Preventing data loss

Backup

Firebird comes with two utilities for backing up and restoring your databases: gbak and nbackup (*note*: nbackup is broken in 2.1; fixed with errors in 2.1.1; completely fixed in 2.1.2). 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):

https://www.firebirdsql.org/manual/nbackup.html
https://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 old database (you can use the gfix command-line tool for this; see the InterBase® 6.0 Operations Guide; a link is given in the next section). Make a file system copy of the old database just in case and then copy the restored database file(s) 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.