dBase

Introduction

These functions allow you to access records stored in dBase-format (dbf) databases.

dBase files are simple sequential files of fixed length records. Records are appended to the end of the file and delete records are kept until you call dbase_pack().

The types of dBase fields available are:

Available types of fields

Field	dBase Type	Format	Additiona
M	Memo	n/a	This type i PHP, such ignored
D	Date	*YYYYMMDD*	The field le
N	Number	A number	You must and a prec of digits af point)
C	String	A string	You must When retri string will with space declared le
L	Boolean	*T* or *Y* for **TRUE**, *F* or *N* for **FALSE**	Stored and integer (1
F	Float	A float number	Support fo was added

Warning

There is no support for indexes or memo fields. There is no support for locking, too. Two concurrent web server processes modifying the same dBase file will very likely ruin your database.

We recommend that you do not use dBase files as your production database. Choose any real SQL server instead; » MySQL or » Postgres are common choices with PHP. dBase support is here to allow you to import and export data to and from your web database, because the file format is commonly understood by Windows spreadsheets and organizers.

Installing/Configuring

Requirements

No external libraries are needed to build this extension.

Installation

In order to enable the bundled dbase library and to use these functions, you must compile PHP with the *--enable-dbase* option.

Runtime Configuration

This extension has no configuration directives defined in *php.ini*.

Resource Types

This extension has no resource types defined.

Predefined Constants

This extension has no constants defined.

dBase Functions

Examples

Many examples in this reference require a dBase database. We will use */tmp/test.dbf* that will be created in the example of dbase_create().

dbase_add_record

dbase_add_record -- Adds a record to a database

Description

bool **dbase_add_record** (int `$dbase_identifier`, array `$record`)

Adds the given data to the database.

Parameters

`dbase_identifier`
The database link identifier, returned by dbase_open() or dbase_create().

`record`
An indexed array of data. The number of items must be equal to the number of fields in the database, otherwise dbase_add_record() will fail.

Note
If you're using dbase_get_record() return value for this parameter, remember to reset the key named *deleted*.

Return Values

Returns **TRUE** on success or **FALSE** on failure.

Examples

Example #1 - Inserting a record in a dBase database

```
<?php

// open in read-write mode
$db = dbase_open('/tmp/test.dbf', 2);

if ($db) {
 dbase_add_record($db, array(
     date('Ymd'),
     'Maxim Topolov',
     '23',
     'max@example.com',
     'T'));
```

```
 dbase_close($db);
}

?>
```

See Also

- dbase_delete_record()
- dbase_replace_record()

dbase_close

dbase_close -- Closes a database

Description

bool **dbase_close** (int `$dbase_identifier`)

Closes the given database link identifier.

Parameters

`dbase_identifier`
The database link identifier, returned by dbase_open() or dbase_create().

Return Values

Returns **TRUE** on success or **FALSE** on failure.

Examples

Example #2 - Closing a dBase database file

```
<?php

// open in read-only mode
$db = dbase_open('/tmp/test.dbf', 0);

if ($db) {
 // read some data ..

 dbase_close($db);
}

?>
```

See Also

- dbase_open()
- dbase_create()

dbase_create

dbase_create -- Creates a database

Description

int **dbase_create** (string `$filename`, array `$fields`)

dbase_create() creates a dBase database with the given definition.

Note
When safe mode is enabled, PHP checks whether the files or directories being operated upon have the same UID (owner) as the script that is being executed.

Note
This function is affected by open_basedir.

Parameters

`filename`
: The name of the database. It can be a relative or absolute path to the file where dBase will store your data.

`fields`
: An array of arrays, each array describing the format of one field of the database. Each field consists of a name, a character indicating the field type, and optionally, a length, and a precision.

Note
The fieldnames are limited in length and must not exceed 10 chars.

Return Values

Returns a database link identifier if the database is successfully created, or **FALSE** if an error occurred.

Examples

Example #3 - Creating a dBase database file

```
<?php

// database "definition"
$def = array(
 array("date",     "D"),
 array("name",     "C",  50),
 array("age",      "N",   3, 0),
 array("email",    "C", 128),
 array("ismember", "L")
);

// creation
if (!dbase_create('/tmp/test.dbf', $def)) {
 echo "Error, can't create the database\n";
}

?>
```

See Also

- dbase_open()
- dbase_close()

dbase_delete_record

dbase_delete_record -- Deletes a record from a database

Description

bool **dbase_delete_record** (int `$dbase_identifier`, int `$record_number`)

Marks the given record to be deleted from the database.

Note
To actually remove the record from the database, you must also call dbase_pack().

Parameters

`dbase_identifier`
: The database link identifier, returned by dbase_open() or dbase_create().

`record_number`
: An integer which spans from 1 to the number of records in the database (as returned by dbase_numrecords()).

Return Values

Returns **TRUE** on success or **FALSE** on failure.

See Also

- dbase_add_record()
- dbase_replace_record()

dbase_get_header_info

dbase_get_header_info -- Gets the header info of a database

Description

array **dbase_get_header_info** (int `$dbase_identifier`)

Returns information on the column structure of the given database link identifier.

Parameters

`dbase_identifier`
The database link identifier, returned by dbase_open() or dbase_create().

Return Values

An indexed array with an entry for each column in the database. The array index starts at 0.

Each array element contains an associative array of column information, as described here:
name
The name of the column

type
The human-readable name for the dbase type of the column (i.e. date, boolean, etc.)

length
The number of bytes this column can hold

precision
The number of digits of decimal precision for the column

format
A suggested printf() format specifier for the column

offset
The byte offset of the column from the start of the row

If the database header information cannot be read, **FALSE** is returned.

Examples

Example #4 - Showing header information for a dBase database file

```
<?php
// Path to dbase file
$db_path = "/tmp/test.dbf";

// Open dbase file
$dbh = dbase_open($db_path, 0)
 or die("Error! Could not open dbase database file '$db_path'.");

// Get column information
$column_info = dbase_get_header_info($dbh);

// Display information
print_r($column_info);
?>
```

dbase_get_record_with_names

dbase_get_record_with_names -- Gets a record from a database as an associative array

Description

array **dbase_get_record_with_names** (int `$dbase_identifier`, int `$record_number`)

Gets a record from a dBase database as an associative array.

Parameters

`dbase_identifier`
The database link identifier, returned by dbase_open() or dbase_create().

`record_number`
The index of the record.

Return Values

An associative array with the record. This will also include a key named *deleted* which is set to 1 if the record has been marked for deletion (see dbase_delete_record()).

Each field is converted to the appropriate PHP type, except:

- Dates are left as strings.
- Integers that would have caused an overflow (> 32 bits) are returned as strings.

On error, dbase_get_record_with_names() will return **FALSE**.

Examples

Example #5 - Listing all the registered members in the database

```
<?php
// open in read-only mode
$db = dbase_open('/tmp/test.dbf', 0);

if ($db) {
 $record_numbers = dbase_numrecords($db);
 for ($i = 1; $i <= $record_numbers; $i++) {
     $row = dbase_get_record_with_names($db, $i);
     if ($row['ismember'] == 1) {
         echo "Member #$i: " . trim($row['name']) . "\n";
```

```
    }
  }
}
?>
```

See Also

- dbase_get_record()

dbase_get_record

dbase_get_record -- Gets a record from a database as an indexed array

Description

array **dbase_get_record** (int `$dbase_identifier`, int `$record_number`)

Gets a record from a database as an indexed array.

Parameters

`dbase_identifier`
The database link identifier, returned by dbase_open() or dbase_create().

`record_number`
The index of the record.

Return Values

An indexed array with the record. This array will also include an associative key named *deleted* which is set to 1 if the record has been marked for deletion (see dbase_delete_record()).

Each field is converted to the appropriate PHP type, except:

- Dates are left as strings.
- Integers that would have caused an overflow (> 32 bits) are returned as strings.

On error, dbase_get_record() will return **FALSE**.

See Also

- dbase_get_record_with_names()

dbase_numfields

dbase_numfields -- Gets the number of fields of a database

Description

int **dbase_numfields** (int `$dbase_identifier`)

Gets the number of fields (columns) in the specified database.

Note
Field numbers are between 0 and dbase_numfields($db)-1, while record numbers are between 1 and dbase_numrecords($db).

Parameters

`dbase_identifier`
The database link identifier, returned by dbase_open() or dbase_create().

Return Values

The number of fields in the database, or **FALSE** if an error occurs.

Examples

Example #6 - dbase_numfields() **Example**

```
<?php

$rec = dbase_get_record($db, $recno);
$nf  = dbase_numfields($db);
for ($i = 0; $i < $nf; $i++) {
 echo $rec[$i], "\n";
}

?>
```

See Also

- dbase_numrecords()

dbase_numrecords

dbase_numrecords -- Gets the number of records in a database

Description

int **dbase_numrecords** (int `$dbase_identifier`)

Gets the number of records (rows) in the specified database.

Note
Record numbers are between 1 and dbase_numrecords($db), while field numbers are between 0 and dbase_numfields($db)-1.

Parameters

`dbase_identifier`
The database link identifier, returned by dbase_open() or dbase_create().

Return Values

The number of records in the database, or **FALSE** if an error occurs.

Examples

Example #7 - Looping over all the records of the database

```
<?php

// open in read-only mode
$db = dbase_open('/tmp/test.dbf', 0);

if ($db) {
 $record_numbers = dbase_numrecords($db);
 for ($i = 1; $i <= $record_numbers; $i++) {
     // do something here, for each record
 }
}

?>
```

See Also

- **dbase_num_fields()**

dbase_open

dbase_open -- Opens a database

Description

int **dbase_open** (string `$filename`, int `$mode`)

dbase_open() opens a dBase database with the given access mode.

Note
When safe mode is enabled, PHP checks whether the files or directories being operated upon have the same UID (owner) as the script that is being executed.

Note
This function is affected by open_basedir.

Parameters

`filename`
: The name of the database. It can be a relative or absolute path to the file where dBase will store your data.

`mode`
: An integer which correspond to those for the *open()* system call (Typically 0 means read-only, 1 means write-only, and 2 means read and write).

Note
You can't open a dBase file in write-only mode as the function will fail to read the headers information and thus you can't use 1 as *`mode`*.

Examples

Example #8 - Opening a dBase database file

```
<?php
```

```
// open in read-only mode
$db = dbase_open('/tmp/test.dbf', 0);

if ($db) {
 // read some data ..

 dbase_close($db);
}

?>
```

Return Values

Returns a database link identifier if the database is successfully opened, or **FALSE** if an error occurred.

See Also

- dbase_create()
- dbase_close()

dbase_pack

dbase_pack -- Packs a database

Description

bool **dbase_pack** (int `$dbase_identifier`)

Packs the specified database by permanently deleting all records marked for deletion using dbase_delete_record().

Parameters

`dbase_identifier`
The database link identifier, returned by dbase_open() or dbase_create().

Return Values

Returns **TRUE** on success or **FALSE** on failure.

Examples

Example #9 - Emptying a dBase database

```
<?php

// open in read-write mode
$db = dbase_open('/tmp/test.dbf', 2);

if ($db) {
 $record_numbers = dbase_numrecords($db);
 for ($i = 1; $i <= $record_numbers; $i++) {
     dbase_delete_record($db, $i);
 }
 // expunge the database
 dbase_pack($db);
}

?>
```

See Also

- dbase_delete_record()

dbase_replace_record

dbase_replace_record -- Replaces a record in a database

Description

bool **dbase_replace_record** (int `$dbase_identifier`, array `$record`, int `$record_number`)

Replaces the given record in the database with the given data.

Parameters

`dbase_identifier`
The database link identifier, returned by dbase_open() or dbase_create().

`record`
An indexed array of data. The number of items must be equal to the number of fields in the database, otherwise dbase_replace_record() will fail.

Note
If you're using dbase_get_record() return value for this parameter, remember to reset the key named *deleted*.

`record_number`
An integer which spans from 1 to the number of records in the database (as returned by dbase_numrecords()).

Return Values

Returns **TRUE** on success or **FALSE** on failure.

Examples

Example #10 - Updating a record in the database

```
<?php

// open in read-write mode
$db = dbase_open('/tmp/test.dbf', 2);

if ($db) {
```

```
 // gets the old row
 $row = dbase_get_record_with_names($db, 1);

 // remove the 'deleted' entry
 unset($row['deleted']);

 // Update the date field with the current timestamp
 $row['date'] = date('Ymd');

 // Replace the record
 dbase_replace_record($db, $row, 1);
 dbase_close($db);
}

?>
```

See Also

- dbase_add_record()
- dbase_delete_record()