling access to external tables.
- *UdfAccess* parameter; controls the permitted locations for user-defined function libraries.

Database aliases live in the file `aliases.conf`, configuration parameters in `firebird.conf`. Please consult the Release Notes of your Firebird version for their exact usage (and some additional well-meant advice).

Backup

Firebird comes with its own utility for backing up and restoring your databases. Its name is `gbak` and it 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 state at the time the backup began.

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

Warning

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!

How to corrupt a database

The following is a list of things *not* to do if you want to keep your Firebird databases healthy.

Modifying metadata tables yourself

Firebird stores and maintains all of the metadata for its own and your user-defined objects in – a Firebird database! More precisely, it stores them in relations (tables) right in the database itself. The identifiers for the 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 on Windows

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.

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 re-booting.

Note

Windows 9x and ME do not support deferred data writes

Disabling Forced Writes on a Linux server

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.

Restoring a backup to a running database

One of the restore options in the gbak utility (`gbak -r[eplace]`) 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.

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.

Note

For gbak instructions see chapter 21, *Database Backup and Restore*, of *Using Firebird*.

For instructions about blocking access to users, see chapter 14, *Getting exclusive access to a database*, of *Using Firebird*.

If is practicable to do so, it is recommended to restore to spare disk space using the `gbak -c [recreate]` 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 -r[ep]ace` 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://firebird.sourceforge.net> and join the user support lists.
- 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.
- See the growing list of documentation that has been produced within the Firebird project itself at <http://firebird.sourceforge.net/manual/>.
- 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.

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 docs will be brought up to date and added 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 bug tracker and a number of technical files which can be downloaded for various purposes 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 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 `firebird-support@yahoo.com`.

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 <i>Using Firebird</i> .
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 March 2005	PV	Changed <code>gbak r[estore]</code> to <code>r[eplace]</code> in two places.
2.1.2	8 April 2005	PV	Reordered Firebird SQL subsections. Added links to Firebird Null Guide.
2.2	2 Dec 05	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.

Appendix B

License notice

The contents of this Documentation are subject to the Public Documentation License Version 1.0 (the “License”); you may only use this Documentation if you comply with the terms of this License. Copies of the License are available at <http://www.firebirdsql.org/pdfmanual/pdl.pdf> (PDF) and <http://www.firebirdsql.org/manual/pdl.html> (HTML).

The Original Documentation is titled *Firebird Quick Start Guide*.

The Initial Writer of the Original Documentation is: IBPhoenix Editors.

Copyright (C) 2002-2004. All Rights Reserved. Initial Writer contact: hborrie at ibphoenix dot com.

Contributor: Paul Vinkenoog - see [document history](#).

Portions created by Paul Vinkenoog are Copyright (C) 2004-2005. All Rights Reserved. Contributor contact: paul at vinkenoog dot nl.

Alphabetical index

A

Admin tools, 13
Aliasing, 20
Apostrophes in strings, 18

B

Backup, 21
Books
 The Firebird Book, 23

C

Checking the server, 8
Classic Server, 3
CONNECT statement, 15
Connecting, 14
Control Panel applets, 10
CREATE DATABASE statement, 16

D

Databases
 aliases, 20
 backup and restore, 21 , 22 , 23
 connecting, 14
 with a GUI clientl, 16
 with isql, 15
 corruption, 21
 creating with isql, 16
 example database, 14
 metadata, 21
 security, 20
 system tables, 21
Disk locations, 5
Document history, 25
Documentation, 23
Double-quoted identifiers, 18

E

Example database, 14

F

Firebird Book, 23
Firebird project, 23
Firebird SQL, 17
Forced writes, 22

G

gsec, 11
Guardian, 7 , 9 , 9 , 10

H

Help, 23

I

Installation, 6
 Classic or Superserver, 3
 client-only, 13
 drives, 6
 kit contents, 3
 script or program, 7
 server, 6
Integer division, 17
isql
 connecting to a database, 15
 creating a database, 16

L

License notice, 26

N

Network address, 11
NULL, 19

P

Passwords
 changing, 11
 default, 11
Ping, 8
Project, 23

R

Restore, 21
 to a running database, 22
 user logins during restore, 23

S

Sample database, 14
Security, 20
Server name and path, 14
Services (Windows), 9
SQL, 17
 CONNECT statement, 15
 CREATE DATABASE statement, 16
Strings
 apostrophes in stringsl, 18
 concatenation, 18
 delimiter symbol, 18
Superserver, 3
SYSDBA, 11
System tables, 21

T

Testing, 8
top command (Linux), 10

U

User names
 default, 11