

## CdbException

CdbCommand failed to execute the SQL statement: SQLSTATE[HY000]: General error: 1 Can't create/write to file '/tmp/MyogxlwL' (Errcode: 13 - Permission denied). The SQL statement executed was: SELECT qa.qid, qa.attribute, qa.value, qa.language FROM lme\_question\_attributes qa JOIN lme\_questions q ON q.qid = qa.qid AND q.sid=969498 group by qa.qid, qa.attribute, qa.value, qa.language

/var/www/limesurvey/framework/db/CdbCommand.php(543)

```
531         {
532             if($this->_connection->enableProfiling)
533                 Yii::endProfile('system.db.CdbCommand.query('.$this->getText().$par.')', 'system.db.CdbCommand.query');
534
535             $errorInfo=$e instanceof PDOException ? $e->errorInfo : null;
536             $message=$e->getMessage();
537             Yii::log(Yii::t('yii', 'CdbCommand::{method}() failed: {error}'. 'The SQL statement executed was: {sql}',
538                 array('{method}'=>$method, '{error}'=>$message, '{sql}'=>$this->getText().$par)), CLogger::LEVEL_ERROR, 'system.db.CdbCommand');
539
540             if(Yii_DEBUG)
541                 $message.=' The SQL statement executed was: '.$this->getText().$par;
542
543             throw new CdbException(Yii::t('yii', 'CdbCommand failed to execute the SQL statement: {error}',
544                 array('{error}'=>$message)), (int)$e->getCode(), $errorInfo);
545         }
546     }
547
548     /**
549      * Builds a SQL SELECT statement from the given query specification.
550      * @param array $query the query specification in name-value pairs. The following
551      *   query options are supported: {@link select}, {@link distinct}, {@link from},
552      *   {@link where}, {@link join}, {@link group}, {@link having}, {@link order},
553      *   {@link limit}, {@link offset} and {@link union}.
554      * @throws CdbException if "from" key is not present in given query parameter
555      * @return string the SQL statement
```

## Stack Trace

#0 [var/www/limesurvey/framework/db/CdbCommand.php(377): CdbCommand->queryInternal("", 0, array())]

```
372     * @return CdbDataReader the reader object for fetching the query result
373     * @throws CException execution failed
374     */
375     public function query($params=array())
376     {
377         return $this->queryInternal('', 0, $params);
378     }
379
380     /**
381     * Executes the SQL statement and returns all rows.
382     * @param boolean $fetchAssociative whether each row should be returned as an associated array with
```

#1 [var/www/limesurvey/application/helpers/export\_helper.php(730): CdbCommand->query()]

```
725         foreach($results as $row){
726             $result[] = $row->decrypt()->attributes;
727         }
728
729     } else {
730         $queryResult = Yii::app()->db->createCommand($query)->limit($iChunkSize, $iStart)->query();
731         $result = $queryResult->readAll();
732     }
733
734     if ($iStart == 0 && safeCount($result) > 0) {
735         $excludes = array_flip($excludes); //Flip key/value in array for faster checks
```

#2 [var/www/limesurvey/application/helpers/export\_helper.php(880): buildXMLFromQuery(XMLWriter, "SELECT qa.qid, qa.attribute, qa.value, qa.language FROM ...", "question\_attributes")]

```
875     $query = "SELECT qa.qid, qa.attribute, qa.value, qa.language
876     FROM {{question_attributes}} qa JOIN {{questions}} q ON q.qid = qa.qid AND q.sid={$iSurveyID}
877     group by qa.qid, qa.attribute, qa.value, qa.language";
878
879     buildXMLFromQuery($xmlWriter, $query, 'question_attributes');
880
881     if (!isset($excludes['quotas'])) {
882         //Quota
883         $query = "SELECT {{quota}}.*
884         FROM {{quota}}";
```

#3 [var/www/limesurvey/application/helpers/export\_helper.php(956): surveyGetXMLStructure("969498", XMLWriter, array())]

```
951     $surveyLanguages[] = Survey::model()->findByPk($iSurveyID)->language;
952     foreach ($surveyLanguages as $surveyLanguage) {
953         $xml->writeElement('language', $surveyLanguage);
954     }
955     $xml->endElement();
956     surveyGetXMLStructure($iSurveyID, $xml, $excludes);
957     // survey theme configuration - db values
958     surveyGetThemeConfiguration($iSurveyID, $xml, false, 'themes');
959     // survey theme configuration - inherited values
960     surveyGetThemeConfiguration($iSurveyID, $xml, true, 'themes_inherited');
961     $xml->endElement(); // close columns
```

#4 [var/www/limesurvey/application/controllers/admin/surveyadmin.php(1228): surveyGetXMLData("969498", array())]

```
1223         } elseif (!Permission::model()->hasSurveyPermission($iSurveyID, 'surveycontent', 'export') && !Permission::model()->hasSurveyPermission($iSurveyID, 'surveycontent', 'export')) {
1224             $aData['errorMessage'] = qT("We are sorry but you don't have permissions to do this.");
1225             $aData['bfailed'] = true;
1226         } else {
1227             Yii::app()->loadHelper('export');
1228             $copySurveyData = surveyGetXMLData($iSurveyID, $aExcludes);
1229         }
1230     }
1231
1232     // Now, we have the survey : start importing
1233     Yii::app()->loadHelper('admin/import');
```

#5 unknown(0): SurveyAdmin->copy()

#6 [var/www/limesurvey/framework/web/actions/CAction.php(109): ReflectionMethod->invokeArgs(SurveyAdmin, array())]

```
104         elseif($param->isDefaultValueAvailable())
105             $ps[]=$param->getDefaultValue();
106         else
107             return false;
108     }
109     $method->invokeArgs($object,$ps);
110     return true;
111 }
112 }
```

#7 [var/www/limesurvey/application/core/Survey\_Common\_Action.php(86): CAction->runWithParamsInternal(SurveyAdmin, ReflectionMethod, array("sa" => "copy", "sAction" => "copysurvey"))]

```
81         $oMethod = new ReflectionMethod($this, $$Default);
82     }
83
84     // We're all good to go, let's execute it
85     // runWithParamsInternal would automatically get the parameters of the method and populate them as required with the params
86     return parent::runWithParamsInternal($this, $oMethod, $params);
87 }
88
89 /**
90  * Some functions have different parameters, which are just an alias of the
91  * usual parameters we're getting in the url. This function just populates
```

#8 [var/www/limesurvey/framework/web/CController.php(308): Survey\_Common\_Action->runWithParams(array("sa" => "copy", "sAction" => "copysurvey"))]

```
303     {
304         $priorAction=$this->_action;
305         $this->_action=$action;
306         if($this->beforeAction($action))
307         {
```

