**CSE-3002**

**INTERNET AND WEB PROGRAMMING**

**CHALLENGING TASK 2**

**NAME:- SWATHI T**

**REGISTRATION NUMBER:- 15BCE0221**

**CODE**

**db\_operations.php**

<?php

\* @package PhpMyAdmin

\*/

use PMA\libraries\Response;

use PMA\libraries\plugins\export\ExportSql;

require\_once 'libraries/common.inc.php';

require\_once 'libraries/display\_create\_table.lib.php';

require\_once 'libraries/check\_user\_privileges.lib.php';

require\_once 'libraries/operations.lib.php';

$response = Response::getInstance();

$header = $response->getHeader();

$scripts = $header->getScripts();

$scripts->addFile('db\_operations.js');

$sql\_query = '';

if (strlen($GLOBALS['db']) > 0

&& (! empty($\_REQUEST['db\_rename']) || ! empty($\_REQUEST['db\_copy']))

) {

if (! empty($\_REQUEST['db\_rename'])) {

$move = true;

} else {

$move = false;

}

if (! isset($\_REQUEST['newname']) || strlen($\_REQUEST['newname']) === 0) {

$message = PMA\libraries\Message::error(\_\_('The database name is empty!'));

} else {

$\_error = false;

if ($move || ! empty($\_REQUEST['create\_database\_before\_copying'])) {

PMA\_createDbBeforeCopy();

}

PMA\_runProcedureAndFunctionDefinitions($GLOBALS['db']);

$GLOBALS['dbi']->selectDb($GLOBALS['db']);

$tables\_full = $GLOBALS['dbi']->getTablesFull($GLOBALS['db']);

include\_once "libraries/plugin\_interface.lib.php";

$export\_sql\_plugin = PMA\_getPlugin(

"export",

"sql",

'libraries/plugins/export/',

array(

'single\_table' => isset($single\_table),

'export\_type' => 'database'

)

);

$views = PMA\_getViewsAndCreateSqlViewStandIn(

$tables\_full, $export\_sql\_plugin, $GLOBALS['db']

);

$sqlConstratints = PMA\_copyTables(

$tables\_full, $move, $GLOBALS['db']

);

if (! $\_error) {

PMA\_handleTheViews($views, $move, $GLOBALS['db']);

}

unset($views);

if (! $\_error && count($sqlConstratints) > 0) {

PMA\_createAllAccumulatedConstraints($sqlConstratints);

}

unset($sqlConstratints);

if (PMA\_MYSQL\_INT\_VERSION >= 50100) {

PMA\_runEventDefinitionsForDb($GLOBALS['db']);

}

$GLOBALS['dbi']->selectDb($GLOBALS['db']);

PMA\_duplicateBookmarks($\_error, $GLOBALS['db']);

if (! $\_error && $move) {

if (isset($\_REQUEST['adjust\_privileges'])

&& ! empty($\_REQUEST['adjust\_privileges'])

) {

PMA\_AdjustPrivileges\_moveDB($GLOBALS['db'], $\_REQUEST['newname']);

}

include\_once 'libraries/relation\_cleanup.lib.php';

PMA\_relationsCleanupDatabase($GLOBALS['db']);

$local\_query = 'DROP DATABASE '

. PMA\libraries\Util::backquote($GLOBALS['db']) . ';';

$sql\_query .= "\n" . $local\_query;

$GLOBALS['dbi']->query($local\_query);

$message = PMA\libraries\Message::success(

\_\_('Database %1$s has been renamed to %2$s.')

);

$message->addParam($GLOBALS['db']);

$message->addParam($\_REQUEST['newname']);

} elseif (! $\_error) {

if (isset($\_REQUEST['adjust\_privileges'])

&& ! empty($\_REQUEST['adjust\_privileges'])

) {

PMA\_AdjustPrivileges\_copyDB($GLOBALS['db'], $\_REQUEST['newname']);

}

$message = PMA\libraries\Message::success(

\_\_('Database %1$s has been copied to %2$s.')

);

$message->addParam($GLOBALS['db']);

$message->addParam($\_REQUEST['newname']);

} else {

$message = PMA\libraries\Message::error();

}

$reload = true;

if (! $\_error && $move) {

$GLOBALS['db'] = $\_REQUEST['newname'];

} elseif (! $\_error) {

if (isset($\_REQUEST['switch\_to\_new'])

&& $\_REQUEST['switch\_to\_new'] == 'true'

) {

$GLOBALS['PMA\_Config']->setCookie('pma\_switch\_to\_new', 'true');

$GLOBALS['db'] = $\_REQUEST['newname'];

} else {

$GLOBALS['PMA\_Config']->setCookie('pma\_switch\_to\_new', '');

}

}

}

if ($response->isAjax()) {

$response->setRequestStatus($message->isSuccess());

$response->addJSON('message', $message);

$response->addJSON('newname', $\_REQUEST['newname']);

$response->addJSON(

'sql\_query',

PMA\libraries\Util::getMessage(null, $sql\_query)

);

$response->addJSON('db', $GLOBALS['db']);

exit;

}

}

$cfgRelation = PMA\_getRelationsParam();

if (isset($\_REQUEST['comment'])) {

PMA\_setDbComment($GLOBALS['db'], $\_REQUEST['comment']);

}

require 'libraries/db\_common.inc.php';

$url\_query .= '&amp;goto=db\_operations.php';

$sub\_part = '\_structure';

list(

$tables,

$num\_tables,

$total\_num\_tables,

$sub\_part,

$is\_show\_stats,

$db\_is\_system\_schema,

$tooltip\_truename,

$tooltip\_aliasname,

$pos

) = PMA\libraries\Util::getDbInfo($db, isset($sub\_part) ? $sub\_part : '');

echo "\n";

if (isset($message)) {

echo PMA\libraries\Util::getMessage($message, $sql\_query);

unset($message);

}

$\_REQUEST['db\_collation'] = $GLOBALS['dbi']->getDbCollation($GLOBALS['db']);

$is\_information\_schema = $GLOBALS['dbi']->isSystemSchema($GLOBALS['db']);

$response->addHTML('<div id="boxContainer" data-box-width="300">');

if (!$is\_information\_schema) {

if ($cfgRelation['commwork']) {

$response->addHTML(PMA\_getHtmlForDatabaseComment($GLOBALS['db']));

}

$response->addHTML('<div class="operations\_half\_width">');

$response->addHTML(PMA\_getHtmlForCreateTable($db));

$response->addHTML('</div>');

if ($GLOBALS['db'] != 'mysql') {

$response->addHTML(PMA\_getHtmlForRenameDatabase($GLOBALS['db']));

}

if (($is\_superuser || $GLOBALS['cfg']['AllowUserDropDatabase'])

&& ! $db\_is\_system\_schema

&& $GLOBALS['db'] != 'mysql'

) {

$response->addHTML(PMA\_getHtmlForDropDatabaseLink($GLOBALS['db']));

}

$response->addHTML(PMA\_getHtmlForCopyDatabase($GLOBALS['db']));

$response->addHTML(PMA\_getHtmlForChangeDatabaseCharset($GLOBALS['db'], $table));

if (! $cfgRelation['allworks']

&& $cfg['PmaNoRelation\_DisableWarning'] == false

) {

$message = PMA\libraries\Message::notice(

\_\_(

'The phpMyAdmin configuration storage has been deactivated. ' .

'%sFind out why%s.'

)

);

$message->addParamHtml('<a href="./chk\_rel.php' . $url\_query . '">');

$message->addParamHtml('</a>');

/\* Show error if user has configured something, notice elsewhere \*/

if (!empty($cfg['Servers'][$server]['pmadb'])) {

$message->isError(true);

}

}

}

$response->addHTML('</div>');

if ($cfgRelation['pdfwork'] && $num\_tables > 0) {

$test\_query = '

SELECT \*

FROM ' . PMA\libraries\Util::backquote($GLOBALS['cfgRelation']['db'])

. '.' . PMA\libraries\Util::backquote($cfgRelation['pdf\_pages']) . '

WHERE db\_name = \'' . $GLOBALS['dbi']->escapeString($GLOBALS['db'])

. '\'';

$test\_rs = PMA\_queryAsControlUser(

$test\_query,

false,

PMA\libraries\DatabaseInterface::QUERY\_STORE

);

}

**db\_operations.js**

AJAX.registerTeardown('db\_operations.js', function () {

$(document).off('submit', "#rename\_db\_form.ajax");

$(document).off('submit', "#copy\_db\_form.ajax");

$(document).off('submit', "#change\_db\_charset\_form.ajax");

$(document).off('click', "#drop\_db\_anchor.ajax");

});

AJAX.registerOnload('db\_operations.js', function () {

$(document).on('submit', "#rename\_db\_form.ajax", function (event) {

event.preventDefault();

var old\_db\_name = PMA\_commonParams.get('db');

var new\_db\_name = $('#new\_db\_name').val();

if (new\_db\_name == old\_db\_name) {

PMA\_ajaxShowMessage(PMA\_messages.strDatabaseRenameToSameName, false, "error");

return false;

}

var $form = $(this);

var question = escapeHtml('CREATE DATABASE ' + new\_db\_name + ' / DROP DATABASE ' + old\_db\_name);

PMA\_prepareForAjaxRequest($form);

$form.PMA\_confirm(question, $form.attr('action'), function (url) {

PMA\_ajaxShowMessage(PMA\_messages.strRenamingDatabases, false);

$.post(url, $("#rename\_db\_form").serialize() + '&is\_js\_confirmed=1', function (data) {

if (typeof data !== 'undefined' && data.success === true) {

PMA\_ajaxShowMessage(data.message);

PMA\_commonParams.set('db', data.newname);

PMA\_reloadNavigation(function () {

$('#pma\_navigation\_tree')

.find("a:not('.expander')")

.each(function (index) {

var $thisAnchor = $(this);

if ($thisAnchor.text() == data.newname) {

$thisAnchor.trigger('click');

}

});

});

} else {

PMA\_ajaxShowMessage(data.error, false);

}

}); // end $.post()

});

});

$(document).on('submit', "#copy\_db\_form.ajax", function (event) {

event.preventDefault();

PMA\_ajaxShowMessage(PMA\_messages.strCopyingDatabase, false);

var $form = $(this);

PMA\_prepareForAjaxRequest($form);

$.post($form.attr('action'), $form.serialize(), function (data) {

$('div.success, div.error').fadeOut();

if (typeof data !== 'undefined' && data.success === true) {

if ($("#checkbox\_switch").is(":checked")) {

PMA\_commonParams.set('db', data.newname);

PMA\_commonActions.refreshMain(false, function () {

PMA\_ajaxShowMessage(data.message);

});

} else {

PMA\_commonParams.set('db', data.db);

PMA\_ajaxShowMessage(data.message);

}

PMA\_reloadNavigation();

} else {

PMA\_ajaxShowMessage(data.error, false);

}

}); // end $.post()

}); // end copy database

$(document).on('submit', "#change\_db\_charset\_form.ajax", function (event) {

event.preventDefault();

var $form = $(this);

PMA\_prepareForAjaxRequest($form);

PMA\_ajaxShowMessage(PMA\_messages.strChangingCharset);

$.post($form.attr('action'), $form.serialize() + "&submitcollation=1", function (data) {

if (typeof data !== 'undefined' && data.success === true) {

PMA\_ajaxShowMessage(data.message);

} else {

PMA\_ajaxShowMessage(data.error, false);

}

}); // end $.post()

}); // end change charset

$(document).on('click', "#drop\_db\_anchor.ajax", function (event) {

event.preventDefault();

var question = PMA\_messages.strDropDatabaseStrongWarning + ' ';

question += PMA\_sprintf(

PMA\_messages.strDoYouReally,

'DROP DATABASE `' + escapeHtml(PMA\_commonParams.get('db') + '`')

);

var params = {

'is\_js\_confirmed': '1',

'ajax\_request': true,

'token': PMA\_commonParams.get('token')

};

$(this).PMA\_confirm(question, $(this).attr('href'), function (url) {

PMA\_ajaxShowMessage(PMA\_messages.strProcessingRequest);

$.post(url, params, function (data) {

if (typeof data !== 'undefined' && data.success) {

//Database deleted successfully, refresh both the frames

PMA\_reloadNavigation();

PMA\_commonParams.set('db', '');

PMA\_commonActions.refreshMain(

'server\_databases.php',

function () {

PMA\_ajaxShowMessage(data.message);

}

);

} else {

PMA\_ajaxShowMessage(data.error, false);

}

});

});

});

});