source: database-toolset / trunk / admin / class-database-toolset-admin.php

Last change on this file was 2964538, checked in by neoslab, 20 months ago
Updated to new version
File size: 21.3 KB

```php
<?php
/**
 * The admin-specific functionality of the plugin
 *
 * @link https://neoslab.com
 * @since 1.0.0
 * @package Database_Toolset
 * @subpackage Database_Toolset/admin
*/

/**
 * Class `Database_Toolset_Admin`
 * @package Database_Toolset
 * @subpackage Database_Toolset/admin
 * @author Neoslab <contact@neoslab.com>
*/
class Database_Toolset_Admin
{
        /**
         * The ID of this plugin
         * @since 1.0.0
         * @access private
         * @var string $pluginName the ID of this plugin
        */
        private $pluginName;

        /**
         * The version of this plugin
         * @since 1.0.0
         * @access private
         * @var string $version the current version of this plugin
        */
        private $version;

        /**
         * Initialize the class and set its properties
         * @since 1.0.0
         * @param string $pluginName the name of this plugin
         * @param string $version the version of this plugin
        */
        public function __construct($pluginName, $version)
        {
                $this->pluginName = $pluginName;
                $this->version = $version;
        }

        /**
         * Register the stylesheets for the admin area
         * @since 1.0.0
```

```
50          */
51          public function enqueue_styles()
52          {
53                  wp_register_style($this->pluginName.'-fontawesome', plugin_dir_url(__FILE__).'assets/styles/fontawesome.min.css', array(), $this->version, 'all');
54                  wp_register_style($this->pluginName.'-datatables', plugin_dir_url(__FILE__).'assets/styles/datatables.min.css', array(), $this->version, 'all');
55                  wp_register_style($this->pluginName.'-dashboard', plugin_dir_url(__FILE__).'assets/styles/database-toolset-admin.min.css', array(), $this->version, 'all');
56                  wp_enqueue_style($this->pluginName.'-fontawesome');
57                  wp_enqueue_style($this->pluginName.'-datatables');
58                  wp_enqueue_style($this->pluginName.'-dashboard');
59          }
60
61          /**
62           * Register the JavaScript for the admin area
63           * @since 1.0.0
64          */
65          public function enqueue_scripts()
66          {
67                  wp_register_script($this->pluginName.'-datatables', plugin_dir_url(__FILE__).'assets/javascripts/datatables.min.js', array('jquery'), $this->version, false);
68                  wp_register_script($this->pluginName.'-script', plugin_dir_url(__FILE__).'assets/javascripts/database-toolset-admin.min.js', array('jquery'), $this->version, false);
69                  wp_enqueue_script($this->pluginName.'-datatables');
70                  wp_enqueue_script($this->pluginName.'-script');
71          }
72
73          /**
74           * Return the plugin header
75          */
76          public function return_plugin_header()
77          {
78                  $html = '<div class="wpbnd-header-plugin"><span class="header-icon"><i class="fas fa-sliders-h"></i></span> <span class="header-text">'.__('Database Toolset', 'database-toolset').'</span></div>';
79                  return $html;
80          }
81
82          /**
83           * Return the tabs menu
84          */
85          public function return_tabs_menu($tab)
86          {
87                  $link = admin_url('admin.php');
88                  $list = array
89                  (
90                          array('tab1', 'database-toolset-cleanup', 'fa-broom', __('Clean-Up', 'database-toolset')),
91                          array('tab2', 'database-toolset-optimize', 'fa-database', __('Optimizer', 'database-toolset')),
92                          array('tab3', 'database-toolset-schedule', 'fa-clock', __('Tasks', 'database-toolset')),
93                          array('tab4', 'database-toolset-table', 'fa-hdd', __('Backups', 'database-toolset')),
94                          array('tab5', 'database-toolset-speedup', 'fa-chart-line', __('Performance', 'database-toolset'))
95                  );
96
97                  $menu = null;
98                  foreach($list as $item => $value)
99                  {
100 $html = array('div' => array('class' => array()), 'a' => array('href' => array()), 'i' => array('class' => array()), 'p' => array(), 'span' => array());
```

```
                        $menu ='<div class="tab-label '.$value[0].' '.
((($tab === $value[0]) ? 'active' : '').'"><a href="'.$link.'?page='.$value[1].'"><p><i class="fas
'.$value[2].'"></i><span>'.$value[3].'</span></p></a></div>';
                        echo wp_kses($menu, $html);
                }
        }

        /**
         * Remove specific backup
        */
        public function remove_specific_backup()
        {
                if((isset($_GET['action'])) && ($_GET['action'] === 'remove'))
                {
                        if(isset($_GET['backup']))
                        {
                                $loadpath = wp_upload_dir();
                                $filepath = $loadpath["basedir"].'/export/database/'.$_GET['backup'];

                                if(file_exists($filepath))
                                {
                                        unlink($filepath);
                                        header('location:'.admin_url('admin.php?page=database-toolset-
table').'&action=deleted');
                                        die();
                                }
                        }
                }
        }

        /**
         * Create Cron Schedules
        */
        public function create_cron_schedules($schedules)
        {
            if(!isset($schedules["5min"]))
            {
                $schedules["5min"] = array
                (
                    'interval' => 5*60,
                    'display' => __('Once every 5 minutes', 'database-toolset')
                );
            }

            if(!isset($schedules["15min"]))
            {
                $schedules["15min"] = array
                (
                    'interval' => 15*60,
                    'display' => __('Once every 15 minutes', 'database-toolset')
                );
            }

            if(!isset($schedules["30min"]))
            {
                $schedules["30min"] = array
                (
                    'interval' => 30*60,
                    'display' => __('Once every 30 minutes', 'database-toolset')
                );
            }

            if(!isset($schedules["60min"]))
            {
```

```
                $schedules["60min"] = array
                (
                    'interval' => 60*60,
                    'display' => __('Once every 60 minutes', 'database-toolset')
                );
            }

            if(!isset($schedules["6h"]))
            {
                $schedules["6h"] = array
                (
                    'interval' => 60*60*6,
                    'display' => __('Once every 6 hours', 'database-toolset')
                );
            }

            if(!isset($schedules["12h"]))
            {
                $schedules["12h"] = array
                (
                    'interval' => 60*60*12,
                    'display' => __('Once every 12 hours', 'database-toolset')
                );
            }

            if(!isset($schedules["24h"]))
            {
                $schedules["24h"] = array
                (
                    'interval' => 60*60*24,
                    'display' => __('Once every 24 hours', 'database-toolset')
                );
            }

            if(!isset($schedules["7d"]))
            {
                $schedules["7d"] = array
                (
                    'interval' => 60*60*24*7,
                    'display' => __('Once weekly', 'database-toolset')
                );
            }

            return $schedules;
        }

        /**
         * Create Cron Event
        */
        public function create_cron_schedule()
        {
                $opts = get_option('_database_toolset');
                if(!wp_next_scheduled('database_toolset_backup'))
                {

if(((isset($opts['status'])) && ($opts['status'] === 'on')) && ((isset($opts['crontab'])) && (!empty($opts['crontab'])) && ($opts['crontab'] !== 'never')))
                        {
                                wp_schedule_event(time(), $opts['crontab'], 'database_toolset_backup');
                        }
                }
                else
                {
```

```
Line
223                         $cronjob = wp_get_schedule('database_toolset_backup');
224                         $allowed = array('never', '5min', '15min', '30min', '60min', '6h', '12h', '24h', '7d');
225                         if((isset($cronjob)) && (in_array($cronjob, $allowed)))
226                         {
227                                 if((!empty($opts['status'])) && ($opts['status'] === 'off'))
228                                 {
229                                         wp_clear_scheduled_hook('database_toolset_backup');
230                                 }
231                                 elseif((!empty($opts['crontab'])) && ($opts['crontab'] === 'never'))
232                                 {
233                                         wp_clear_scheduled_hook('database_toolset_backup');
234                                 }
235                                 elseif($cronjob !== $opts['crontab'])
236                                 {
237                                         wp_clear_scheduled_hook('database_toolset_backup');
238                                         wp_schedule_event(time(), $opts['crontab'], 'database_toolset_backup');
239                                 }
240                         }
241                 }
242         }
243
244         /**
245          * Create Backup Folders
246         */
247         public function create_backup_folders()
248         {
249                 $loadpath = wp_upload_dir();
250                 $filepath = $loadpath["basedir"].'/export/database/';
251                 if(!file_exists($filepath))
252                 {
253                         mkdir($filepath, 0755, true);
254                 }
255         }
256
257         /**
258          * Create index files to protect each created folder
259         */
260         public function create_backup_indexes()
261         {
262                 $loadpath = wp_upload_dir();
263                 $filepath = $loadpath["basedir"].'/export/index.php';
264                 if(!file_exists($filepath))
265                 {
266                         $fh = fopen($filepath, 'w') or die("can't open file");
267                         fwrite($fh, '<?php // Silence is golden ?>');
268                         fclose($fh);
269                         chmod($filepath, 0600);
270                 }
271
272                 $filepath = $loadpath["basedir"].'/export/database/index.php';
273                 if(!file_exists($filepath))
274                 {
275                         $fh = fopen($filepath, 'w') or die("can't open file");
276                         fwrite($fh, '<?php // Silence is golden ?>');
277                         fclose($fh);
278                         chmod($filepath, 0600);
279                 }
280         }
281
282         /**
283          * Convert File size unit
284         */
285     public function return_formatted_size($bytes)
```

```
    {
        if($bytes >= 1073741824)
        {
            $bytes = number_format($bytes / 1073741824, 2).' GB';
        }
        elseif($bytes >= 1048576)
        {
            $bytes = number_format($bytes / 1048576, 2).' MB';
        }
        elseif($bytes >= 1024)
        {
            $bytes = number_format($bytes / 1024, 2).' KB';
        }
        elseif($bytes > 1)
        {
            $bytes = $bytes.' bytes';
        }
        elseif($bytes === 1)
        {
            $bytes = $bytes.' byte';
        }
        else
        {
            $bytes = '0 bytes';
        }

        return $bytes;
        }

        /**
         * Send notification to website owner
        */
        public function send_user_notice($site, $link)
    {
        $opts = get_option('_database_toolset');

                $get_upload = wp_upload_dir();
                if((isset($opts['gzip'])) && ($opts['gzip'] === 'on'))
                {
                        $get_backup = $get_upload['baseurl'].'/export/database/'.$link.'.gz';
                }
                else
                {
                        $get_backup = $get_upload['baseurl'].'/export/database/'.$link;
                }

                $backupDate = date('d/m/Y');
                $backupTime = date('H:i:s');

        $message = __('Hi,', 'database-toolset')."\r\n\r\n";
        $message.= sprintf(__('This email was sent from your website %s by the Database Toolset plugin to inform you that a new backup of your database was successfully created on %s at %s.', 'database-toolset'), $site, $backupDate, $backupTime)."\r\n\r\n";
        $message.= sprintf(__('You can download a copy of this backup at anytime by clicking the following link: %s', 'database-toolset'), $get_backup)."\r\n\r\n";
        $message.= __('Email generated by', 'database-toolset')."\r\n";
        $message.= __('Database Toolset', 'database-toolset')."\r\n";

        $subject = '['.get_site_url().'] - '.__('New Database Backup Available', 'database-toolset');
        wp_mail($opts['email'], $subject, $message);
    }

        /**
         * Create database backup
```

```
347         */
348         public function create_backup_database()
349         {
350                 $opts = get_option('_database_toolset');
351                 $gzip = false;
352
353                 $loadpath = wp_upload_dir();
354                 $filepath = $loadpath["basedir"].'/export/database/';
355
356                 if((isset($opts['gzip'])) && ($opts['gzip'] === 'on'))
357                 {
358                         $gzip = true;
359                 }
360
361                 $folders = $this->create_backup_folders();
362                 $indexes = $this->create_backup_indexes();
363
364                 /**
365                  * Define the constants
366                 */
367                 define("DBT_BACKUP_PATH", $filepath);           // The full path of the directory where to save the backup
368                 define("DBT_BACKUP_TABLES", '*');                       // By default all tables will be save "*", but it can be also a list of tables such as "define("DBT_BACKUP_TABLES", 'table1, table2, table3');"
369                 define("DBT_BACKUP_GZIP", $gzip);                       // Set to false if you want plain SQL backup files (not gzipped)
370                 define("DBT_BACKUP_KEY_CHECKS", true);          // Set to true if you are having foreign key constraint fails
371                 define("DBT_BACKUP_BATCH_SIZE", 1000);          // Batch size when selecting rows from database in order to not exhaust system memory
372
373                 /**
374                  * The class responsible for orchestrating the backup of the database
375                 */
376                 require_once plugin_dir_path(dirname(__FILE__)).'admin/class-database-toolset-backup.php';
377                 $sqlBackup = new Database_Toolset_Backup(DB_HOST, DB_USER, DB_PASSWORD, DB_NAME, DB_CHARSET);
378                 $sqlResult = $sqlBackup->backupTables(DBT_BACKUP_TABLES, DBT_BACKUP_PATH);
379
380 if(((isset($opts['notify'])) && ($opts['notify'] === 'on')) && ((isset($opts['email'])) && (!empty($opts['email'])))) && ($sqlResult !== false))
381                 {
382                         $this->send_user_notice(get_bloginfo('name'), $sqlResult);
383                 }
384         }
385
386         /**
387          * Update `Options` on form submit
388         */
389         public function return_update_options()
390         {
391                 if((isset($_POST['dbt-update-option'])) && ($_POST['dbt-update-option'] === 'true')
392                 && check_admin_referer('dbt-referer-form', 'query-option'))
393                 {
394 $opts = array('status' => 'off', 'gzip' => 'off', 'notify' => 'off', 'email' => 'none', 'crontab' => 'never');
395                         if((isset($_POST['_database_toolset']['email']))
396                         && (isset($_POST['_database_toolset']['crontab'])))
397                         {
398                                 if(isset($_POST['_database_toolset']['status']))
399                                 {
400                                         $opts['status'] = 'on';
401                                 }
402
403                                 if(isset($_POST['_database_toolset']['gzip']))
```

```
404                                 {
405                                         $opts['gzip'] = 'on';
406                                 }
407
408                                 if(isset($_POST['_database_toolset']['notify']))
409                                 {
410                                         $opts['notify'] = 'on';
411                                 }
412
413                                 if((isset($_POST['_database_toolset']
['email'])) && ((isset($_POST['_database_toolset']['notify'])) && ($_POST['_database_toolset']
['notify'] === 'on')))
414                                 {
415                                         $opts['email'] = sanitize_email($_POST['_database_toolset']['email']);
416                                         if(is_email($opts['email']) === false)
417                                         {
418                                                 header('location:'.admin_url('admin.php?page=database-toolset-
schedule').'&status=error&type=email');
419                                                 die();
420                                         }
421                                 }
422
423
$allowed = array('never', '5min', '15min', '30min', '60min', '6h', '12h', '24h', '7d');
424                                 if((isset($_POST['_database_toolset']
['crontab'])) && (in_array($_POST['_database_toolset']['crontab'], $allowed)))
425                                 {
426                                         $opts['crontab'] = sanitize_text_field($_POST['_database_toolset']
['crontab']);
427                                 }
428
429                                 $data = update_option('_database_toolset', $opts);
430                                 header('location:'.admin_url('admin.php?page=database-toolset-
schedule').'&status=updated');
431                                 die();
432                         }
433                         else
434                         {
435                                 header('location:'.admin_url('admin.php?page=database-toolset-
schedule').'&status=error&type=unknown');
436                                 die();
437                         }
438                 }
439         }
440
441         /**
442          * Return Database Speed-Up
443         */
444         public function return_query_speedup($options)
445         {
446                 global $wpdb;
447
448                 if($options === true)
449                 {
450                         /**
451                          * Reset Options Increment
452                         */
453                         $query = "ALTER TABLE `".$wpdb->options."` DROP `option_id`;";
454                         $fetch = $wpdb->query($query);
455
456                         $query = "ALTER TABLE `".$wpdb->options."` ADD `option_id` bigint(20) NOT NULL
AUTO_INCREMENT PRIMARY KEY FIRST;";
457                         $fetch = $wpdb->query($query);
458                 }
459         }
```

```
Line
460
461         /**
462          * Delete entries from database
463          * @param $type the type of post_type, post_status or comment_approved to search for
464         */
465         public function return_query_cleanup($type)
466         {
467                 global $wpdb;
468
469                 switch($type)
470                 {
471                         case "posts-revision":
472                         $query = "DELETE FROM `".$wpdb->posts."` WHERE `post_type` = 'revision'";
473                         $fetch = $wpdb->query($query);
474                         break;
475
476                         case "posts-draft":
477                         $query = "DELETE FROM `".$wpdb->posts."` WHERE `post_status` = 'draft'";
478                         $fetch = $wpdb->query($query);
479                         break;
480
481                         case "posts-autodraft":
482                         $query = "DELETE FROM `".$wpdb->posts."` WHERE `post_status` = 'auto-draft'";
483                         $fetch = $wpdb->query($query);
484                         break;
485
486                         case "posts-changeset":
487                         $query = "SELECT `ID` FROM `".$wpdb->posts."` WHERE `post_type` = 'customize_changeset'
    AND `post_status` = 'trash'";
488                         $fetch = $wpdb->get_results($query, OBJECT);
489
490                         foreach($fetch as $object)
491                         {
492                                 $query = "DELETE FROM `".$wpdb->posts."` WHERE `ID` = '".$object->ID."'";
493                                 $fetch = $wpdb->query($query);
494
495                                 $query = "DELETE FROM `".$wpdb->postmeta."` WHERE `post_id` = '".$object-
    >ID."'";
496                                 $fetch = $wpdb->query($query);
497                         }
498                         break;
499
500                         case "comments-moderated":
501                         $query = "DELETE FROM `".$wpdb->comments."` WHERE `comment_approved` = '0'";
502                         $fetch = $wpdb->query($query);
503                         break;
504
505                         case "comments-spam":
506                         $query = "DELETE FROM `".$wpdb->comments."` WHERE `comment_approved` = 'spam'";
507                         $fetch = $wpdb->query($query);
508                         break;
509
510                         case "comments-trash":
511                         $query = "DELETE FROM `".$wpdb->comments."` WHERE `comment_approved` = 'trash'";
512                         $fetch = $wpdb->query($query);
513                         break;
514
515                         case "orphan-postmeta":
516                         $query = "DELETE `pm` FROM `".$wpdb->postmeta."` pm LEFT JOIN `".$wpdb->posts."` wp ON
    wp.ID = pm.post_id WHERE wp.ID IS NULL";
517                         $fetch = $wpdb->query($query);
518                         break;
519
520                         case "orphan-commentmeta":
```

```
                        $query = "DELETE FROM `".$wpdb->commentmeta."` WHERE `comment_id` NOT IN (SELECT comment_id FROM ".$wpdb->comments.")";
                        $fetch = $wpdb->query($query);
                        break;

                        case "orphan-relationships":
                        $query = "DELETE FROM `".$wpdb->term_relationships."` WHERE `term_taxonomy_id` = 1 AND `object_id` NOT IN (SELECT id FROM ".$wpdb->posts.")";
                        $fetch = $wpdb->query($query);
                        break;

                        case "orphan-transient-feed":
                        $query = "DELETE FROM `".$wpdb->options."` WHERE `option_name` LIKE '_site_transient_browser_%' OR option_name LIKE '_site_transient_timeout_browser_%' OR option_name LIKE '_transient_feed_%' OR option_name LIKE '_transient_timeout_feed_%'";
                        $fetch = $wpdb->query($query);
                        break;
                }
        }

        /**
         * Count entries from database
         * @param $type the type of post_type, post_status or comment_approved to search for
        */
        public function return_query_count($type)
        {
                global $wpdb;

                switch($type)
                {
                        case "posts-revision":
                        $query = "SELECT COUNT(*) FROM `".$wpdb->posts."` WHERE `post_type` = 'revision'";
                        $fetch = $wpdb->get_var($query);
                        break;

                        case "posts-draft":
                        $query = "SELECT COUNT(*) FROM `".$wpdb->posts."` WHERE `post_status` = 'draft'";
                        $fetch = $wpdb->get_var($query);
                        break;

                        case "posts-autodraft":
                        $query = "SELECT COUNT(*) FROM `".$wpdb->posts."` WHERE `post_status` = 'auto-draft'";
                        $fetch = $wpdb->get_var($query);
                        break;

                        case "posts-changeset":
                        $query = "SELECT COUNT(*) FROM `".$wpdb->posts."` WHERE `post_type` = 'customize_changeset' AND `post_status` = 'trash'";
                        $fetch = $wpdb->get_var($query);
                        break;

                        case "comments-moderated":
                        $query = "SELECT COUNT(*) FROM `".$wpdb->comments."` WHERE `comment_approved` = '0'";
                        $fetch = $wpdb->get_var($query);
                        break;

                        case "comments-spam":
                        $query = "SELECT COUNT(*) FROM `".$wpdb->comments."` WHERE `comment_approved` = 'spam'";
                        $fetch = $wpdb->get_var($query);
                        break;

                        case "comments-trash":
                        $query = "SELECT COUNT(*) FROM `".$wpdb->comments."` WHERE `comment_approved` = 'trash'";
```

```
                        $fetch = $wpdb->get_var($query);
                        break;

                        case "orphan-postmeta":
                        $query = "SELECT COUNT(*) FROM `".$wpdb->postmeta."` pm LEFT JOIN `".$wpdb->posts."` wp ON wp.ID = pm.post_id WHERE wp.ID IS NULL";
                        $fetch = $wpdb->get_var($query);
                        break;

                        case "orphan-commentmeta":
                        $query = "SELECT COUNT(*) FROM `".$wpdb->commentmeta."` WHERE `comment_id` NOT IN (SELECT comment_id FROM ".$wpdb->comments.")";
                        $fetch = $wpdb->get_var($query);
                        break;

                        case "orphan-relationships":
                        $query = "SELECT COUNT(*) FROM `".$wpdb->term_relationships."` WHERE `term_taxonomy_id` = 1 AND `object_id` NOT IN (SELECT id FROM ".$wpdb->posts.")";
                        $fetch = $wpdb->get_var($query);
                        break;

                        case "orphan-transient-feed":
                        $query = "SELECT COUNT(*) FROM `".$wpdb->options."` WHERE `option_name` LIKE '_site_transient_browser_%' OR option_name LIKE '_site_transient_timeout_browser_%' OR option_name LIKE '_transient_feed_%' OR option_name LIKE '_transient_timeout_feed_%'";
                        $fetch = $wpdb->get_var($query);
                        break;
                }

                $count = $fetch;
                return $count;
        }

        /**
         * Return Database Optimization
        */
        public function return_query_optimize()
        {
                global $wpdb;

                $query = 'SHOW TABLE STATUS FROM `'.DB_NAME.'`';
                $fetch = $wpdb->get_results($query);

                foreach($fetch as $row)
                {
                        $mysql = 'OPTIMIZE TABLE '.$row->Name;
                        $wpdb->query($mysql);
                }
        }

        /**
         * Return the `Clean-Up` page
        */
        public function return_cleanup_page()
        {
                global $wpdb, $table_prefix;
                require_once plugin_dir_path(__FILE__).'partials/database-toolset-admin-cleanup.php';
        }

        /**
         * Return the `Optimizer` page
        */
        public function return_optimizer_page()
        {
```

```
638                 global $wpdb, $table_prefix;
639                 require_once plugin_dir_path(__FILE__).'partials/database-toolset-admin-optimize.php';
640         }
641
642         /**
643          * Return the `Schedule` page
644         */
645         public function return_schedule_page()
646         {
647                 $opts = get_option('_database_toolset');
648                 require_once plugin_dir_path(__FILE__).'partials/database-toolset-admin-schedule.php';
649         }
650
651         /**
652          * Return the `Table` page
653         */
654         public function return_table_page()
655         {
656                 require_once plugin_dir_path(__FILE__).'partials/database-toolset-admin-table.php';
657         }
658
659         /**
660          * Return the `Speedup` page
661         */
662         public function return_speedup_page()
663         {
664                 require_once plugin_dir_path(__FILE__).'partials/database-toolset-admin-speedup.php';
665         }
666
667         /**
668          * Return Backend Menu
669         */
670         public function return_admin_menu()
671         {
672                 add_menu_page('Database Toolset', 'Database Toolset', 'administrator', 'database-toolset-
    admin', array($this, 'return_cleanup_page'), 'dashicons-feedback');
673                 add_submenu_page('database-toolset-admin', 'Clean-Up', 'Clean-Up', 'administrator', 'database-
    toolset-cleanup', array($this, 'return_cleanup_page'));
674                 add_submenu_page('database-toolset-
    admin', 'Optimizer', 'Optimizer', 'administrator', 'database-toolset-
    optimize', array($this, 'return_optimizer_page'));
675                 add_submenu_page('database-toolset-admin', 'Tasks', 'Tasks', 'administrator', 'database-
    toolset-schedule', array($this, 'return_schedule_page'));
676                 add_submenu_page('database-toolset-admin', 'Backups', 'Backups', 'administrator', 'database-
    toolset-table', array($this, 'return_table_page'));
677                 add_submenu_page('database-toolset-
    admin', 'Performance', 'Performance', 'administrator', 'database-toolset-
    speedup', array($this, 'return_speedup_page'));
678                 remove_submenu_page('database-toolset-admin', 'database-toolset-admin');
679         }
680 }
681
682 ?>
```

Donate

Swag

Documentation

Developers

Get Involved

Learn

Showcase

Plugins

Themes

Patterns

WordCamp

WordPress.TV

BuddyPress

bbPress

WordPress.com

Matt

Privacy

Public Code