



# Bacula Catalog Database Guide

It comes in the night and sucks the essence from your computers.

Kern Sibbald

January 13, 2010

This manual documents Bacula version 3.0.3a (06 December 2009)

Copyright ©1999-2009, Free Software Foundation Europe e.V.

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License".



# Contents

<b>1</b>	<b>Catalog Maintenance</b>	<b>5</b>
1.1	Setting Retention Periods . . . . .	5
1.2	Compacting Your MySQL Database . . . . .	6
1.3	Repairing Your MySQL Database . . . . .	7
1.4	MySQL Table is Full . . . . .	7
1.5	MySQL Server Has Gone Away . . . . .	8
1.6	MySQL Temporary Tables . . . . .	8
1.7	Repairing Your PostgreSQL Database . . . . .	8
1.8	Database Performance Issues . . . . .	8
1.9	Performance Issues Indexes . . . . .	9
1.9.1	PostgreSQL Indexes . . . . .	9
1.9.2	MySQL Indexes . . . . .	9
1.9.3	SQLite Indexes . . . . .	10
1.10	Compacting Your PostgreSQL Database . . . . .	10
1.11	Compacting Your SQLite Database . . . . .	11
1.12	Migrating from SQLite to MySQL or PostgreSQL . . . . .	11
1.13	Backing Up Your Bacula Database . . . . .	11
1.14	Security considerations . . . . .	12
1.15	Backing Up Third Party Databases . . . . .	13
1.16	Database Size . . . . .	13
<b>2</b>	<b>Installing and Configuring MySQL</b>	<b>15</b>
2.1	Installing and Configuring MySQL – Phase I . . . . .	15
2.2	Installing and Configuring MySQL – Phase II . . . . .	16
2.3	Re-initializing the Catalog Database . . . . .	17
2.4	Linking Bacula with MySQL . . . . .	17

2.5	Installing MySQL from RPMs . . . . .	18
2.6	Upgrading MySQL . . . . .	18
<b>3</b>	<b>Installing and Configuring PostgreSQL</b>	<b>19</b>
3.1	Installing PostgreSQL . . . . .	19
3.2	Configuring PostgreSQL . . . . .	20
3.3	Re-initializing the Catalog Database . . . . .	22
3.4	Installing PostgreSQL from RPMs . . . . .	22
3.5	Converting from MySQL to PostgreSQL . . . . .	22
3.6	Upgrading PostgreSQL . . . . .	24
3.7	Tuning PostgreSQL . . . . .	24
3.8	Credits . . . . .	24
<b>4</b>	<b>Installing and Configuring SQLite</b>	<b>25</b>
4.1	Installing and Configuring SQLite – Phase I . . . . .	25
4.2	Installing and Configuring SQLite – Phase II . . . . .	26
4.3	Linking Bacula with SQLite . . . . .	26
4.4	Testing SQLite . . . . .	26
4.5	Re-initializing the Catalog Database . . . . .	26
<b>5</b>	<b>The internal database is not supported, please do not use it.</b>	<b>29</b>
5.1	Internal Bacula Database . . . . .	29
<b>6</b>	<b>GNU Free Documentation License</b>	<b>31</b>

# Chapter 1

## Catalog Maintenance

Without proper setup and maintenance, your Catalog may continue to grow indefinitely as you run Jobs and backup Files, and/or it may become very inefficient and slow. How fast the size of your Catalog grows depends on the number of Jobs you run and how many files they backup. By deleting records within the database, you can make space available for the new records that will be added during the next Job. By constantly deleting old expired records (dates older than the Retention period), your database size will remain constant.

If you started with the default configuration files, they already contain reasonable defaults for a small number of machines (less than 5), so if you fall into that case, catalog maintenance will not be urgent if you have a few hundred megabytes of disk space free. Whatever the case may be, some knowledge of retention periods will be useful.

### 1.1 Setting Retention Periods

**Bacula** uses three Retention periods: the **File Retention** period, the **Job Retention** period, and the **Volume Retention** period. Of these three, the File Retention period is by far the most important in determining how large your database will become.

The **File Retention** and the **Job Retention** are specified in each Client resource as is shown below. The **Volume Retention** period is specified in the Pool resource, and the details are given in the next chapter of this manual.

**File Retention = <time-period-specification>** The File Retention record defines the length of time that Bacula will keep File records in the Catalog database. When this time period expires, and if **AutoPrune** is set to **yes**, Bacula will prune (remove) File records that are older than the specified File Retention period. The pruning will occur at the end of a backup Job for the given Client. Note that the Client database record contains a copy of the File and Job retention periods, but Bacula uses the current values found in the Director's Client resource to do the pruning.

Since File records in the database account for probably 80 percent of the size of the database, you should carefully determine exactly what File Retention period you need. Once the File records have been removed from the database, you will no longer be able to restore individual files in a Job. However, with Bacula version 1.37 and later, as long as the Job record still exists, you will be able to restore all files in the job.

Retention periods are specified in seconds, but as a convenience, there are a number of modifiers that permit easy specification in terms of minutes, hours, days, weeks, months, quarters, or years on the record. See the Configuration chapter of this manual for additional details of modifier specification.

The default File retention period is 60 days.

**Job Retention = <time-period-specification>** The Job Retention record defines the length of time that **Bacula** will keep Job records in the Catalog database. When this time period expires, and if

**AutoPrune** is set to **yes** Bacula will prune (remove) Job records that are older than the specified Job Retention period. Note, if a Job record is selected for pruning, all associated File and JobMedia records will also be pruned regardless of the File Retention period set. As a consequence, you normally will set the File retention period to be less than the Job retention period.

As mentioned above, once the File records are removed from the database, you will no longer be able to restore individual files from the Job. However, as long as the Job record remains in the database, you will be able to restore all the files backed up for the Job (on version 1.37 and later). As a consequence, it is generally a good idea to retain the Job records much longer than the File records.

The retention period is specified in seconds, but as a convenience, there are a number of modifiers that permit easy specification in terms of minutes, hours, days, weeks, months, quarters, or years. See the Configuration chapter of this manual for additional details of modifier specification.

The default Job Retention period is 180 days.

**AutoPrune** = **<yes/no>** If **AutoPrune** is set to **yes** (default), Bacula will automatically apply the File retention period and the Job retention period for the Client at the end of the Job.

If you turn this off by setting it to **no**, your Catalog will grow each time you run a Job.

## 1.2 Compacting Your MySQL Database

Over time, as noted above, your database will tend to grow. I've noticed that even though Bacula regularly prunes files, **MySQL** does not effectively use the space, and instead continues growing. To avoid this, from time to time, you must compact your database. Normally, large commercial database such as Oracle have commands that will compact a database to reclaim wasted file space. MySQL has the **OPTIMIZE TABLE** command that you can use, and SQLite version 2.8.4 and greater has the **VACUUM** command. We leave it to you to explore the utility of the **OPTIMIZE TABLE** command in MySQL.

All database programs have some means of writing the database out in ASCII format and then reloading it. Doing so will re-create the database from scratch producing a compacted result, so below, we show you how you can do this for MySQL, PostgreSQL and SQLite.

For a **MySQL** database, you could write the Bacula database as an ASCII file (bacula.sql) then reload it by doing the following:

```
mysqldump -f --opt bacula > bacula.sql
mysql bacula < bacula.sql
rm -f bacula.sql
```

Depending on the size of your database, this will take more or less time and a fair amount of disk space. For example, if I cd to the location of the MySQL Bacula database (typically /opt/mysql/var or something similar) and enter:

```
du bacula
```

I get **620,644** which means there are that many blocks containing 1024 bytes each or approximately 635 MB of data. After doing the **mysqldump**, I had a bacula.sql file that had **174,356** blocks, and after doing the **mysql** command to recreate the database, I ended up with a total of **210,464** blocks rather than the original **629,644**. In other words, the compressed version of the database took approximately one third of the space of the database that had been in use for about a year.

As a consequence, I suggest you monitor the size of your database and from time to time (once every six months or year), compress it.

## 1.3 Repairing Your MySQL Database

If you find that you are getting errors writing to your MySQL database, or Bacula hangs each time it tries to access the database, you should consider running MySQL's database check and repair routines. The program you need to run depends on the type of database indexing you are using. If you are using the default, you will probably want to use **myisamchk**. For more details on how to do this, please consult the MySQL document at: <http://www.mysql.com/doc/en/Repair.html>.

If the errors you are getting are simply SQL warnings, then you might try running `dbcheck` before (or possibly after) using the MySQL database repair program. It can clean up many of the orphaned record problems, and certain other inconsistencies in the Bacula database.

A typical cause of MySQL database problems is if your partition fills. In such a case, you will need to create additional space on the partition or free up some space then repair the database probably using **myisamchk**. Recently my root partition filled and the MySQL database was corrupted. Simply running **myisamchk -r** did not fix the problem. However, the following script did the trick for me:

```
#!/bin/sh
for i in *.MYD ; do
  mv $i x${i}
  t='echo $i | cut -f 1 -d \. ' -'
  mysql bacula <<END_OF_DATA
set autocommit=1;
truncate table $t;
quit
END_OF_DATA
  cp x${i} ${i}
  chown mysql:mysql ${i}
  myisamchk -r ${t}
done
```

I invoked it with the following commands:

```
cd /var/lib/mysql/bacula
./repair
```

Then after ensuring that the database was correctly fixed, I did:

```
cd /var/lib/mysql/bacula
rm -f x*.MYD
```

## 1.4 MySQL Table is Full

If you are running into the error **The table 'File' is full ...**, it is probably because on version 4.x MySQL, the table is limited by default to a maximum size of 4 GB and you have probably run into the limit. The solution can be found at: <http://dev.mysql.com/doc/refman/5.0/en/full-table.html>

You can display the maximum length of your table with:

```
mysql bacula
SHOW TABLE STATUS FROM bacula like "File";
```

If the column labeled "Max\_data.length" is around 4Gb, this is likely to be the source of your problem, and you can modify it with:

```
mysql bacula
ALTER TABLE File MAX_ROWS=281474976710656;
```

Alternatively you can modify your `/etc/my.conf` file before creating the Bacula tables, and in the `[mysqld]` section set:

```
set-variable = myisam_data_pointer_size=6
```

The above myisam data pointer size must be made before you create your Bacula tables or it will have no effect.

The row and pointer size changes should already be the default on MySQL version 5.x, so making these changes should only be necessary on MySQL 4.x depending on the size of your catalog database.

## 1.5 MySQL Server Has Gone Away

If you are having problems with the MySQL server disconnecting or with messages saying that your MySQL server has gone away, then please read the MySQL documentation, which can be found at:

<http://dev.mysql.com/doc/refman/5.0/en/gone-away.html>

## 1.6 MySQL Temporary Tables

When doing backups with large numbers of files, MySQL creates some temporary tables. When these tables are small they can be held in system memory, but as they approach some size, they spool off to disk. The default location for these temp tables is `/tmp`. Once that space fills up, Bacula daemons such as the Storage daemon doing spooling can get strange errors. E.g.

```
Fatal error: spool.c:402 Spool data read error.  
Fatal error: backup.c:892 Network send error to SD. ERR=Connection reset by  
peer
```

What you need to do is setup MySQL to use a different (larger) temp directory, which can be set in the `/etc/my.cnf` with these variables set:

```
tmpdir=/path/to/larger/tmpdir  
bdb_tmpdir=/path/to/larger/tmpdir
```

## 1.7 Repairing Your PostgreSQL Database

The same considerations apply that are indicated above for MySQL. That is, consult the PostgreSQL documents for how to repair the database, and also consider using Bacula's `dbcheck` program if the conditions are reasonable for using (see above).

## 1.8 Database Performance Issues

There are a considerable number of ways each of the databases can be tuned to improve the performance. Going from an untuned database to one that is properly tuned can make a difference of a factor of 100 or more in the time to insert or search for records.

For each of the databases, you may get significant improvements by adding additional indexes. The comments in the Bacula `make_XXX_tables` give some indications as to what indexes may be appropriate. Please see below for specific instructions on checking indexes.

For MySQL, what is very important is to use the examine the my.cnf file (usually in /etc/my.cnf). You may obtain significant performances by switching to the my-large.cnf or my-huge.cnf files that come with the MySQL source code.

For SQLite3, one significant factor in improving the performance is to ensure that there is a "PRAGMA synchronous = NORMAL;" statement. This reduces the number of times that the database flushes the in memory cache to disk. There are other settings for this PRAGMA that can give even further performance improvements at the risk of a database corruption if your system crashes.

For PostgreSQL, you might want to consider turning fsync off. Of course doing so can cause corrupted databases in the event of a machine crash. There are many different ways that you can tune PostgreSQL, the following document discusses a few of them: <http://www.varlena.com/varlena/GeneralBits/Tidbits/perf.html>.

There is also a PostgreSQL FAQ question number 3.3 that may answer some of your questions about how to improve performance of the PostgreSQL engine: <http://www.postgresql.org/docs/faqs.FAQ.html#3.3>.

Also for PostgreSQL, look at what "effective\_cache\_size". For a 2GB memory machine, you probably want to set it at 131072, but don't set it too high. In addition, for a 2GB system, work\_mem = 256000 and maintenance\_work\_mem = 256000 seem to be reasonable values. Make sure your checkpoint\_segments is set to at least 8.

## 1.9 Performance Issues Indexes

One of the most important considerations for improving performance on the Bacula database is to ensure that it has all the appropriate indexes. Several users have reported finding that their database did not have all the indexes in the default configuration. In addition, you may find that because of your own usage patterns, you need additional indexes.

The most important indexes for performance are the two indexes on the **File** table. The first index is on **FileId** and is automatically made because it is the unique key used to access the table. The other one is the (JobId, PathId, Filename) index. If these Indexes are not present, your performance may suffer a lot.

### 1.9.1 PostgreSQL Indexes

On PostgreSQL, you can check to see if you have the proper indexes using the following commands:

```
psql bacula
select * from pg_indexes where tablename='file';
```

If you do not see output that indicates that all three indexes are created, you can create the two additional indexes using:

```
psql bacula
CREATE INDEX file_jobid_idx on file (jobid);
CREATE INDEX file_fp_idx on file (jobid, pathid, filenameid);
```

### 1.9.2 MySQL Indexes

On MySQL, you can check if you have the proper indexes by:

```
mysql bacula
show index from File;
```

If the indexes are not present, especially the JobId index, you can create them with the following commands:

```
mysql bacula
CREATE INDEX file_jobid_idx on File (JobId);
CREATE INDEX file_jpf_idx on File (JobId, FilenameId, PathId);
```

Though normally not a problem, you should ensure that the indexes defined for Filename and Path are both set to 255 characters. Some users reported performance problems when their indexes were set to 50 characters. To check, do:

```
mysql bacula
show index from Filename;
show index from Path;
```

and what is important is that for Filename, you have an index with Key\_name "Name" and Sub\_part "255". For Path, you should have a Key\_name "Path" and Sub\_part "255". If one or the other does not exist or the Sub\_part is less than 255, you can drop and recreate the appropriate index with:

```
mysql bacula
DROP INDEX Path on Path;
CREATE INDEX Path on Path (Path(255));

DROP INDEX Name on Filename;
CREATE INDEX Name on Filename (Name(255));
```

### 1.9.3 SQLite Indexes

On SQLite, you can check if you have the proper indexes by:

```
sqlite <path>/bacula.db
select * from sqlite_master where type='index' and tbl_name='File';
```

If the indexes are not present, especially the JobId index, you can create them with the following commands:

```
sqlite <path>/bacula.db
CREATE INDEX file_jobid_idx on File (JobId);
CREATE INDEX file_jfp_idx on File (JobId, PathId, FilenameId);
```

## 1.10 Compacting Your PostgreSQL Database

Over time, as noted above, your database will tend to grow. I've noticed that even though Bacula regularly prunes files, PostgreSQL has a **VACUUM** command that will compact your database for you. Alternatively you may want to use the **vacuumdb** command, which can be run from a cron job.

All database programs have some means of writing the database out in ASCII format and then reloading it. Doing so will re-create the database from scratch producing a compacted result, so below, we show you how you can do this for PostgreSQL.

For a **PostgreSQL** database, you could write the Bacula database as an ASCII file (bacula.sql) then reload it by doing the following:

```
pg_dump -c bacula > bacula.sql
cat bacula.sql | psql bacula
rm -f bacula.sql
```

Depending on the size of your database, this will take more or less time and a fair amount of disk space. For example, you can **cd** to the location of the Bacula database (typically /usr/local/pgsql/data or possible /var/lib/pgsql/data) and check the size.

There are certain PostgreSQL users who do not recommend the above procedure. They have the following to say: PostgreSQL does not need to be dumped/restored to keep the database efficient. A normal process of vacuuming will prevent the database from every getting too large. If you want to fine-tweak the database storage, commands such as `VACUUM FULL`, `REINDEX`, and `CLUSTER` exist specifically to keep you from having to do a dump/restore.

Finally, you might want to look at the PostgreSQL documentation on this subject at <http://www.postgresql.org/docs/8.1/interactive/maintenance.html>.

## 1.11 Compacting Your SQLite Database

First please read the previous section that explains why it is necessary to compress a database. SQLite version 2.8.4 and greater have the **Vacuum** command for compacting the database.

```
cd {\bf working-directory}
echo 'vacuum;' | sqlite bacula.db
```

As an alternative, you can use the following commands, adapted to your system:

```
cd {\bf working-directory}
echo '.dump' | sqlite bacula.db > bacula.sql
rm -f bacula.db
sqlite bacula.db < bacula.sql
rm -f bacula.sql
```

Where **working-directory** is the directory that you specified in the Director's configuration file. Note, in the case of SQLite, it is necessary to completely delete (`rm`) the old database before creating a new compressed version.

## 1.12 Migrating from SQLite to MySQL or PostgreSQL

You may begin using Bacula with SQLite then later find that you want to switch to MySQL or Postgres for any of a number of reasons: SQLite tends to use more disk than MySQL; when the database is corrupted it is often more catastrophic than with MySQL or PostgreSQL. Several users have succeeded in converting by exporting the SQLite data and then processing it with Perl scripts prior to putting it into MySQL or PostgreSQL. This is, however, not a simple process. Scripts are available on bacula source distribution under `examples/database`.

## 1.13 Backing Up Your Bacula Database

If ever the machine on which your Bacula database crashes, and you need to restore from backup tapes, one of your first priorities will probably be to recover the database. Although Bacula will happily backup your catalog database if it is specified in the FileSet, this is not a very good way to do it, because the database will be saved while Bacula is modifying it. Thus the database may be in an instable state. Worse yet, you will backup the database before all the Bacula updates have been applied.

To resolve these problems, you need to backup the database after all the backup jobs have been run. In addition, you will want to make a copy while Bacula is not modifying it. To do so, you can use two scripts provided in the release **make\_catalog\_backup** and **delete\_catalog\_backup**. These files will be automatically generated along with all the other Bacula scripts. The first script will make an ASCII copy of your Bacula database into **bacula.sql** in the working directory you specified in your configuration, and the second will delete the **bacula.sql** file.

The basic sequence of events to make this work correctly is as follows:

- Run all your nightly backups
- After running your nightly backups, run a Catalog backup Job
- The Catalog backup job must be scheduled after your last nightly backup
- You use **RunBeforeJob** to create the ASCII backup file and **RunAfterJob** to clean up

Assuming that you start all your nightly backup jobs at 1:05 am (and that they run one after another), you can do the catalog backup with the following additional Director configuration statements:

```
# Backup the catalog database (after the nightly save)
Job {
  Name = "BackupCatalog"
  Type = Backup
  Client=rufus-fd
  FileSet="Catalog"
  Schedule = "WeeklyCycleAfterBackup"
  Storage = DLTDrive
  Messages = Standard
  Pool = Default
  # WARNING!!! Passing the password via the command line is insecure.
  # see comments in make_catalog_backup for details.
  RunBeforeJob = "/home/kern/bacula/bin/make_catalog_backup"
  RunAfterJob = "/home/kern/bacula/bin/delete_catalog_backup"
  Write Bootstrap = "/home/kern/bacula/working/BackupCatalog.bsr"
}
# This schedule does the catalog. It starts after the WeeklyCycle
Schedule {
  Name = "WeeklyCycleAfterBackup"
  Run = Level=Full sun-sat at 1:10
}
# This is the backup of the catalog
FileSet {
  Name = "Catalog"
  Include {
    Options {
      signature=MD5
    }
    File = \lt{}working_directory\gt{/bacula.sql
  }
}
```

Be sure to write a bootstrap file as in the above example. However, it is preferable to write or copy the bootstrap file to another computer. It will allow you to quickly recover the database backup should that be necessary. If you do not have a bootstrap file, it is still possible to recover your database backup, but it will be more work and take longer.

## 1.14 Security considerations

We provide `make_catalog_backup` as an example of what can be used to backup your Bacula database. We expect you to take security precautions relevant to your situation. `make_catalog_backup` is designed to take a password on the command line. This is fine on machines with only trusted users. It is not acceptable on machines without trusted users. Most database systems provide a alternative method, which does not place the password on the command line.

The `make_catalog_backup` script contains some warnings about how to use it. Please read those tips.

To help you get started, we know PostgreSQL has a password file, `.pgpass`, and we know MySQL has `.my.cnf`.

Only you can decide what is appropriate for your situation. We have provided you with a starting point. We hope it helps.

## 1.15 Backing Up Third Party Databases

If you are running a database in production mode on your machine, Bacula will happily backup the files, but if the database is in use while Bacula is reading it, you may back it up in an unstable state.

The best solution is to shutdown your database before backing it up, or use some tool specific to your database to make a valid live copy perhaps by dumping the database in ASCII format. I am not a database expert, so I cannot provide you advice on how to do this, but if you are unsure about how to backup your database, you might try visiting the Backup Central site, which has been renamed Storage Mountain ([www.backupcentral.com](http://www.backupcentral.com)). In particular, their Free Backup and Recovery Software page has links to scripts that show you how to shutdown and backup most major databases.

## 1.16 Database Size

As mentioned above, if you do not do automatic pruning, your Catalog will grow each time you run a Job. Normally, you should decide how long you want File records to be maintained in the Catalog and set the **File Retention** period to that time. Then you can either wait and see how big your Catalog gets or make a calculation assuming approximately 154 bytes for each File saved and knowing the number of Files that are saved during each backup and the number of Clients you backup.

For example, suppose you do a backup of two systems, each with 100,000 files. Suppose further that you do a Full backup weekly and an Incremental every day, and that the Incremental backup typically saves 4,000 files. The size of your database after a month can roughly be calculated as:

$$\text{Size} = 154 * \text{No. Systems} * (100,000 * 4 + 10,000 * 26)$$

where we have assumed four weeks in a month and 26 incremental backups per month. This would give the following:

$$\begin{aligned} \text{Size} &= 154 * 2 * (100,000 * 4 + 10,000 * 26) \\ \text{or} \\ \text{Size} &= 308 * (400,000 + 260,000) \\ \text{or} \\ \text{Size} &= 203,280,000 \text{ bytes} \end{aligned}$$

So for the above two systems, we should expect to have a database size of approximately 200 Megabytes. Of course, this will vary according to how many files are actually backed up.

Below are some statistics for a MySQL database containing Job records for five Clients beginning September 2001 through May 2002 (8.5 months) and File records for the last 80 days. (Older File records have been pruned). For these systems, only the user files and system files that change are backed up. The core part of the system is assumed to be easily reloaded from the Red Hat rpms.

In the list below, the files (corresponding to Bacula Tables) with the extension .MYD contain the data records whereas files with the extension .MYI contain indexes.

You will note that the File records (containing the file attributes) make up the large bulk of the number of records as well as the space used (459 Mega Bytes including the indexes). As a consequence, the most important Retention period will be the **File Retention** period. A quick calculation shows that for each File that is saved, the database grows by approximately 150 bytes.

Size in Bytes	Records	File
168	5	Client.MYD
3,072		Client.MYI
344,394,684	3,080,191	File.MYD
115,280,896		File.MYI

2,590,316	106,902	Filename.MYD
3,026,944		Filename.MYI
184	4	FileSet.MYD
2,048		FileSet.MYI
49,062	1,326	JobMedia.MYD
30,720		JobMedia.MYI
141,752	1,378	Job.MYD
13,312		Job.MYI
1,004	11	Media.MYD
3,072		Media.MYI
1,299,512	22,233	Path.MYD
581,632		Path.MYI
36	1	Pool.MYD
3,072		Pool.MYI
5	1	Version.MYD
1,024		Version.MYI

This database has a total size of approximately 450 Megabytes.

If we were using SQLite, the determination of the total database size would be much easier since it is a single file, but we would have less insight to the size of the individual tables as we have in this case.

Note, SQLite databases may be as much as 50% larger than MySQL databases due to the fact that all data is stored as ASCII strings. That is even binary integers are stored as ASCII strings, and this seems to increase the space needed.

## Chapter 2

# Installing and Configuring MySQL

### 2.1 Installing and Configuring MySQL – Phase I

If you use the `./configure --with-mysql=mysql-directory` statement for configuring **Bacula**, you will need MySQL version 4.1 or later installed in the **mysql-directory**. If you are using one of the new modes such as ANSI/ISO compatibility, you may experience problems.

If MySQL is installed in the standard system location, you need only enter `--with-mysql` since the configure program will search all the standard locations. If you install MySQL in your home directory or some other non-standard directory, you will need to provide the full path to it.

Installing and Configuring MySQL is not difficult but can be confusing the first time. As a consequence, below, we list the steps that we used to install it on our machines. Please note that our configuration leaves MySQL without any user passwords. This may be an undesirable situation if you have other users on your system.

The notes below describe how to build MySQL from the source tar files. If you have a pre-installed MySQL, you can return to complete the installation of Bacula, then come back to Phase II of the MySQL installation. If you wish to install MySQL from rpms, you will probably need to install the following:

```
mysql-<version>.rpm  
mysql-server-<version>.rpm  
mysql-devel-<version>.rpm
```

The names of the packages may vary from distribution to distribution. It is important to have the devel package loaded as it contains the libraries and header files necessary to build Bacula. There may be additional packages that are required to install the above, for example, `zlib` and `openssl`.

Once these packages are installed, you will be able to build Bacula (using the files installed with the `mysql` package, then run MySQL using the files installed with `mysql-server`). If you have installed MySQL by rpms, please skip Phase I below, and return to complete the installation of Bacula, then come back to Phase II of the MySQL installation when indicated to do so.

Beginning with Bacula version 1.31, the thread safe version of the MySQL client library is used, and hence you should add the `--enable-thread-safe-client` option to the `./configure` as shown below:

1. Download MySQL source code from [www.mysql.com/downloads](http://www.mysql.com/downloads)
2. Detar it with something like:

```
tar xvfz mysql-filename
```

Note, the above command requires GNU tar. If you do not have GNU tar, a command such as:

```
zcat mysql-filename — tar xvf -
```

will probably accomplish the same thing.

3. `cd mysql-source-directory`

where you replace **mysql-source-directory** with the directory name where you put the MySQL source code.

4. `./configure --enable-thread-safe-client --prefix=mysql-directory`

where you replace **mysql-directory** with the directory name where you want to install mysql. Normally for system wide use this is `/usr/local/mysql`. In my case, I use `~/kern/mysql`.

5. `make`

This takes a bit of time.

6. `make install`

This will put all the necessary binaries, libraries and support files into the **mysql-directory** that you specified above.

7. `./scripts/mysql_install_db`

This will create the necessary MySQL databases for controlling user access. Note, this script can also be found in the **bin** directory in the installation directory

The MySQL client library **mysqlclient** requires the gzip compression library **libz.a** or **libz.so**. If you are using rpm packages, these libraries are in the **libz-devel** package. On Debian systems, you will need to load the **zlib1g-dev** package. If you are not using rpms or debs, you will need to find the appropriate package for your system.

At this point, you should return to completing the installation of **Bacula**. Later after Bacula is installed, come back to this chapter to complete the installation. Please note, the installation files used in the second phase of the MySQL installation are created during the Bacula Installation.

## 2.2 Installing and Configuring MySQL – Phase II

At this point, you should have built and installed MySQL, or already have a running MySQL, and you should have configured, built and installed **Bacula**. If not, please complete these items before proceeding.

Please note that the `./configure` used to build **Bacula** will need to include `--with-mysql=mysql-directory`, where **mysql-directory** is the directory name that you specified on the `./configure` command for configuring MySQL. This is needed so that Bacula can find the necessary include headers and library files for interfacing to MySQL.

**Bacula** will install scripts for manipulating the database (create, delete, make tables etc) into the main installation directory. These files will be of the form `*_bacula_*` (e.g. `create_bacula_database`). These files are also available in the `<bacula-src>/src/cats` directory after running `./configure`. If you inspect `create_bacula_database`, you will see that it calls `create_mysql_database`. The `*_bacula_*` files are provided for convenience. It doesn't matter what database you have chosen; `create_bacula_database` will always create your database.

Now you will create the Bacula MySQL database and the tables that Bacula uses.

1. Start **mysql**. You might want to use the **startmysql** script provided in the Bacula release.
2. `cd <install-directory>` This directory contains the Bacula catalog interface routines.
3. `./grant_mysql_privileges` This script creates unrestricted access rights for the user **bacula**. You may want to modify it to suit your situation. Please note that none of the userids, including root, are password protected. If you need more security, please assign a password to the root user and to bacula. The program **mysqladmin** can be used for this.

4. `./create_mysql_database` This script creates the MySQL **bacula** database. The databases you create as well as the access databases will be located in `<install-dir>/var/` in a subdirectory with the name of the database, where `<install-dir>` is the directory name that you specified on the `--prefix` option. This can be important to know if you want to make a special backup of the Bacula database or to check its size.
5. `./make_mysql_tables` This script creates the MySQL tables used by **Bacula**.

Each of the three scripts (`grant_mysql_privileges`, `create_mysql_database` and `make_mysql_tables`) allows the addition of a command line argument. This can be useful for specifying the user and or password. For example, you might need to add `-u root` to the command line to have sufficient privilege to create the Bacula tables.

To take a closer look at the access privileges that you have setup with the above, you can do:

```
mysql-directory/bin/mysql -u root mysql
select * from user;
```

## 2.3 Re-initializing the Catalog Database

After you have done some initial testing with **Bacula**, you will probably want to re-initialize the catalog database and throw away all the test Jobs that you ran. To do so, you can do the following:

```
cd <install-directory>
./drop_mysql_tables
./make_mysql_tables
```

Please note that all information in the database will be lost and you will be starting from scratch. If you have written on any Volumes, you must write an end of file mark on the volume so that Bacula can reuse it. Do so with:

```
(stop Bacula or unmount the drive)
mt -f /dev/nst0 rewind
mt -f /dev/nst0 weof
```

Where you should replace `/dev/nst0` with the appropriate tape drive device name for your machine.

## 2.4 Linking Bacula with MySQL

After configuring Bacula with

```
./configure --enable-thread-safe-client --prefix=<mysql-directory>
```

where `<mysql-directory>` is in my case `/home/kern/mysql`, you may have to configure the loader so that it can find the MySQL shared libraries. If you have previously followed this procedure and later add the `--enable-thread-safe-client` options, you will need to rerun the `ldconfig` program shown below. If you put MySQL in a standard place such as `/usr/lib` or `/usr/local/lib` this will not be necessary, but in my case it is. The description that follows is Linux specific. For other operating systems, please consult your manuals on how to do the same thing:

First edit: `/etc/ld.so.conf` and add a new line to the end of the file with the name of the `mysql-directory`. In my case, it is:

```
/home/kern/mysql/lib/mysql
```

then rebuild the loader's cache with:

```
/sbin/ldconfig
```

If you upgrade to a new version of **MySQL**, the shared library names will probably change, and you must re-run the `/sbin/ldconfig` command so that the runtime loader can find them.

Alternatively, your system may have a loader environment variable that can be set. For example, on a Solaris system where I do not have root permission, I use:

```
LD_LIBRARY_PATH=/home/kern/mysql/lib/mysql
```

Finally, if you have encryption enabled in MySQL, you may need to add `-lssl -lcrypto` to the link. In that case, you can either export the appropriate LDFLAGS definition, or alternatively, you can include them directly on the `./configure` line as in:

```
LDFLAGS="-lssl -lcrypto" \  
./configure \  
<your-options>
```

## 2.5 Installing MySQL from RPMs

If you are installing MySQL from RPMs, you will need to install both the MySQL binaries and the client libraries. The client libraries are usually found in a devel package, so you must install:

```
mysql  
mysql-devel
```

This will be the same with most other package managers too.

## 2.6 Upgrading MySQL

If you upgrade MySQL, you must reconfigure, rebuild, and re-install Bacula otherwise you are likely to get bizarre failures. If you install from rpms and you upgrade MySQL, you must also rebuild Bacula. You can do so by rebuilding from the source rpm. To do so, you may need to modify the `bacula.spec` file to account for the new MySQL version.

## Chapter 3

# Installing and Configuring PostgreSQL

If you are considering using PostgreSQL, you should be aware of their philosophy of upgrades, which could be destabilizing for a production shop. Basically at every major version upgrade, you are required to dump your database in an ASCII format, do the upgrade, and then reload your database (or databases). This is because they frequently update the "data format" from version to version, and they supply no tools to automatically do the conversion. If you forget to do the ASCII dump, your database may become totally useless because none of the new tools can access it due to the format change, and the PostgreSQL server will not be able to start.

If you are building PostgreSQL from source, please be sure to add the `--enable-thread-safety` option when doing the `./configure` for PostgreSQL.

### 3.1 Installing PostgreSQL

If you use the `./configure --with-postgresql=PostgreSQL-Directory` statement for configuring **Bacula**, you will need PostgreSQL version 7.4 or later installed. NOTE! PostgreSQL versions earlier than 7.4 do not work with Bacula. If PostgreSQL is installed in the standard system location, you need only enter `--with-postgresql` since the configure program will search all the standard locations. If you install PostgreSQL in your home directory or some other non-standard directory, you will need to provide the full path with the `--with-postgresql` option.

Installing and configuring PostgreSQL is not difficult but can be confusing the first time. If you prefer, you may want to use a package provided by your chosen operating system. Binary packages are available on most PostgreSQL mirrors.

If you prefer to install from source, we recommend following the instructions found in the PostgreSQL documentation.

If you are using FreeBSD, this FreeBSD Diary article will be useful. Even if you are not using FreeBSD, the article will contain useful configuration and setup information.

If you configure the Batch Insert code in Bacula (attribute inserts are 10 times faster), you **must** be using a PostgreSQL that was built with the `--enable-thread-safety` option, otherwise you will get data corruption. Most major Linux distros have thread safety turned on, but it is better to check. One way is to see if the PostgreSQL library that Bacula will be linked against references pthreads. This can be done with a command such as:

```
nm /usr/lib/libpq.a | grep pthread_mutex_lock
```

The above command should print a line that looks like:

```
U pthread_mutex_lock
```

if does, then everything is OK. If it prints nothing, do not enable batch inserts when building Bacula.

After installing PostgreSQL, you should return to completing the installation of **Bacula**. Later, after Bacula is installed, come back to this chapter to complete the installation. Please note, the installation files used in the second phase of the PostgreSQL installation are created during the Bacula Installation. You must still come back to complete the second phase of the PostgreSQL installation even if you installed binaries (e.g. rpm, deb, ...).

## 3.2 Configuring PostgreSQL

At this point, you should have built and installed PostgreSQL, or already have a running PostgreSQL, and you should have configured, built and installed **Bacula**. If not, please complete these items before proceeding.

Please note that the `./configure` used to build **Bacula** will need to include `--with-postgresql=PostgreSQL-directory`, where `PostgreSQL-directory` is the directory name that you specified on the `./configure` command for configuring PostgreSQL (if you didn't specify a directory or PostgreSQL is installed in a default location, you do not need to specify the directory). This is needed so that Bacula can find the necessary include headers and library files for interfacing to PostgreSQL.

**Bacula** will install scripts for manipulating the database (create, delete, make tables etc) into the main installation directory. These files will be of the form `*_bacula_*` (e.g. `create_bacula_database`). These files are also available in the `<bacula-src>/src/cats` directory after running `./configure`. If you inspect `create_bacula_database`, you will see that it calls `create_postgresql_database`. The `*_bacula_*` files are provided for convenience. It doesn't matter what database you have chosen; `create_bacula_database` will always create your database.

Now you will create the Bacula PostgreSQL database and the tables that Bacula uses. These instructions assume that you already have PostgreSQL running. You will need to perform these steps as a user that is able to create new databases. This can be the PostgreSQL user (on most systems, this is the `pgsql` user).

1. `cd <install-directory>`

This directory contains the Bacula catalog interface routines.

2. `./create_bacula_database`

This script creates the PostgreSQL **bacula** database. Before running this command, you should carefully think about what encoding sequence you want for the text fields (paths, files, ...). Ideally, the encoding should be set to UTF8. However, many Unix systems have filenames that are not encoded in UTF8, either because you have not set UTF8 as your default character set or because you have imported files from elsewhere (e.g. MacOS X). For this reason, Bacula uses `SQL_ASCII` as the default encoding. If you want to change this, please modify the script before running it, but be forewarned that Bacula backups will fail if PostgreSQL finds any non-UTF8 sequences.

If running the script fails, it is probably because the database is owned by a user other than yourself. On many systems, the database owner is **pgsql** and on others such as Red Hat and Fedora it is **postgres**. You can find out which it is by examining your `/etc/passwd` file. To create a new user under either your name or with say the name **bacula**, you can do the following:

```
su
(enter root password)
su postgres (or postgres)
createuser kern (or perhaps bacula)
Shall the new user be allowed to create databases? (y/n) y
Shall the new user be allowed to create more new users? (y/n) (choose
what you want)
exit
```

At this point, you should be able to execute the `./create_bacula_database` command.

3. `./make_bacula_tables`

This script creates the PostgreSQL tables used by **Bacula**.

4. `./grant_bacula_privileges`

This script creates the database user **bacula** with restricted access rights. You may want to modify it to suit your situation. Please note that this database is not password protected.

Each of the three scripts (`create_bacula_database`, `make_bacula_tables`, and `grant_bacula_privileges`) allows the addition of a command line argument. This can be useful for specifying the user name. For example, you might need to add **-h hostname** to the command line to specify a remote database server.

To take a closer look at the access privileges that you have setup with the above, you can do:

```
PostgreSQL-directory/bin/psql --command \dp bacula
```

Also, I had an authorization problem with the password. In the end, I had to modify my `pg_hba.conf` file (in `/var/lib/pgsql/data` on my machine) from:

```
local all all ident sameuser
to
local all all trust
```

This solved the problem for me, but it is not always a good thing to do from a security standpoint. However, it allowed me to run my regression scripts without having a password.

A more secure way to perform database authentication is with md5 password hashes. Begin by editing the `pg_hba.conf` file, and just prior the the existing “local” and “host” lines, add the line:

```
local bacula bacula md5
```

and restart the Postgres database server (frequently, this can be done using `"/etc/init.d/postgresql restart"` or `"service postgresql restart"`) to put this new authentication rule into effect.

Next, become the Postgres administrator, `postgres`, either by logging on as the `postgres` user, or by using `su` to become root and then using `su - postgres` to become `postgres`. Add a password to the bacula database for the bacula user using:

```
\$ psql bacula
bacula=# alter user bacula with password 'secret';
ALTER USER
bacula=# \q
```

You'll have to add this password to two locations in the `bacula-dir.conf` file: once to the Catalog resource and once to the `RunBeforeJob` entry in the BackupCatalog Job resource. With the password in place, these two lines should look something like:

```
dbname = bacula; user = bacula; password = "secret"
... and ...
# WARNING!!! Passing the password via the command line is insecure.
# see comments in make_catalog_backup for details.
RunBeforeJob = "/etc/make_catalog_backup bacula bacula secret"
```

Naturally, you should choose your own significantly more random password, and ensure that the `bacula-dir.conf` file containing this password is readable only by the root.

Even with the files containing the database password properly restricted, there is still a security problem with this approach: on some platforms, the environment variable that is used to supply the password to Postgres is available to all users of the local system. To eliminate this problem, the Postgres team have deprecated the use of the environment variable password-passing mechanism and recommend the use of a .pgpass file instead. To use this mechanism, create a file named .pgpass containing the single line:

```
localhost:5432:bacula:bacula:secret
```

This file should be copied into the home directory of all accounts that will need to gain access to the database: typically, root, bacula, and any users who will make use of any of the console programs. The files must then have the owner and group set to match the user (so root:root for the copy in root, and so on), and the mode set to 600, limiting access to the owner of the file.

### 3.3 Re-initializing the Catalog Database

After you have done some initial testing with **Bacula**, you will probably want to re-initialize the catalog database and throw away all the test Jobs that you ran. To do so, you can do the following:

```
cd <install-directory>
./drop_bacula_tables
./make_bacula_tables
./grant_bacula_privileges
```

Please note that all information in the database will be lost and you will be starting from scratch. If you have written on any Volumes, you must write an end of file mark on the volume so that Bacula can reuse it. Do so with:

```
(stop Bacula or unmount the drive)
mt -f /dev/nst0 rewind
mt -f /dev/nst0 weof
```

Where you should replace `/dev/nst0` with the appropriate tape drive device name for your machine.

### 3.4 Installing PostgreSQL from RPMs

If you are installing PostgreSQL from RPMs, you will need to install both the PostgreSQL binaries and the client libraries. The client libraries are usually found in a devel package, so you must install:

```
postgresql
postgresql-devel
postgresql-server
postgresql-libs
```

These will be similar with most other package managers too. After installing from rpms, you will still need to run the scripts that set up the database and create the tables as described above.

### 3.5 Converting from MySQL to PostgreSQL

The conversion procedure presented here was worked out by Norm Dressler <ndressler at dinmar dot com>

This process was tested using the following software versions:

- Linux Mandrake 10/Kernel 2.4.22-10 SMP
- Mysql Ver 12.21 Distrib 4.0.15, for mandrake-linux-gnu (i586)
- PostgreSQL 7.3.4
- Bacula 1.34.5

WARNING: Always as a precaution, take a complete backup of your databases before proceeding with this process!

1. Shutdown bacula (cd /etc/bacula;./bacula stop)
2. Run the following command to dump your Mysql database:

```
mysqldump -f -t -n >bacula-backup.dmp
```

3. Make a backup of your /etc/bacula directory (but leave the original in place).
4. Go to your Bacula source directory and rebuild it to include PostgreSQL support rather than Mysql support. Check the config.log file for your original configure command and replace enable-mysql with enable-postgresql.
5. Recompile Bacula with a make and if everything compiles completely, perform a make install.
6. Shutdown Mysql.
7. Start PostgreSQL on your system.
8. Create a bacula user in Postgres with the createuser command. Depending on your Postgres install, you may have to SU to the user who has privileges to create a user.
9. Verify your pg\_hba.conf file contains sufficient permissions to allow bacula to access the server. Mine has the following since it's on a secure network:

```
local all all trust
host all all 127.0.0.1 255.255.255.255 trust

NOTE: you should restart your postgres server if you
      made changes
```

10. Change into the /etc/bacula directory and prepare the database and tables with the following commands:

```
./create_postgresql_database
./make_postgresql_tables
./grant_postgresql_privileges
```

11. Verify you have access to the database:

```
psql -Ubacula bacula
```

You should not get any errors.

12. Load your database from the Mysql database dump with:

```
psql -Ubacula bacula <bacula-backup.dmp>
```

13. Resequence your tables with the following commands:

```
psql -Ubacula bacula

SELECT SETVAL('basefiles_baseid_seq', (SELECT
MAX(baseid) FROM basefiles));
SELECT SETVAL('client_clientid_seq', (SELECT
MAX(clientid) FROM client));
SELECT SETVAL('file_fileid_seq', (SELECT MAX(fileid)
FROM file));
SELECT SETVAL('filename_filenameid_seq', (SELECT
MAX(filenameid) FROM filename));

SELECT SETVAL('fileset_filesetid_seq', (SELECT
MAX(filesetid) FROM fileset));

SELECT SETVAL('job_jobid_seq', (SELECT MAX(jobid) FROM job));
SELECT SETVAL('jobmedia_jobmediaid_seq', (SELECT
MAX(jobmediaid) FROM jobmedia));
SELECT SETVAL('media_mediaid_seq', (SELECT MAX(mediaid) FROM media));
SELECT SETVAL('path_pathid_seq', (SELECT MAX(pathid) FROM path));

SELECT SETVAL('pool_poolid_seq', (SELECT MAX(poolid) FROM pool));
```

14. At this point, start up Bacula, verify your volume library and perform a test backup to make sure everything is working properly.

## 3.6 Upgrading PostgreSQL

If you upgrade PostgreSQL, you must reconfigure, rebuild, and re-install Bacula otherwise you are likely to get bizarre failures. If you to modify the bacula.spec file to account for the new PostgreSQL version. You can do so by rebuilding from the source rpm. To do so, you may need install from rpms and you upgrade PostgreSQL, you must also rebuild Bacula.

## 3.7 Tuning PostgreSQL

If you despool attributes for many jobs at the same time, you can tune the sequence object for the `FileId` field.

```
psql -Ubacula bacula

ALTER SEQUENCE file_fileid_seq CACHE 1000;
```

## 3.8 Credits

Many thanks to Dan Langille for writing the PostgreSQL driver. This will surely become the most popular database that Bacula supports.

## Chapter 4

# Installing and Configuring SQLite

Please note that SQLite both versions 2 and 3 are not network enabled, which means that they must be linked into the Director rather than accessed by the network as MySQL and PostgreSQL are. This has two consequences:

1. SQLite cannot be used in the **bweb** web GUI package.
2. If you use SQLite, and your Storage daemon is not on the same machine as your Director, you will need to transfer your database to the Storage daemon's machine before you can use any of the SD tools such as **bscan**, ...

### 4.1 Installing and Configuring SQLite – Phase I

If you use the `./configure --with-sqlite` statement for configuring **Bacula**, you will need SQLite version 2.8.16 or later installed. Our standard location (for the moment) for SQLite is in the dependency package **depkgs/sqlite-2.8.16**. Please note that the version will be updated as new versions are available and tested.

Installing and Configuring is quite easy.

1. Download the Bacula dependency packages
2. Detar it with something like:  
**tar xvfz depkgs.tar.gz**

Note, the above command requires GNU tar. If you do not have GNU tar, a command such as:

**zcat depkgs.tar.gz — tar xvf -**

will probably accomplish the same thing.

3. **cd depkgs**
4. **make sqlite**

Please note that the `./configure` used to build **Bacula** will need to include `--with-sqlite` or `--with-sqlite3` depending on which version of SQLite you are using. You should not use the `--enable-batch-insert` configuration parameter for Bacula if you are using SQLite version 2 as it is probably not thread safe. If you are using SQLite version 3, you may use the `--enable-batch-insert` configuration option with Bacula, but when building SQLite3 you MUST configure it with `--enable-threadsafe` and `--enable-cross-thread-connections`.

By default, SQLite3 is now run with **PRAGMA synchronous=OFF** this increases the speed by more than 30 times, but it also increases the possibility of a corrupted database if your server crashes (power failure or kernel bug). If you want more security, you can change the PRAGMA that is used in the file `src/version.h`.

At this point, you should return to completing the installation of **Bacula**.

## 4.2 Installing and Configuring SQLite – Phase II

This phase is done **after** you have run the `./configure` command to configure **Bacula**.

**Bacula** will install scripts for manipulating the database (create, delete, make tables etc) into the main installation directory. These files will be of the form `*_bacula_*` (e.g. `create_bacula_database`). These files are also available in the `<bacula-src>/src/cats` directory after running `./configure`. If you inspect `create_bacula_database`, you will see that it calls `create_sqlite_database`. The `*_bacula_*` files are provided for convenience. It doesn't matter what database you have chosen; `create_bacula_database` will always create your database.

At this point, you can create the SQLite database and tables:

1. `cd <install-directory>`

This directory contains the Bacula catalog interface routines.

2. `./make_sqlite_tables`

This script creates the SQLite database as well as the tables used by **Bacula**. This script will be automatically setup by the `./configure` program to create a database named `bacula.db` in **Bacula's** working directory.

## 4.3 Linking Bacula with SQLite

If you have followed the above steps, this will all happen automatically and the SQLite libraries will be linked into **Bacula**.

## 4.4 Testing SQLite

We have much less "production" experience using SQLite than using MySQL. SQLite has performed flawlessly for us in all our testing. However, several users have reported corrupted databases while using SQLite. For that reason, we do not recommend it for production use.

If Bacula crashes with the following type of error when it is started:

```
Using default Catalog name=MyCatalog DB=bacula
Could not open database "bacula".
sqlite.c:151 Unable to open Database=/var/lib/bacula/bacula.db.
ERR=malformed database schema - unable to open a temporary database file
for storing temporary tables
```

this is most likely caused by the fact that some versions of SQLite attempt to create a temporary file in the current directory. If that fails, because Bacula does not have write permission on the current directory, then you may get this err. The solution is to start Bacula in a current directory where it has write permission.

## 4.5 Re-initializing the Catalog Database

After you have done some initial testing with **Bacula**, you will probably want to re-initialize the catalog database and throw away all the test Jobs that you ran. To do so, you can do the following:

```
cd <install-directory>
./drop_sqlite_tables
./make_sqlite_tables
```

Please note that all information in the database will be lost and you will be starting from scratch. If you have written on any Volumes, you must write an end of file mark on the volume so that Bacula can reuse it. Do so with:

```
(stop Bacula or unmount the drive)
mt -f /dev/nst0 rewind
mt -f /dev/nst0 weof
```

Where you should replace `/dev/nst0` with the appropriate tape drive device name for your machine.



## Chapter 5

# The internal database is not supported, please do not use it.

### 5.1 Internal Bacula Database

Previously it was intended to be used primarily by Bacula developers for testing; although SQLite is also a good choice for this. We do not recommend its use in general.

This database is simplistic in that it consists entirely of Bacula's internal structures appended sequentially to a file. Consequently, it is in most cases inappropriate for sites with many clients or systems with large numbers of files, or long-term production environments.

Below, you will find a table comparing the features available with SQLite and MySQL and with the internal Bacula database. At the current time, you cannot dynamically switch from one to the other, but must rebuild the Bacula source code. If you wish to experiment with both, it is possible to build both versions of Bacula and install them into separate directories.

Feature	SQLite or MySQL	Bacula
Job Record	Yes	Yes
Media Record	Yes	Yes
FileName Record	Yes	No
File Record	Yes	No
FileSet Record	Yes	Yes
Pool Record	Yes	Yes
Client Record	Yes	Yes
JobMedia Record	Yes	Yes
List Job Records	Yes	Yes
List Media Records	Yes	Yes
List Pool Records	Yes	Yes
List JobMedia Records	Yes	Yes
Delete Pool Record	Yes	Yes
Delete Media Record	Yes	Yes
Update Pool Record	Yes	Yes
Implement Verify	Yes	No
MD5 Signatures	Yes	No

In addition, since there is no SQL available, the Console commands: **sqlquery**, **query**, **retention**, and any other command that directly uses SQL are not available with the Internal database.



## Chapter 6

# GNU Free Documentation License

Version 1.2, November 2002

Copyright ©2000,2001,2002 Free Software Foundation, Inc.

51 Franklin St, Fifth Floor, Boston, MA 02110-1301 USA

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

### Preamble

The purpose of this License is to make a manual, textbook, or other functional and useful document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondly, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

## 1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "**Document**", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "**you**". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A "**Modified Version**" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "**Secondary Section**" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject

(or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The **”Invariant Sections”** are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The **”Cover Texts”** are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A **”Transparent”** copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not **”Transparent”** is called **”Opaque”**.

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The **”Title Page”** means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, **”Title Page”** means the text near the most prominent appearance of the work’s title, preceding the beginning of the body of the text.

A section **”Entitled XYZ”** means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as **”Acknowledgements”**, **”Dedications”**, **”Endorsements”**, or **”History”**.) To **”Preserve the Title”** of such a section when you modify the Document means that it remains a section **”Entitled XYZ”** according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

## 2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

## 3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

## 4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

- A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.
- B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.
- C. State on the Title page the name of the publisher of the Modified Version, as the publisher.
- D. Preserve all the copyright notices of the Document.
- E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.
- F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.
- G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.
- H. Include an unaltered copy of this License.
- I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.

- J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.
- K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.
- L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.
- M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.
- N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.
- O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties—for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

## 5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements".

## 6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

## 7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

## 8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

## 9. TERMINATION

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

## 10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See <http://www.gnu.org/copyleft/>.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

## ADDENDUM: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

Copyright ©YEAR YOUR NAME. Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License".

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the "with...Texts." line with this:

with the Invariant Sections being LIST THEIR TITLES, with the Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST.

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.

# General Index

- AutoPrune , 6
- Backing Up Third Party Databases , 13
- Backing Up Your Bacula Database , 11
- Backing Up Your Bacula Database - Security Considerations , 12
- Catalog Maintenance , 5
- Compacting Your MySQL Database , 6
- Compacting Your PostgreSQL Database , 10
- Compacting Your SQLite Database , 11
- Converting from MySQL to PostgreSQL , 22
- Credits , 24
- Database
  - Backing Up Your Bacula , 11
  - Backing Up Your Bacula Database - Security Considerations , 12
  - Compacting Your MySQL , 6
  - Compacting Your PostgreSQL , 10
  - Compacting Your SQLite , 11
  - Internal Bacula , 29
  - MySQL Server Has Gone Away, 8
  - MySQL Table is Full, 7
  - Re-initializing the Catalog , 17, 22, 26
  - Repairing Your MySQL , 7
  - Repairing Your PostgreSQL , 8
- Database Performance Issues, 8
- Database Performance Issues Indexes, 9
- Database Size , 13
- Databases
  - Backing Up Third Party , 13
- File Retention , 5
- GNU Free Documentation License, 31
- Installing and Configuring MySQL , 15
- Installing and Configuring MySQL – Phase I , 15
- Installing and Configuring MySQL – Phase II , 16
- Installing and Configuring PostgreSQL , 19
- Installing and Configuring SQLite , 25
- Installing and Configuring SQLite – Phase I , 25
- Installing and Configuring SQLite – Phase II , 26
- Installing MySQL from RPMs, 18
- Installing PostgreSQL from RPMs, 22
- Internal Bacula Database , 29
- Job Retention , 5
- License
  - GNU Free Documentation, 31
- Linking Bacula with MySQL , 17
- Linking Bacula with SQLite , 26
- Maintenance
  - Catalog , 5
- Migrating from SQLite to MySQL or PostgreSQL, 11
- MySQL
  - Installing and Configuring , 15
  - Installing from RPMs, 18
  - Linking Bacula with , 17
  - Migrating from SQLite to , 11
- MySQL Server Has Gone Away, 8
- MySQL Table is Full, 7
- Performance
  - Database, 8, 9
- Periods
  - Setting Retention , 5
- Phase I
  - Installing and Configuring MySQL – , 15
  - Installing and Configuring SQLite – , 25
- Phase II
  - Installing and Configuring MySQL – , 16
  - Installing and Configuring SQLite – , 26
- PostgreSQL
  - Configuring PostgreSQL – , 20
  - Converting from MySQL to , 22
  - Installing , 19
  - Installing and Configuring , 19
  - Installing from RPMs, 22
- Re-initializing the Catalog Database , 17, 22, 26
- Repairing Your MySQL Database , 7
- Repairing Your PostgreSQL Database , 8
- Setting Retention Periods , 5
- Size
  - Database , 13
- SQLite
  - Installing and Configuring , 25
  - Linking Bacula with , 26
  - Testing , 26
- Testing SQLite , 26
- The internal database is not supported, please do not use it. , 29
- Tuning, 24
- Upgrading, 17–19, 24
  - MySQL , 18

PostgreSQL , 24  
Upgrading MySQL , 18  
Upgrading PostgreSQL , 24  
Use it  
The internal database is not supported please  
do not , 29