Database Backup Manager v1.5.2 - Sam (Spooks) 2012/04/12 of original by JanZ

Download from: http://addons.oscommerce.com/info/5769
Support thread: http://forums.oscommerce.com/index.php?showtopic=293949

About Database Backup Manager

This contribution aims to get the database backup manager in osC to work better with large databases especially on servers where safe_mode is set to on (safe_mode will not allow changes of the max execution time from within the script). A reasonable new version of PHP 4 (> 4.1.0 at least) is needed to work correctly.

Other changes:

Existing Backup files can be compressed plus improved support for compression.
Selective Backups can be created (on a per table bases)
Selective restores can be made (on a per table bases) given a compatible backup file (from this or Auto-Backup Database)
Backups can be emailed
Log files within the backup folder can be viewed or deleted
Backup file list can be sorted on name, date and size (ascending or descending).

Under the hood a large number of changes have been made.

- If the settings in application top for the compression programs are not set correctly:

Line 40-43 in admin/includes/application_top:

```
// Used in the "Backup Manager" to compress backups
  define('LOCAL_EXE_GZIP', '/usr/bin/gzip');
  define('LOCAL_EXE_GUNZIP', '/usr/bin/gunzip');
  define('LOCAL_EXE_ZIP', '/usr/local/bin/zip');
  define('LOCAL_EXE_UNZIP', '/usr/local/bin/unzip');
```

or due to safe_mode settings (or "exec" being on the list of disabled functions in php.ini) they are not allowed to be used, the database manager will try to use the zlib library for compression and decompression. Support for zip would have meant adding external libraries but since file compression programs on Windows can also decompress gzip'ed files (as I understand), that does not make much sense.

- You can compress an sql file or decompress a gz or zip file which is not possible in the original version (only when you make a backup you can choose to make the compressed file).

- You can choose to make a complete backup (the table sessions and whos_online will never get backed up), a backup of a single table or a number of tables.
The name of the file depends on the number of tables. If only one table the first part of the name is the name of the table, if two or more tables are chosen the first part is "partial_" and a complete backup will have the same name as before, starting with "db_".

- In the top of the sql file a line with the tables that were chosen for the backup will be written. When the backup is a partial one and is highlighted in the table of files, in the right column the table

names found in that line will be shown (when the backup file is sql, or gz compressed, but not a zip: the zlib functions contain commands to read uncompressed text from a gz compressed file without decompressing first). For an example see screenshot_db_restored.gif.

- If safe mode settings are on and the backup gets close to the maximum execution time (usually 30 seconds) the backup will be halted, an intermediate screen shown (see screenshot_reload_backup.gif) that contains all the variables for the backup. This page will be submitted automagically with a bit of JavaScript in 4 seconds and then the backup continues where it left off. The idea for this came from Xt-Dump, see:
http://dreaxteam.free.fr/forums/viewtopic.php?id=2

- Instead of trying to imitate the "create table" query that mysql does, the create table query itself is used instead.

- Instead of using:

insert into 'table_name' (column1, column2, column3) values (value1, value2, value3);

for each row in a table an extended insert is used where (maximum) 20 rows are added to the "insert into" statement:

insert into 'table_name' (column1, column2, column3) values (value_a1, value_a2, value_a3),
(value_b1, value_b2, value_b3),
(value_c1, value_c2, value_c3),
.... etcetera
(value_t1, value_t2, value_t3);

This saves on disk writes when making the backup, disk space (file size about 1/3 less) and saves a lot on mysql insert queries when restoring a backup (20 rows inserted instead of 1 per query).

- Backticks around table names and field names are used to avoid problems with mysql (e.g. when a reserved word for a table has been used by a contribution author).

- The restore procedure was totally changed. The idea and core code that was used was taken from Bigdump: http://www.ozerov.de/bigdump.php

In the original backup.php first the whole sql file was read in memory (with a large backup this would have caused a memory exceed error as still will happen with a large download), processed in separate queries and then all queries executed.
Now when a complete query is found it will be executed immediately. Then if the time gets close to the max execution time an intermediate screen (see screenshot_restore_1.gif & screenshot_restore_2.gif) will be shown that contains all the necessary information. The page will be submitted by the JavaScript in 4 seconds and the restore continues (see screenshot_restore.gif) using the file offset to start where the restore left off. NO USER INTERVENTION IS NEEDED. If you do click on some link, the restore will be likely incomplete!

- You can choose which tables to restore from a backup and also if you want to empty the tables sessions and whos_online (advisable when you do a complete restore, but not necessarily needed when you restore only certain tables).

- The uploading/restore of an sql file (with the restore button on the bottom of the page) has been changed. In the original version you could only upload an sql file. Now you can also upload a gz or

zip compressed file (provided the server can decompress them!). After the upload the temporary file was read in memory, processed in queries and then the queries executed.
Now the file is moved to the backups directory first (file name starts with "upl_" then four random characters (1-9, a-z, A-Z), an underscore "_" and then the name of database, datetime etcetera.
Note that the maximum file size you can upload is usually not more than 2 Mbyte. Larger files need to be ftp'ed to the backups directory.
The uploaded file and the decompressed file from it will be deleted from the backups directory after the restore. So if you find a file that starts with "upl_" there has been an error.

With 1.5 + Compatible with osC 2.3.x and any ealier. Any log files if found in backup folder will appear in listing and can be download for display or deleted. Backup files can now be emailed to the store owner.

Word of Caution

There is a lot of new code in this version of the database backup manager. It has been tested often on the demo osC database that is included with osC to which two large (but simple) tables were added. It has not been tested on any other databases, for example ones that use utf-8. To be certain you have a reliable backup use other software too (a mysqldump would be the best of course, phpMyAdmin as second choice). Test if the backup restores reliably on another computer (locally in another database for example, see also below for some more options).
New code usually means new bugs. Post them in the contribution thread:

http://forums.oscommerce.com/index.php?showtopic=293949

Installation

New buttons and icons need to be uploaded (the first two might already be there if you use Quick Price Updates for example, replace "english" for the language you use in the admin):

admin/includes/images/icon_down.gif
admin/includes/images/icon_up.gif
admin/includes/icons/small_tick.gif
admin/includes/languages/english/images/buttons/button_gunzip.gif
admin/includes/languages/english/images/buttons/button_gzip.gif
admin/includes/languages/english/images/buttons/button_submit.gif
admin/includes/languages/english/images/buttons/button_unzip.gif
admin/includes/languages/english/images/buttons/button_zip.gif

Then a new file should be added to functions:

admin/includes/functions/db_backup.php

The backup.php and the corresponding language file need to be replaced by the one in this package (only the English one is included):

admin/includes/languages/english/backup.php
admin/backup.php

Add new:

admin/includes/mail_backups.php

Addition for osC 2.2 and earlier only:

admin/includes/template_top.php

That's it. Of course you need to have a directory to store the backup files that can be written to by the webserver.

Upgrading:

Just replace all old files and add the new, there are no changes to image files. If upgrading from 1.5, remove /admin/includes/classes/class.phpmailer.php & class.smtp.php.

Emailing Backups:

Before you can send backups by email, you must create an email account to send them (or use an existing) then edit admin/includes/mail_backups.php and add your host, username and password details for that account, where shown in the file. Backups will be emailed to the store owners email address.

Backups will be emailed to the store owners email address as set in Admin->My Store->E-Mail Address. Backups will be emailed through the pear mail function if available, otherwise the less secure php mail function. If you receive a warning when selecting 'email backup' that pear has issues ask your host to add pear mail with smtp & mime classes. pear mail is safer & more robust the php mail, so enable if you can.

Testing

To be able to test the functions to do a backup or a restore in several "runs" a large database table (gzip'ed) that has nothing to do with osC has been added to the download of version 1.3 (June 7, 2008): digitemp_osc22rc1-20080607141540.sql.gz. Download that version for the file.

The backup was made with this contribution and took 130 seconds to restore (about 300,000 rows are in the table). The ungzip'ed file is 19.4 MB.
If your server is really fast and/or does not have save settings set to 'on' you can change a few lines in the file admin/backup.php to emulate that situation. Comment line 24 and uncomment line 25 like so:

```
// $safe_mode_setting = (@ini_get('safe_mode') == 'On' || (@ini_get('safe_mode') === 1)) ? true : false;
 $safe_mode_setting = true; // for testing
```

If you are on a fast server or have a long time before a PHP script times out you can also enlarge the safety margin (but of course never make that larger than the ini value for max_execution_time (see admin->Tools->Server Info->Configuration [PHP Core]).

The safety margin is set in line 36 in admin/backup.php:

```
$safety_margin = 5; // margin to be kept till end of max_execution_time
```