sqlite_array_query

Descripción

Ejecuta una consulta contra una base de datos y devuelve el resultado en forma de matriz

array **sqlite_array_query** (resource manejador_bd, string consulta [, int tipo_resultado [, bool decodificar_binario]])

array **sqlite_array_query** (string consulta, resource manejador_bd [, int tipo_resultado [, bool decodificar_binario]])

Método que sigue el estilo orientado a objetos:

class **SQLiteDatabase** {

array **arrayQuery** (string consulta [, int tipo_resultado [, bool decodificar_binario]])

}

Lista de parámetros

consulta

La consulta que se quiere ejecutar.

manejador_bd

El recurso que identifica la base de datos SQLite (y que es el que devuelve la función **sqlite_open()**). Este parámetro no se requiere cuando se emplea el método orientado a objetos.

tipo_resultado

Este parametro es opcional y controla la forma en la que se indexa el array devuelto. Las constantes que se pueden utilizar son:

- **SQLITE_ASSOC** el índice del array será el nombre de los campos.
- **SQLITE_NUM** el indice del array devuelto sera numerico.
- **SQLITE_BOTH** el indice del array devuelto sera asociativo y numerico.

decodificar_binario

Por defecto esta a true. Estando a true se decoficaran los datos codificados con **sqlite_escape_string()**. Si se coloca a false podemos tener problemas con otras bases de datos creadas con SQLite.

Valores retornados

Devuelve una matriz que contiene todo el resultado completo o **FALSE** en cualquier otro caso.

sqlite_array_query() está especialmente preparada para las consultas que devuelven 45 filas o menos. Si el resultado es más grande se recomienda emplear la función **sqlite_unbuffered_query()** que tiene un mejor rendimiento con resultados grandes.

Ver también

sqlite_query(), sqlite_fetch_array(), sqlite_fetch_string().

sqlite_busy_timeout

Descripción

Establece la duración del temporizador de ocupado o deshabilita los temporizadores

void **sqlite_busy_timeout** (resource manejador_bd, int milisegundos)

Método que sigue el estilo orientado a objetos:

class **SQLiteDatabase** {

void **busyTimeout** (int milisegundos)

}

Establece la duración máxima, en milisegundos, que SQLite espera para que la base de datos indicada por *manejador_bd* esté lista para ser usada.

Lista de parámetros

manejador_bd

El recurso que identifica la base de datos SQLite (y que es el que devuelve la función **sqlite_open()**). Este parámetro no se requiere cuando se emplea el método orientado a objetos.

milisegundos

El número de milisegundos. Cuando se establece a *0*, se deshabilitan los temporizadores de ocupado y SQLite devuelve el valor *SQLITE_BUSY* de forma inmediata cuando otro proceso ha bloqueado la base de datos para realizar una actualización de datos.

PHP establece el temporizador de ocupado por defecto en 60 segundos a partir de la apertura de la base de datos.

Ver también

sqlite_open()

sqlite_changes

Descripción

Devuelve el número de filas que se han modificado en la última sentencia SQL

int **sqlite_changes** (resource manejador_bd)

Método que sigue el estilo orientado a objetos:

class **SQLiteDatabase** {

int **changes** (void)

}

Devuelve el número de filas que se han modificado en la última sentencia SQL ejecutada en la base de datos indicada con el parámetro *manejador_bd*.

Lista de parámetros

manejador_bd

El recurso que identifica la base de datos SQLite (y que es el que devuelve la función **sqlite_open()**). Este parámetro no se requiere cuando se emplea el método orientado a objetos.

Ver también

sqlite_open()

sqlite_close

Descripción

Cierra una base de datos SQLite abierta

void **sqlite_close** (resource manejador_bd)

Cierra la base de datos indicada por el parámetro *manejador_bd*. Si la conexión era persistente, se cierra y se borra de la lista de conexiones persistentes.

Lista de parámetros

manejador_bd

El manejador de la base de datos SQLite devuelto por la función **sqlite_open()**.

Ver también

sqlite_open(), sqlite_popen()

sqlite_column

Descripción

Obtiene una columna de la fila actual del resultado

mixed **sqlite_column** (resource manejador_resultado, mixed indice_o_nombre [, bool decodificar_binario])

class **SQLiteResult** {

mixed **column** (mixed indice_o_nombre [, bool decodificar_binario])

}class **SQLiteUnbuffered** {

mixed **column** (mixed indice_o_nombre [, bool decodificar_binario])

}

Obtiene el valor de la columna llamada *indice_o_nombre* (si el parámetro es una cadena) o el valor de la columna cuyo índice es *indice_o_nombre* (si el parámetro es un número entero) dentro de la fila actual del resultado identificado por *manejador_resultado*.

Lista de parámetros

manejador_resultado

El identificador del resultado de SQLite. Este parámetro no es obligatorio cuando se emplea el método orientado a objetos.

indice_o_nombre

El índice o nombre de la columna que se quiere obtener.

decodificar_binario

Por defecto esta a true. Estando a true se decoficaran los datos codificados con **sqlite_escape_string()**. Si se coloca a false podemos tener problemas con otras bases de datos creadas con SQLite.

Ver también

sqlite_fetch_string()

sqlite_create_aggregate

Descripción

Registra una FDU (función definida por el usuario) de grupo normal para su uso en sentencias SQL

void **sqlite_create_aggregate** (resource manejador_bd, string nombre_funcion, callback funcion_intermedia, callback funcion_final [, int numero_argumentos])

Método que sigue el estilo orientado a objetos:

class **SQLiteDatabase** {

void **createAggregate** (string nombre_funcion, callback funcion_intermedia, callback funcion_final [, int numero_argumentos])

}

sqlite_create_aggregate() es similar a **sqlite_create_function()** salvo que registrar funciones que se emplean para calcular resultados agrupados de todas las filas del resultado de una consulta.

La diferencia principal entre esta función y **sqlite_create_function()** es que se requieren 2 funciones para manejar los cálculos agrupados. La función indicada en *funcion_intermedia* se llama para cada una de las filas del resultado. La función PHP acumula y guarda el resultado temporal en el llamado "contexto de agregación". Una vez que todas las filas han sido procesadas, se llama a la función *funcion_final*, que obtendrá los resultados intermedios a partir del contexto de agregación y devolverá el resultado calculado. Las funciones de callback deben devolver un tipo de datos entendido por SQLite (**boolean, integer, float, string**).

Lista de parámetros

manejador_bd

El recurso que identifica la base de datos SQLite (y que es el que devuelve la función **sqlite_open()**). Este parámetro no se requiere cuando se emplea el método orientado a objetos.

nombre_funcion

El nombre de la función a utilizar en las sentencias SQL.

funcion_intermedia

La función de callback que se llama para cada una de las filas del resultado.

funcion_final

La función de callback que se llama para realizar los cálculos agrupados a partir de los datos temporales almacenados por la función intermedia que se ejecuta sobre cada una de las filas.

numero_argumentos

Valor destinado al parseador de SQLite si la función de callback acepta un número predeterminado de argumentos.

Ver también

sqlite_Create_function(), sqlite_udf_encode_binary(), sqlite_udf_decode_binary().

sqlite_create_function

Descripción

Registra una FDU (función definida por el usuario) normal para su uso en sentencias SQL

void **sqlite_create_function** (resource manejador_bd, string nombre_funcion, callback callback [, int numero_argumentos])

Método que sigue el estilo orientado a objetos:

class **SQLiteDatabase** {

void **createFunction** (string nombre_funcion, callback callback [, int numero_argumentos])

}

sqlite_create_function() permite registrar una función PHP para que SQLite la emplee como FDU (función definida por el usuario). De esta forma, la función se puede llamar desde las sentencias SQL.

La función FDU se puede utilizar en cualquier sentencia que pueda invocar funciones, como SELECT y UPDATE, además de en triggers.

Lista de parámetros

manejador_bd

El recurso que identifica la base de datos SQLite (y que es el que devuelve la función **sqlite_open()**). Este parámetro no se requiere cuando se emplea el método orientado a objetos.

nombre_funcion

El nombre de la función a utilizar en las sentencias SQL.

callback

Función de callback que se encarga de manejar la función SQL. Una vez que todas las filas han sido procesadas, se llama a la función *funcion_final*, que

obtendrá los resultados intermedios a partir del contexto de agregación y devolverá el resultado calculado

Nota: Las funciones de callback deben devolver un tipo de datos entendido por SQLite (boolean, integer, float, string).

numero_argumentos

Valor destinado al parseador de SQLite si la función de callback acepta un número predeterminado de argumentos.

Ver también

sqlite_Create_aggregate()

sqlite_current

Descripción

Obtiene la fila actual del resultado en forma de matriz

array **sqlite_current** (resource manejador_resultado [, int tipo_resultado [, bool decodificar_binario]])

Método que sigue el estilo orientado a objetos:

class **SQLiteResult** {

array **current** ([int tipo_resultado [, bool decodificar_binario]])

}class **SQLiteUnbuffered** {

array **current** ([int tipo_resultado [, bool decodificar_binario]])

}

sqlite_current() es idéntica a la función **sqlite_fetch_array()** salvo que no se avanza a la siguiente fila del resultado antes de devolver los datos. De esta forma, solamente devuelve los datos de la posición actual.

Lista de parámetros

manejador_resultado

El identificador del resultado de SQLite. Este parámetro no es obligatorio cuando se emplea el método orientado a objetos.

tipo_resultado

Este parametro es opcional y controla la forma en la que se indexa el array devuelto. Las constantes que se pueden utilizar son:

- **SQLITE_ASSOC** el índice del array será el nombre de los campos.
- **SQLITE_NUM** el indice del array devuelto sera numerico.
- **SQLITE_BOTH** el indice del array devuelto sera asociativo y numerico.

decodificar_binario

Por defecto esta a true. Estando a true se decoficaran los datos codificados con **sqlite_escape_string()**. Si se coloca a false podemos tener problemas con otras bases de datos creadas con SQLite.

Valores retornados

Devuelve una matriz que contiene los valores de la fila actual del resultado o **FALSE** si la posición actual se encuentra más allá de la última fila.

La forma de devolver los valores fijados con las constantes **SQLITE_ASSOC** y **SQLITE_BOTH** puede ser especificada en **sqlite.assoc_case**.

Ver también

sqlite_Seek(), sqlite_next, sqlite_fetch_array()

sqlite_error_string

Descripción

Devuelve la descripción del error producido a partir de un código de error

string **sqlite_error_string** (int codigo_error)

Devuelve una descripción detallada del error producido a partir del código de error proporcionado por *codigo_error* y obtenido con la función **sqlite_last_error()**.

Ver también

sqlite_last_error()

sqlite_escape_string

Descripción

Formatea una cadena de texto para poder usarla como parámetro en una consulta

string **sqlite_escape_string** (string cadena)

sqlite_escape_string() modifica la cadena proporcionada en el parámetro *cadena* de forma que se escapan los caracteres especiales y así la cadena está lista para emplearse en las sentencias SQL de SQLite. Los cambios realizados incluyen doblar las comillas simples (') y comprobar que no existen caracteres binarios inseguros que se vayan a emplear en la cadena de consulta SQL.

Si la *cadena* contiene caracteres de tipo *NULL* o si empieza con un carácter cuyo valor ordinal sea *0x01*, PHP aplica un esquema de codificación binaria para asegurar que los datos binarios pueden almacenarse (y recuperarse posteriormente) con seguridad.

Aunque la codificación permite transformar los datos binarios en seguros, impedirá que se puedan emplear comparaciones como *LIKE* en las columnas que contienen datos binarios. En la práctica sin embargo, este hecho no debería ser un problema ya que en los esquemas de bases datos no se suelen emplear ese tipo de comparaciones con las columnas que almacenan datos binarios (de hecho, es mejor emplear otros medios para almacenar los datos binarios, como por ejemplo archivos en el sistema).

No se debe emplear la función addslashes() para formatear las cadenas que se utilizan en las consultas de SQLite, ya que puede provocar comportamientos indeseados a la hora de recuperar los datos.

No se debe emplear esta función para codificar los datos devueltos por FDU (funciones definidas por el usuario) creadas con las funciones sqlite_create_function() o sqlite_create_aggregate(). Se debe emplear la función sqlite_udf_encode_binary() en su lugar.

Ver también

sqlite_udf_encode_binary()

sqlite_exec

Descripción

Ejecuta una consulta que no produce resultado

bool **sqlite_exec** (resource manejador_bd, string consulta)

bool **sqlite_exec** (string consulta, resource manejador_bd)

Ambas alternativas son admitidas por SQLite para mantener la portabilidad con otros sistemas como MySQL

Método que sigue el estilo orientado a objetos:

class **SQLiteDatabase** {

bool **exec** (string consulta)

}

Ejecuta la sentencia SQL indicada por el parámetro *consulta* en la base de datos identificada por el parámetro *manejador_bd*.

SQLite permite ejecutar múltiples consultas seguidas separadas por un punto y coma. De esta forma, se pueden ejecutar de una vez una serie de consultas SQL que se han cargado por ejemplo de un archivo o que se han incluido en un script.

Lista de parámetros

consulta

La consulta que se quiere ejecutar.

manejador_bd

El recurso que identifica la base de datos SQLite (y que es el que devuelve la función **sqlite_open()**). Este parámetro no se requiere cuando se emplea el método orientado a objetos.

Valores retornados

La función devuelve un resultado booleano: **TRUE** si tiene éxito y **FALSE** en caso contrario. Si se requiere que la consulta ejecutada devuelva las filas en el resultado, se debe emplear la función **sqlite_query()**.

Ver también

sqlite_query(),sqlite_unbuffered_query(),sqlite_array_query()

sqlite_factory

Abre una base de datos SQLite y devuelve un objeto SQLiteDatabase

Descripción

SQLiteDatabase **sqlite_factory** (string nombre_archivo [, int modo [, string &mensaje_error]])

sqlite_factory() se comporta de forma similar a la función **sqlite_open()** ya que abre una base de datos SQLite o intenta crearla si no existe. La diferencia es que se devuelve un objeto de tipo **SQLiteDataBase** y no un manejador de base de datos. Consulte la documentación de la función **sqlite_open()** para obtener mas información sobre su uso.

Lista de parámetros

nombre_archivo

El nombre de archivo de la base de datos de SQLite.

modo

El modo del archivo, que se emplea para abrir la base de datos en modo solo lectura. Actualmente, la librería sqlite ignora este parámetro. El valor por defecto en formato octal es *0666*, que además es el valor recomendado.

mensaje_error

Se pasa por referencia y contiene un mensaje de error descriptivo que explica el motivo por el que no se pudo abrir la base de datos.

Valores retornados

Devuelve un objeto de tipo SQLiteDatabase en caso de éxito o `NULL` si se ha producido un error.

Ver también

sqlite_open(), sqlite_popen()

sqlite_fetch_all

Descripción

Obtiene todas las filas del resultado en forma de matriz de matrices

array **sqlite_fetch_all** (resource manejador_resultado [, int tipo_resultado [, bool decodificar_binario]])

Método que sigue el estilo orientado a objetos:

class **SQLiteResult** {

array **fetchAll** ([int tipo_resultado [, bool decodificar_binario]])

}class **SQLiteUnbuffered** {

array **fetchAll** ([int tipo_resultado [, bool decodificar_binario]])

}

sqlite_fetch_all() devuelve una matriz que contiene todos los valores del resultado representado por el parámetro *manejador_resultado*. Su funcionamiento es similar a emplear la función **sqlite_query()** (o **sqlite_unbuffered_query()**) y después llamar a **sqlite_fetch_array()** para cada una de las filas del resultado.

Lista de parámetros

manejador_resultado

El identificador del resultado de SQLite. Este parámetro no es obligatorio cuando se emplea el método orientado a objetos.

tipo_resultado

Este parametro es opcional y controla la forma en la que se indexa el array devuelto. Las constantes que se pueden utilizar son:

- **SQLITE_ASSOC** el índice del array será el nombre de los campos.
- **SQLITE_NUM** el indice del array devuelto sera numerico.
- **SQLITE_BOTH** el indice del array devuelto sera asociativo y numerico.

decodificar_binario

Por defecto esta a true. Estando a true se decoficaran los datos codificados con **sqlite_escape_string()**. Si se coloca a false podemos tener problemas con otras bases de datos creadas con SQLite.

Valores retornados

Devuelve una matriz que contiene las filas restantes del resultado. Si se llama a la función justo después de **sqlite_query()**, devuelve todas las filas. Si se llama después de la función **sqlite_fetch_array()**, devuelve el resto. Si el resultado no contiene filas, devuelve una matriz vacía.

Ver también

sqlite_fetch_array()

sqlite_fetch_array

Descripción

Obtiene la siguiente fila del resultado en forma de matriz

array **sqlite_fetch_array** (resource manejador_resultado [, int tipo_resultado [, bool decodificar_binario]])

Método que sigue el estilo orientado a objetos:

class **SQLiteResult** {

array **fetch** ([int tipo_resultado [, bool decodificar_binario]])

}class **SQLiteUnbuffered** {

array **fetch** ([int tipo_resultado [, bool decodificar_binario]])

}

Obtiene la siguiente fila del resultado identificado mediante el parámetro *manejador_resultado*. Si no existen mas filas, devuelve **FALSE**, en cualquier otro caso, devuelve una matriz asociativa que contiene los datos de la fila.

Lista de parámetros

manejador_resultado

El identificador del resultado de SQLite. Este parámetro no es obligatorio cuando se emplea el método orientado a objetos.

manejador_bd

El recurso que identifica la base de datos SQLite (y que es el que devuelve la función **sqlite_open()**). Este parámetro no se requiere cuando se emplea el método orientado a objetos.

tipo_resultado

Este parametro es opcional y controla la forma en la que se indexa el array devuelto. Las constantes que se pueden utilizar son:

- **SQLITE_ASSOC** el índice del array será el nombre de los campos.
- **SQLITE_NUM** el indice del array devuelto sera numerico.
- **SQLITE_BOTH** el indice del array devuelto sera asociativo y numerico.

decodificar_binario

Por defecto esta a true. Estando a true se decoficaran los datos codificados con **sqlite_escape_string()**. Si se coloca a false podemos tener problemas con otras bases de datos creadas con SQLite.

Valores retornados

Devuelve una matriz con los datos de la siguiente fila del resultado o **FALSE** si la siguiente posición se encuentra más allá de la última fila.

Ver también

Sqlite_Array_query(), sqlite_fetch_string()

sqlite_fetch_column_types

Descripción

Obtiene una matriz con los tipos de las columnas de una tabla

array **sqlite_fetch_column_types** (string nombre_tabla, resource manejador_bd [, int tipo_resultado])

Método que sigue el estilo orientado a objetos:

class **SQLiteDatabase** {

array **fetchColumnTypes** (string nombre_tabla [, int tipo_resultado])

}

sqlite_fetch_column_types() devuelve una matriz con los tipos de las columnas de la tabla indicada por el parámetro *nombre_tabla*.

Lista de parámetros

nombre_tabla

El nombre de la tabla que se quiere consultar.

manejador_bd

El recurso que identifica la base de datos SQLite (y que es el que devuelve la función **sqlite_open()**). Este parámetro no se requiere cuando se emplea el método orientado a objetos.

tipo_resultado

El parámetro opcional *result_type* acepta una serie de constantes que controlan como se indexa la matriz devuelta. Si se emplea el valor **SQLITE_ASSOC** se devuelven solamente índices asociativos (esto es, los campos tienen nombre) mientras que el valor **SQLITE_NUM** hace que se devuelvan solamente índices numéricos (los campos se referencian por números ordinales). Por último, el valor **SQLITE_BOTH** hace que se devuelvan tanto los índices numéricos como los asociativos. El valor por defecto para esta función es **SQLITE_ASSOC**.

Valores retornados

Devuelve una matriz con los tipos de las columnas de la tabla o **FALSE** en caso de error.

sqlite_fetch_object

Descripción

Obtiene la siguiente fila del resultado en forma de objeto

object **sqlite_fetch_object** (resource result [, string class_name [, array ctor_params [, bool decode_binary]]])

Método que sigue el estilo orientado a objetos:

class **SQLiteResult** {

object **fetchObject** ([string class_name [, array ctor_params [, bool decode_binary]]])

}class **SQLiteUnbuffered** {

object **fetchObject** ([string class_name [, array ctor_params [, bool decode_binary]]])

}

sqlite_fetch_single

Descripción

Obtiene la primera columna del resultado en forma de cadena

string **sqlite_fetch_single** (resource manejador_resultado [, bool decodificar_binario])

Método que sigue el estilo orientado a objetos:

class **SQLiteResult** {

string **fetchSingle** ([bool decodificar_binario])

}class **SQLiteUnbuffered** {

string **fetchSingle** ([bool decodificar_binario])

}

sqlite_fetch_single() es idéntica a la función **sqlite_fetch_array()** salvo que devuelve el valor de la primera columna del resultado.

Esta es la forma óptima de obtener datos cuando solamente se requieren los valores de una columna del resultado.

Lista de parámetros

manejador_resultado

El identificador del resultado de SQLite. Este parámetro no es obligatorio cuando se emplea el método orientado a objetos.

decodificar_binario

When the *decode_binary* parameter is set to **TRUE** (the default), PHP will decode the binary encoding it applied to the data if it was encoded using the **sqlite_escape_string()**. You should normally leave this value at its default, unless you are interoperating with databases created by other sqlite capable applications.

Ver también

sqlite_fetch_array()

sqlite_fetch_string

Descripción

Esta función es un alias de la función **sqlite_fetch_single()**.

sqlite_field_name

Descripción

Obtiene el nombre de un campo

string **sqlite_field_name** (resource manejador_resultado, int indice_campo)

Método que sigue el estilo orientado a objetos:

class **SQLiteResult** {

string **fieldName** (int indice_campo)

}class **SQLiteUnbuffered** {

string **fieldName** (int indice_campo)

}

A partir del número de columna (*indice_campo*) **sqlite_field_name()** devuelve el nombre de ese campo dentro del resultado indicado con *manejador_resultado*.

Lista de parámetros

manejador_resultado

El identificador del resultado de SQLite. Este parámetro no es obligatorio cuando se emplea el método orientado a objetos.

indice_campo

El número de columna dentro del resultado.

Valores retornados

Devuelve el nombre del campo del resultado de SQLite a partir de su número de columna o **FALSE** si se ha producido un error.

sqlite_has_more

Descripción

Indica si existen más filas disponibles

bool **sqlite_has_more** (resource manejador_resultado)

Averigua si existen más filas disponibles para el resultado indicado.

Lista de parámetros

manejador_resultado

El identificador del resultado de SQLite.

Valores retornados

Devuelve **TRUE** si hay más filas disponibles en el resultado identificado por el parámetro *manejador_resultado* o **FALSE** en cualquier otro caso.

Ver también

sqlite_num_rows(), sqlite_changes()

sqlite_has_prev

Descripción

Indica si está disponible una fila anterior

bool **sqlite_has_prev** (resource manejador_resultado)

Método que sigue el estilo orientado a objetos:

class **SQLiteResult** {

bool **hasPrev** (void)

}

Averigua si existen filas anteriores disponibles para el resultado indicado.

Lista de parámetros

manejador_resultado

El identificador del resultado de SQLite. Este parámetro no es obligatorio cuando se emplea el método orientado a objetos.

Nota: Esta funcion no puede usarse con resultados que no se encuentren almacenados en un almacenador intermedio (buffer).

Valores retornados

Devuelve **TRUE** si existen filas anteriores disponibles para el resultado identificado mediante el parámetro *manejador_resultado* o **FALSE** en cualquier otro caso.

Ver también

sqlite_prev(),sqlite_has_more(),sqlite_num_rows()

sqlite_key

Descripción

Devuelve el índice de la fila actual

int **sqlite_key** (resource manejador_resultado)

Método que sigue el estilo orientado a objetos:

class **SQLiteResult** {

int **key** (void)

}

sqlite_key() devuelve el índice de la fila actual del resultado almacenado (buffered) indicado por el parámetro *manejador_resultado*.

Lista de parámetros

manejador_resultado

El identificador del resultado de SQLite. Este parámetro no es obligatorio cuando se emplea el método orientado a objetos.

Nota: Esta funcion no puede usarse con resultados que no se encuentren almacenados en un almacenador intermedio (buffer).

Valores retornados

Devuelve el índice de la fila actual del resultado almacenado (buffered) indicado por el parámetro *manejador_resultado*.

Ver también

sqlite_next(), sqlite_current(), sqlite_rewind()

sqlite_last_error

Descripción

Devuelve el código de error del último error producido en la base de datos

int **sqlite_last_error** (resource manejador_bd)

Método que sigue el estilo orientado a objetos:

class **SQLiteDatabase** {

int **ultimo_error** (void)

}

Devuelve el código de error de la última operación realizada en la base de datos identificada mediante el parámetro *manejador_bd*. Para obtener un mensaje de error descriptivo sobre el problema surgido se debe emplear la función **sqlite_error_string()**.

Lista de parámetros

manejador_bd

El recurso que identifica la base de datos SQLite (y que es el que devuelve la función **sqlite_open()**). Este parámetro no se requiere cuando se emplea el método orientado a objetos.

Ver también

sqlite_error_string()

sqlite_last_insert_rowid

Descripción

Devuelve el identificador de fila de la última fila insertada

int **sqlite_last_insert_rowid** (resource manejador_bd)

Método que sigue el estilo orientado a objetos:

class **SQLiteDatabase** {

int **lastInsertRowid** (void)

}

Devuelve el identificador de fila de la última fila insertada en la base de datos identificada mediante el parámetro *manejador_bd*, si se creó como un campo autoincremental.

Lista de parámetros

manejador_bd

El recurso que identifica la base de datos SQLite (y que es el que devuelve la función **sqlite_open()**). Este parámetro no se requiere cuando se emplea el método orientado a objetos.

sqlite_libencoding

Descripción

Devuelve la codificación de la librería SQLite que se está empleando

string **sqlite_libencoding** (void)

La librería SQLite puede compilarse en uno de los 2 modos siguientes: ISO-8859-1 o UTF-8. Esta función permite averiguar qué esquema de codificación se ha definido para la librería que se está utilizando. Cuando se compila con soporte UTF-8, SQLite se encarga de la codificación y decodificación de las secuencias de caracteres multi-byte de UTF-8. Sin embargo, no se encarga por ejemplo de la normalización de los datos y algunas de las operaciones de comparación no se realizan de forma correcta.

Ver también

sqlite_lib_version()

sqlite_libversion

Descripción

Devuelve la versión de la librería SQLite que se está empleando

string **sqlite_libversion** (void)

Devuelve la versión de la librería SQLite que se está empleando

Ver también

sqlite_libenconding()

sqlite_next

Descripción

Se desplaza hasta el siguiente número de fila

bool **sqlite_next** (resource manejador_resultado)

Método que sigue el estilo orientado a objetos:

class **SQLiteResult** {

bool **next** (void)

}class **SQLiteUnbuffered** {

bool **next** (void)

}

sqlite_next() desplaza el manejador de resultado (indicado en el parámetro *manejador_resultado*) hasta la siguiente fila.

Lista de parámetros

manejador_resultado

El identificador del resultado de SQLite. Este parámetro no es obligatorio cuando se emplea el método orientado a objetos.

Valores retornados

Devuelve **TRUE** en caso de éxito o **FALSE** si no existen más filas.

Ver también

sqlite_seek(), sqlite_current(),sqlite_rewind()

sqlite_num_fields

Descripción

Obtiene el número de campos de un resultado

int **sqlite_num_fields** (resource manejador_resultado)

Método que sigue el estilo orientado a objetos:

class **SQLiteResult** {

int **numFields** (void)

}class **SQLiteUnbuffered** {

int **numFields** (void)

}

Devuelve el número de campos del resultado identificado por el parámetro *manejador_resultado*.

Lista de parámetros

manejador_resultado

El identificador del resultado de SQLite. Este parámetro no es obligatorio cuando se emplea el método orientado a objetos.

Ver también

sqlite_changes(), sqlite_num_rows()

sqlite_num_rows

Description

Obtiene el número de filas de un resultado almacenado

int **sqlite_num_rows** (resource manejador_resultado)

Método que sigue el estilo orientado a objetos:

class **SQLiteResult** {

int **numRows** (void)

}

Devuelve el número de filas del resultado almacenado (buffered) identificado por el parámetro *manejador_resultado* set.

Lista de parámetros

manejador_resultado

El identificador del resultado de SQLite. Este parámetro no es obligatorio cuando se emplea el método orientado a objetos.

Nota: Esta funcion no puede usarse con resultados que no se encuentren almacenados en un almacenador intermedio (buffer).

Ver también

sqlite_changes(), sqlite_query(), sqlite_num_fields()

sqlite_open

Descripción

Abre una base de datos de SQLite y la crea si no existía

resource **sqlite_open** (string nombre_archivo [, int modo [, string &mensaje_error]])

(Constructor) según el estilo orientado a objetos:

class **SQLiteDatabase** {

__construct (string nombre_archivo [, int modo [, string &mensaje_error]])

}

Abre una base de datos de SQLite o la crea si no existía

Lista de parámetros

nombre_archivo

El nombre de archivo de la base de datos de SQLite. Si el archivo no existe, SQLite intenta crearlo. PHP debe tener permisos de escritura en el archivo si se van a insertar datos, si el esquema de la base de datos se modifica o si se va a crear la propia base de datos por no existir.

modo

El modo del archivo, que se emplea para abrir la base de datos en modo solo lectura. Actualmente, la librería sqlite ignora este parámetro. El valor por defecto en formato octal es *0666*, que además es el valor recomendado.

mensaje_error

Se pasa por referencia y contiene un mensaje de error descriptivo que explica el motivo por el que no se pudo abrir la base de datos.

Valores retornados

Devuelve un recurso en caso de éxito (que realmente es un manejador de base de datos) o FALSE en caso de error.

Nota: Desde la versión 2.8.2 de la librería SQLite, se puede indicar como *nombre_archivo* el valor *:memory:*, con lo que se creará la base de datos en la memoria del sistema. Este tipo de bases de datos son útiles cuando se necesita

realizar un procesamiento temporal, ya que las bases de datos residentes en memoria se eliminan cuando termina la ejecución del proceso. También puede ser útil cuando se emplea junto con la sentencia SQL *ATTACH DATABASE*, de forma que se puedan cargar bases de datos y mover y consultar datos entre ellas.

Ver también

sqlite_popen(), sqlite_close(), sqlite_factory()

sqlite_popen

Descripción

Abre una base de datos de SQLite de forma persistente y la crea si no existía

resource **sqlite_popen** (string nombre_archivo [, int modo [, string &mensaje_error]])

Esta función se comporta de la misma forma que **sqlite_open()**, salvo que emplea el mecanismo de persistencia de recursos de PHP. Para más información sobre el significado de los parámetros, consulte la página del manual de la función **sqlite_open()**.

sqlite_popen() comprueba en primer lugar si ya existe un manejador creado para el *nombre_archivo* indicado. Si encuentra uno, devuelve ese manejador al script; en caso contrario, abre una nueva conexión con la base de datos.

La ventaja de esta forma de actuar es que no se penaliza el rendimiento del script debido al proceso de creación de la conexión con la base de datos, la lectura de su esquema, etc. Si se emplean SAPIs de servidor web que permiten la persistencia, no se tiene que crear una nueva conexión con la base de datos cada vez que se produce un acceso a la página web. (Los SAPIs que no permiten esta persistencia son el CGI y el CLI).

Nota: Si se emplean conexiones persistentes y se utilizan procesos en segundo plano (por ejemplo, con el crontab) que actualizan la base de datos volviendola a crear (debido a una reconstrucción o copiando una versión actualizada machacando la anterior versión) puede que se produzcan comportamientos no deseados con los manejadores que se crearon para la base de datos que ha sido reemplazada.

Para solucionar este problema, los procesos en segundo plano deberían abrir la base de datos y realizar sus actualizaciones en una transacción.

Lista de parámetros

nombre_archivo

El nombre de archivo de la base de datos de SQLite. Si el archivo no existe, SQLite intenta crearlo. PHP debe tener permisos de escritura en el archivo si se van a insertar datos, si el esquema de la base de datos se modifica o si se va a crear la propia base de datos por no existir.

modo

El modo del archivo, que se emplea para abrir la base de datos en modo solo lectura. Actualmente, la librería sqlite ignora este parámetro. El valor por defecto en formato octal es *0666*, que además es el valor recomendado.

mensaje_error

Se pasa por referencia y contiene un mensaje de error descriptivo que explica el motivo por el que no se pudo abrir la base de datos.

Valores retornados

Devuelve un recurso en caso de éxito (que realmente es un manejador de base de datos) o **FALSE** en caso de error.

Ver también

sqlite_open(), sqlite_close(), sqlite_factory()

sqlite_prev

Se desplaza hasta el anterior número de fila

Descripción

bool **sqlite_prev** (resource manejador_resultado)

Método que sigue el estilo orientado a objetos:

class **SQLiteResult** {

bool **prev** (void)

}

sqlite_prev() desplaza el manejador de resultado (indicado en el parámetro *manejador_resultado*) hasta la anterior fila.

Lista de parámetros

manejador_resultado

El identificador del resultado de SQLite. Este parámetro no es obligatorio cuando se emplea el método orientado a objetos.

Nota: Esta funcion no puede usarse con resultados que no se encuentren almacenados en un almacenador intermedio (buffer).

Valores retornados

Devuelve **TRUE** en caso de éxito o **FALSE** si no existen más filas previas.

Ver también

sqlite_has_prev(), sqlite_rewind(), sqlite_next()

sqlite_query

Descripción

Ejecuta una consulta sobre la base de datos y devuelve un manejador del resultado

resource **sqlite_query** (resource manejador_bd, string consulta [, int tipo_resultado])

resource **sqlite_query** (string consulta, resource manejador_bd [, int tipo_resultado])

Método que sigue el estilo orientado a objetos:

class **SQLiteDatabase** {

SQLiteResult **query** (string consulta [, int tipo_resultado])

}

Ejecuta una sentencia SQL dada por el parámetro *consulta* contra una base de datos identificada mediante el parámetro *manejador_bd*.

Lista de parámetros

consulta

La consulta que se quiere ejecutar.

manejador_bd

El recurso que identifica la base de datos SQLite (y que es el que devuelve la función **sqlite_open()**). Este parámetro no se requiere cuando se emplea el método orientado a objetos.

tipo_resultado

Este parametro es opcional y controla la forma en la que se indexa el array devuelto. Las constantes que se pueden utilizar son:

- **SQLITE_ASSOC** el índice del array será el nombre de los campos.
- **SQLITE_NUM** el indice del array devuelto sera numerico.
- **SQLITE_BOTH** el indice del array devuelto sera asociativo y numerico.

Valores retornados

La función devuelve el manejador del resultado o **FALSE** si se produce un error. Para las consultas que devuelven una serie de filas, el manejador del resultado se puede emplear con funciones como **sqlite_fetch_array()** y **sqlite_seek()**.

Independientemente del tipo de consulta, la función devuelve **FALSE** si la consulta ha producido un error.

sqlite_query() devuelve un manejador de resultado que referencia un resultado almacenado (buffered) y que se puede recorrer hacia delante y hacia detrás. Este tipo de resultado es útil para consultas pequeñas en las que es necesario recorrer el resultado en cualquier sentido para poder acceder de forma directa a cualquiera de las filas. Los resultados almacenados reservan la memoria suficiente como para almacenar todo el resultado y el manejador no se devuelve hasta que se han obtenido todas las filas. Si solamente es necesario el acceso secuencial a los resultados, se recomienda emplear la función **sqlite_unbuffered_query()**, que tiene un rendimiento muy superior, ya que el resultado no es almacenado (unbuffered).

Nota: SQLite permite ejecutar múltiples consultas seguidas separadas por un punto y coma. De esta forma, se pueden ejecutar de una vez una serie de consultas SQL que se han cargado por ejemplo de un archivo o que se han incluido en un script. De todas formas, este funcionamiento solo es válido cuando el resultado de una consulta no se utiliza en las demás. Si se utiliza el resultado, entonces solamente se ejecuta la primera sentencia SQL. La función sqlite_exec() siempre ejecuta múltiples consultas SQL.

Ver también

sqlite_unbuffered_query(), sqlite_array_query()

sqlite_rewind

Descripción

Se desplaza hasta el primer número de fila

bool **sqlite_rewind** (resource manejador_resultado)

Método que sigue el estilo orientado a objetos:

class **SQLiteResult** {

bool **rewind** (void)

}

sqlite_rewind() desplaza el resultado identificado por el parámetro *manejador_resultado* hasta el primer número de fila.

Lista de parámetros

manejador_resultado

El identificador del resultado de SQLite. Este parámetro no es obligatorio cuando se emplea el método orientado a objetos.

Nota: Esta funcion no puede usarse con resultados que no se encuentren almacenados en un almacenador intermedio (buffer).

Valores retornados

Devuelve **FALSE** si no hay filas en el resultado, o **TRUE** en cualquier otro caso.

Ver también

sqlite_next(), sqlite_current(), sqlite_seek()

sqlite_seek

Se desplaza hasta un determinado número de fila de un resultado almacenado (buffered)

Descripción

bool **sqlite_seek** (resource manejador_resultado, int numero_fila)

Método que sigue el estilo orientado a objetos:

class **SQLiteResult** {

bool **seek** (int rownum)

}

sqlite_seek() se desplaza hasta el número de fila indicado en el parámetro *numero_fila* dentro del resultado indicado en el parámetro *manejador_resultado*.

Lista de parámetros

manejador_resultado

El identificador del resultado de SQLite. Este parámetro no es obligatorio cuando se emplea el método orientado a objetos.

Nota: Esta funcion no puede usarse con resultados que no se encuentren almacenados en un almacenador intermedio (buffer).

numero_fila

El número de fila al que se va a desplazar. El número de fila empieza a contarse en el número 0 (cero) para la primera fila.

Nota: Esta funcion no puede usarse con resultados que no se encuentren almacenados en un almacenador intermedio (buffer).

Valores retornados

Devuelve **FALSE** si la fila no existe o **TRUE** en cualquier otro caso.

Ver también

sqlite_next(), sqlite_current(), sqlite_rewind()

sqlite_single_query

Descripción

Ejecuta una consulta y devuelve o una matriz para una columna o el valor de la primera fila

array **sqlite_single_query** (resource manejador_bd, string consulta [, bool solo_primera_fila [, bool decodificar_binario]])

Object oriented style (method):

class **SQLiteDatabase** {

array **singleQuery** (string consulta [, bool solo_primera_fila [, bool decodificar_binario]])

}

sqlite_udf_decode_binary

Descripción

Decodifica los datos binarios que se pasan como parámetro a las funciones FDU (función definida por el usuario)

string **sqlite_udf_decode_binary** (string datos)

sqlite_udf_decode_binary() decodifica la codificación binaria aplicada por las funciones **sqlite_udf_encode_binary()** o **sqlite_escape_string()**.

Se debe utilizar esta función para los parámetros que se pasan a las funciones FDU (función definida por el usuario) si éstos pueden contener datos binarios.

PHP no realiza este proceso de codificación/decodificación de forma automática, ya que esto podría penalizar seriamente el rendimiento de las aplicaciones.

Ver también

Sqlite_udf_encode_binary(), sqlite_create_function(), sqlite_create_aggregate()

sqlite_udf_encode_binary

Descripción

Codifica los datos binarios antes de devolverlos de una FDU (función definida por el usuario)

string **sqlite_udf_encode_binary** (string datos)

sqlite_udf_encode_binary() aplica una codificación binaria a los *datos*, de forma que sea seguro devolverlos en los resultados de las consultas (la API de la librería libsqlite no es segura con los datos binarios).

Si existe alguna posibilidad de que los datos no sean seguros de forma binaria (por ejemplo porque contienen bytes de tipo NUL o tienen un byte de tipo *0x01* como primer carácter) se debe emplear esta función para codificar los valores devueltos por la FDU.

PHP no realiza este proceso de codificación/decodificación de forma automática, ya que esto podría penalizar seriamente el rendimiento de las aplicaciones.

Nota: No se debe emplear la función **sqlite_escape_string()** para formatear las cadenas devueltas por las funciones FDU, ya que provocaría un escape doble de los datos. Se debe emplear la función **sqlite_udf_encode_binary()** en su lugar.

Ver también

sqlite_udf_decode_binary(), sqlite_escape_String(), sqlite_create_function(), sqlite_create_aggregate()

sqlite_unbuffered_query

Descripción

Ejecuta una consulta sobre la base de datos cuyo resultado no almacena todos los datos devueltos

resource **sqlite_unbuffered_query** (resource manejador_bd, string consulta [, int tipo_resultado])

resource **sqlite_unbuffered_query** (string consulta, resource manejador_bd [, int tipo_resultado])

Método que sigue el estilo orientado a objetos:

class **SQLiteDatabase** {

SQLiteUnbuffered **unbufferedQuery** (string consulta [, int tipo_resultado])

}

sqlite_unbuffered_query() es idéntica a **sqlite_query()** savlo que el resultado devuelto es secuencial y solamente se puede avanzar en el hacia delante. Por tanto, este tipo de resultados solamente puede emplearse para leer todas y cada una de las filas de forma secuencial una detrás de otra.

Esta función es útil por ejemplo para generar tablas HTML donde solo se necesita procesar una fila cada vez y no es necesario acceder de forma directa a cualquier fila del resultado en cualquier momento.

Nota: Las funciones **sqlite_seek()**, **sqlite_rewind()**, **sqlite_next()**, **sqlite_current()** y **sqlite_num_rows()** no funcionan con manejadores de resultados devueltos por **sqlite_unbuffered_query()**.

Lista de parámetros

consulta

La consulta que se quiere ejecutar.

manejador_bd

El recurso que identifica la base de datos SQLite (y que es el que devuelve la función **sqlite_open()**). Este parámetro no se requiere cuando se emplea el método orientado a objetos.

tipo_resultado

Este parametro es opcional y controla la forma en la que se indexa el array devuelto. Las constantes que se pueden utilizar son:

- **SQLITE_ASSOC** el índice del array será el nombre de los campos.
- **SQLITE_NUM** el indice del array devuelto sera numerico.
- **SQLITE_BOTH** el indice del array devuelto sera asociativo y numerico.

Valores retornados

La función devuelve el manejador del resultado o **FALSE** si se produce un error.

sqlite_unbuffered_query() devuelve un resultado secuencial que solamente puede recorrerse hacia delante, de forma que solo se puede emplear para leer todas las filas del resultado una detrás de otra.

Ver también

sqlite_query()

sqlite_valid

Descripción

Indica si hay más filas disponibles en el resultado

bool **sqlite_valid** (resource manejador_resultado)

Método que sigue el estilo orientado a objetos:

class **SQLiteResult** {

bool **valido** (void)

}class **SQLiteUnbuffered** {

bool **valido** (void)

}

Indica si hay más filas disponibles en el resultado identificado por el parámetro *manejador_resultado*.

Lista de parámetros

manejador_resultado

El identificador del resultado de SQLite. Este parámetro no es obligatorio cuando se emplea el método orientado a objetos.

Nota: Esta funcion no puede usarse con resultados que no se encuentren almacenados en un almacenador intermedio (buffer).

Valores retornados

Devuelve **TRUE** si hay más filas disponibles en el resultado identificado por *manejador_resultado* o **FALSE** en cualquier otro caso.

Ver también

sqlite_num_rows(), sqlite_changes()