SharDB Installation Guide

by
Ron & Andrea Rennick
http://wpebooks.com

Table of Contents

Introduction

The following instructions are for setting up the freely available SharDB plugin, located at http://wordpress.org/extend/plugins/shardb/. This plugin allows those running a WordPress network (multisite) to spread the data across 16, 256 or 4096 databases plus a database for global tables.

Initially, you will create those empty databases, then use the included migration script to copy all the tables from your original single database into new ones. The plugin includes a new database class that uses those databases.

This guide describes the process of implementing using 16 databases. See Appendix A for notes on the on implementing 256 or 4096 databases.

Setting up the databases

Before we run the migration script or the plugin code, we need to create all the databases to house the data.

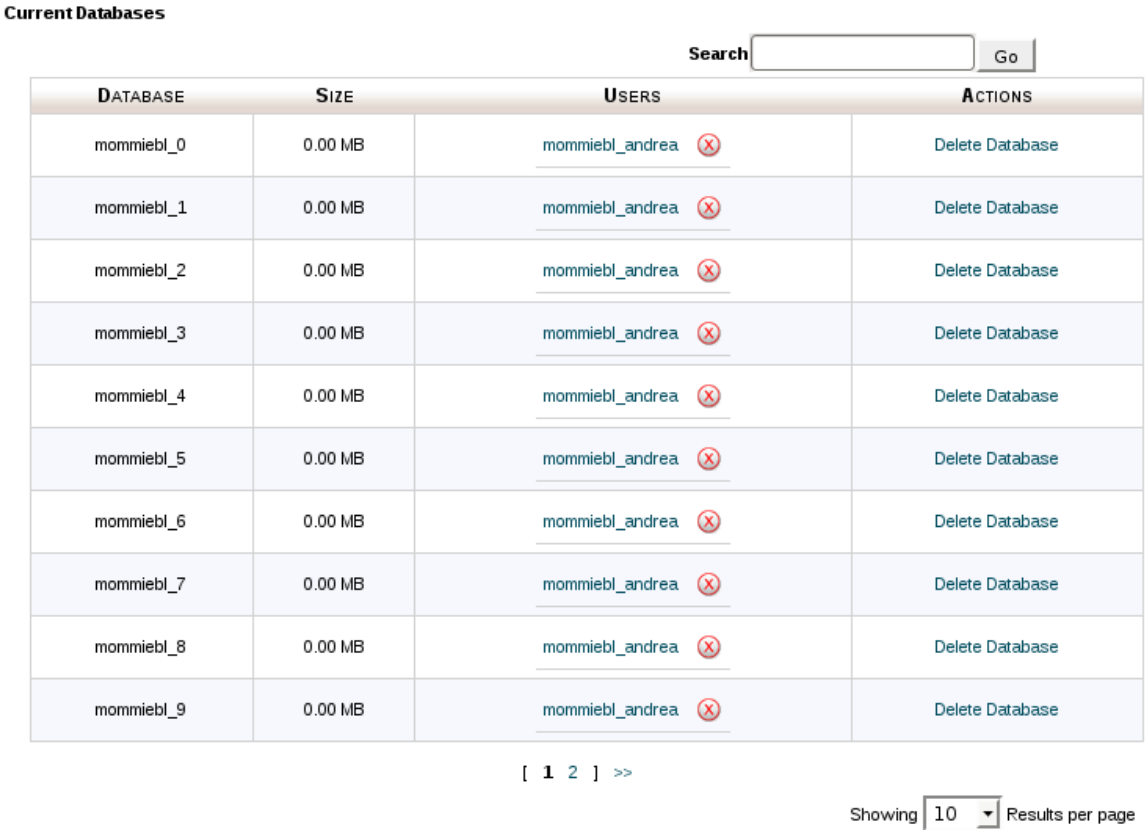
Login to your webhost control panel, and look for a database related icon. In cPanel, this has a MySQL logo and is called MySQL Databases.

1. Create 16 databases, named 0-9, a-f and “global”. You are free to call the databases anything you like, this is just an example for ease of use. The global table **must** be named “global”.

    - If you are using cPanel each database is prefixed with your webhost user account name. (See my example screenshot.)
    - For other server configurations, ensure the database names all have the same prefix.
    - In either case, it is important that the a-f and global be lowercase letters. **MySQL database names are case sensitive.**

2. Add the same database user to all databases we just created, with ALL privileges. (This is in the same menu screen.)

In the screenshot below, you can see the list of empty databases I created, with a user attached to each.



Command line

If you have SSH access, you can also create the databases via the terminal by using the following commands:

```
$ for D in 0 1 2 3 4 5 6 7 8 9 a b c d e f global; do \
echo CREATE DATABASE your_prefix_$D';' >> db-create.sql; \
echo GRANT ALL ON your_prefix_$D.* TO your_user';' >> db-
create.sql; \
done
```

Check that the file contains the correct SQL statements with

```
$ cat db-create.sql
CREATE DATABASE your_prefix_0;
GRANT ALL ON your_prefix_0.* TO your_user;
CREATE DATABASE your_prefix_1;
GRANT ALL ON your_prefix_1.* TO your_user;
CREATE DATABASE your_prefix_2;
GRANT ALL ON your_prefix_2.* TO your_user;
CREATE DATABASE your_prefix_3;
GRANT ALL ON your_prefix_3.* TO your_user;
CREATE DATABASE your_prefix_4;
GRANT ALL ON your_prefix_4.* TO your_user;
```

```
CREATE DATABASE your_prefix_5;
GRANT ALL ON your_prefix_5.* TO your_user;
CREATE DATABASE your_prefix_6;
GRANT ALL ON your_prefix_6.* TO your_user;
CREATE DATABASE your_prefix_7;
GRANT ALL ON your_prefix_7.* TO your_user;
CREATE DATABASE your_prefix_8;
GRANT ALL ON your_prefix_8.* TO your_user;
CREATE DATABASE your_prefix_9;
GRANT ALL ON your_prefix_9.* TO your_user;
CREATE DATABASE your_prefix_a;
GRANT ALL ON your_prefix_a.* TO your_user;
CREATE DATABASE your_prefix_b;
GRANT ALL ON your_prefix_b.* TO your_user;
CREATE DATABASE your_prefix_c;
GRANT ALL ON your_prefix_c.* TO your_user;
CREATE DATABASE your_prefix_d;
GRANT ALL ON your_prefix_d.* TO your_user;
CREATE DATABASE your_prefix_e;
GRANT ALL ON your_prefix_e.* TO your_user;
CREATE DATABASE your_prefix_f;
GRANT ALL ON your_prefix_f.* TO your_user;
CREATE DATABASE your_prefix_global;
GRANT ALL ON your_prefix_global.* TO your_user;
```

If you made an error, delete db-create.sql, correct the script above and re-run it.

Once you have the correct SQL statements, you can execute those with

```
$ mysql –u admin_user –padmin_pass < db-create.sql
```

Verify that the databases were created correctly with

```
$ mysql –u admin_user –padmin_pass
mysql>show databases;
mysql>exit
```

Installing the plugin

The SharDB plugin contains 3 files that will be placed in the different folders on the web server. You'll also be editing a couple files. I unzipped the plugin package in a spare folder on the webserver, made my edits, then moved them in place. You may be more comfortable unzipping the plugin locally, editing the files, then FTPing the files in place. Go with whatever you're more comfortable with.

1. Open up the db-settings.php file. Note a large section near the top that has been commented out. This area contains all the possible settings. In our example, we'll be using the bare minimum to get up and running.

2. Instead of uncommenting the commands I wanted to use, I copied the lines I wanted and pasted them up to the configuration area, like this:

```
/* Add your configuration here */
$shardb_hash_length = 1;
$shardb_prefix = 'mommiebl_';
$shardb_dataset = 'mb';
```

   The hash length of **1** means I used one character naming scheme for each database. (0 through 9 and a through f). The prefix is the webhost username I got from my host. Dataset can be anything, it's used internally like a secret key.

3. Save the file, and put in the same folder where your wp-config.php file is.
4. Edit wp-config.php to tell the SharDB migration script what your new database configuration will be. After the **`define('DB_COLLATE',`** line, add this line:

```
require( 'db-settings.php' );
```

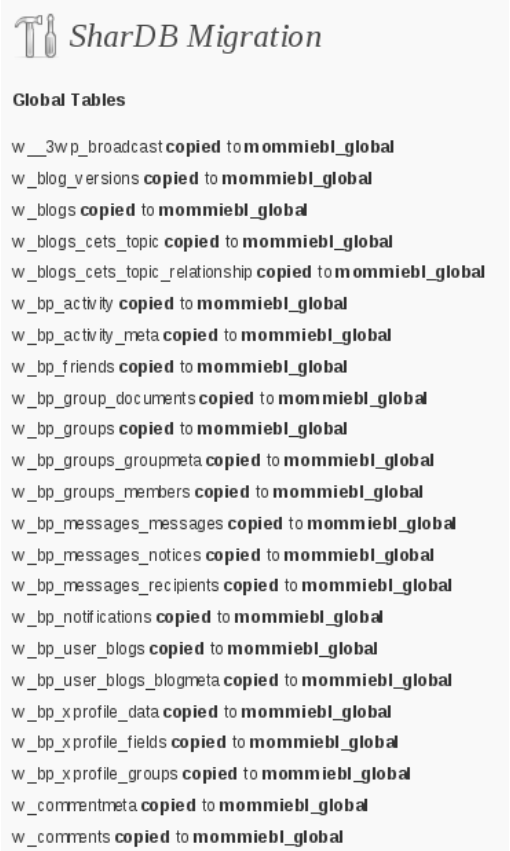
5. Save wp-config.php
6. Put the shardb-admin.php file in the mu-plugins folder, then visit admin area. If you do not have a mu-plugins folder, make one at /wp-content/mu-plugins/. Code placed in this folder will execute automatically, without activation, and cannot be disabled from the admin area.

   In the next step, we'll be migrating tables from the existing single database into our new databases.

Table migration

The menu item for SharDB is under the main site's Tools menu, and is only visible to Super Admins. Upon successful implementation, this menu item will be removed in the final step.

Click the **Migrate Global Tables** button. Check the list to make sure it copied all of the global tables. If you created your WordPress network after installing or upgrading to Wordpress 3.0, the tables belonging to the main site should also have been copied.



Once the global tables are moved successfully, you can click the button labelled **Migrate Sites**. This action will step through 5 sites at a time, much like the update network button, plus show which blog tables are going in what databases.

In our final step, move the db.php file into wp-content. The SharDB menu item will disappear. You will be able to see this file in the Network Admin -> Plugins screen, under the “Drop-Ins” tab.

Once db.php is installed a column is added to the Network Admin Sites screen that shows which database SharDB expects to find the the tables for that site. If you do not want to have that column, you may remove shardb-admin.php from the wp-content/mu-plugins folder.

Domain	LastUpdated	Registered	Users	Dataset / Partition
mommieblogs.com	2011/03/21	2010/06/27	adamaria admin alicia anthonycole beth **Only showing first 5 users.** **More**	global / mommiebl_global
sillyandrea.com	2010/11/07	2010/07/11	hsjsupport test	mb / mommiebl_c
quilti.com	2011/01/26	2010/08/08	admin hsjsupport	mb / mommiebl_e
shannon	2011/03/21	2010/08/11	shannon	mb / mommiebl_a

Appendix A

Using 256 or 4096 databases

In your db-settings.php, set `$shardb_hash_length` to 2 for 256 database or 3 for 4096 databases. For 256 databases the names will be `your_prefix_00` through `your_prefix_ff`. For 4096 databases the names will be `your_prefix_000` through `your_prefix_fff`.

If you have SSH access, you can create db-create.sql via the terminal by using the following commands:

For 256 databases:

```
$ for D in 0 1 2 3 4 5 6 7 8 9 a b c d e f; do \
for E in 0 1 2 3 4 5 6 7 8 9 a b c d e f; do \
echo CREATE DATABASE your_prefix_$D$E';' >> db-create.sql; \
echo GRANT ALL ON your_prefix_$D$E.* TO your_user';' >> db-
create.sql; \
done ; \
done ;
```

For 4096 databases:

```
$ for D in 0 1 2 3 4 5 6 7 8 9 a b c d e f; do \
for E in 0 1 2 3 4 5 6 7 8 9 a b c d e f; do \
for F in 0 1 2 3 4 5 6 7 8 9 a b c d e f; do \
echo CREATE DATABASE your_prefix_$D$E$F';' >> db-create.sql; \
echo GRANT ALL ON your_prefix_$D$E$F.* TO your_user';' >> db-
create.sql; \
done ; \
done ; \
done ;
```

Note: Neither of the above scripts includes the global database.

Using a home database

Sometimes you may wish to have the main site in the network in its own database. Reasons for this may include a main site that receives a lot of traffic, or you may be using a plugin like Sitewide Tags which creates larger than normal database tables for the main site.

Follow the same setup steps as above, but include this line in the config area of db-settings.php $enable_home_db = true;

Also create an additional database called your_prefix_home when you create the rest of the databases. When the tables are migrated, all the default WordPress tables for the main site (blog) will be placed in the “home” database, and the global tables, such as users and multisite global tables, will still go in the global database. BuddyPress tables, if you are using it, will go in the global database.

If you are creating your databases via SSH access, you can include the home database as follows:

```
$ for D in 0 1 2 3 4 5 6 7 8 9 a b c d e f home global; do \
echo CREATE DATABASE your_prefix_$D';' >> db-create.sql; \
echo GRANT ALL ON your_prefix_$D.* TO your_user';' >> db-
create.sql; \
done
```

Backup script

If you want to backup your databases using cron, there is a bash script included in the zip.

- Create a folder under your linux user account called backups
- create a text file called db-list.txt
- add the name of each of your databases to db-list.txt with each database name on its own line
- upload shardb-back & db-list.txt to the backups folder
- set the executable permission on shardb-back
- schedule your backup job in cron with the command
    - `$HOME/backups/shardb-back your_db_user your_db_password`

Each set of backups will be stored in a date based directory under the backups directory (ex. `/home/webaccount/backups/2011-03-31`). The script automatically purges off the back from 7 days prior. As long as the script is scheduled daily, all aged backups will be removed.

Troubleshooting

We recommend that you check several sub sites & all of you themes and plugins once you have completed the migration to ensure that there are no issues. You should also check your error log for database errors. Although it is rare, a plugin or theme may access the database outside of the normal use of the WordPress database class.

If you encounter any issues after migration, remove wp-content/db.php to revert to your original database.

Support

Support for this plugin will be available through the wordpress.org forums, and we will be monitoring the shardb tag.

Credits

Our thanks go to the Automatticians who developed the Hyperdb plugin, on which this work is based. Also, much thanks to Boone Gorges at http://teleogistic.net/ for being an awesome tester.