

mysql egg

MySQL bindings for Chicken.
Extension for Chicken Scheme
Version 1.0

Toby Butzon

Table of Contents

1	About this egg	1
1.1	Version history	1
1.2	Requirements	1
1.3	Usage	1
2	Documentation	2
3	Examples	4
4	Data Type Conversion	5
5	Bugs	6
6	License	7
	Index	8

1 About this egg

1.1 Version history

1.0 Initial release

1.2 Requirements

This egg requires the following extensions:

MySQL client library (`-lmysqlclient`)

1.3 Usage

Load this egg like so:

`(require-extension mysql)`

2 Documentation

The MySQL egg provides (most of) the functions offered by the MySQL C API (the `foreign-mysql-*` functions). It also maps those to a set of slightly more convenient Scheme functions (the `mysql-*` functions). Finally, a few extra functions are provided for easy of use.

Only the most often used MySQL functions are described in this document. If you really want to see the full listing, consult the mole documentation.

Please send bug reports and suggestions to toby@butzon.com.

mysql-connect [procedure]
(mysql-connect [KEYWORDS])

Connect to a MySQL server.

Returns a MySQL connection object suitable for passing to the other MySQL functions. This object is referred to as `DB` when passed by all the other MySQL functions. Returns `#f` when the connection fails.

Any number of the following `KEYWORDS` may be included:

- `host`
- `user`
- `passwd`
- `db`
- `port`
- `unix-socket`
- `client-flag`

Note that default values are available for all of these arguments. (Consult the [MySQL C API](#) for details on how these defaults are determined.)

mysql-query [procedure]
(mysql-query DB SQL-STRING)

Executes `SQL-STRING` on the MySQL server and stores the result in memory. Generates an error (calls `error`) if the query fails.

mysql-fetch-row [procedure]
(mysql-fetch-row DB)

Fetches a row from the result set returned by the most recent call to `mysql-query`. If the last query failed, or if there are no more rows left in the result set, returns `#f`; otherwise returns a row object.

A row object is defined as a function that takes a single argument. If the argument is a number, the function returns the value of the field for which that number is the index, or `#f` if the index is out of range. Otherwise, the argument must be a symbol or string, in which case the function returns the value of the field for which that string (or symbol converted into a string) is the field/column name. If no such field exists, returns `#f`.

`mysql-rewind` [procedure]

(mysql-rewind DB)

Rewinds the result set; that is, resets the pointer used by `mysql-fetch-row` so that the next call to it will return the first row of the result set. If there is no current result set, does nothing.

`mysql-close` [procedure]

(mysql-close DB)

Closes the connection to DB. This frees any remaining MySQL resources from memory, but invalidates the MySQL connection object (DB) so that it may no longer be used.

`mysql-foreach-row` [procedure]

(mysql-foreach-row DB BODY)

Iterates over the entire result set (regardless of any rows that may have already been returned by `mysql-fetch-row`), calling `BODY` on each row.

`BODY` must take two arguments: the row (as described for `mysql-fetch-row` and the index of that row in the result set, starting with 1 and ending with `(mysql-num-rows DB)`).

`mysql-num-rows` [procedure]

(mysql-num-rows DB)

Returns the number of rows in the current result set. If no result set exists, returns `#f`.

`mysql-query-foreach` [procedure]

(mysql-query-foreach DB QUERY BODY)

Combines `mysql-query` and `mysql-foreach-row`.

3 Examples

A bulky usage might look like:

```
(use mysql)
```

```
(let [(db (mysql-connect host: "mysql.example.com" user: "example"
                        passwd: "secret"))]
  (if (not db) (error (conc "MySQL connection failed: " (mysql-error db))))
  (mysql-query db "SHOW DATABASES")
  (do [(row (mysql-fetch-row db) (mysql-fetch-row db))]
      [(not row)]
      (display (conc "Row " idx ": " (row "Database") "\n")))
  (mysql-close db))
```

A slightly more compact version that does the same thing:

```
(use mysql)
```

```
(let [(db (mysql-connect host: "mysql.example.com" user: "example"
                        passwd: "secret"))]
  (if (not db) (error (conc "MySQL connection failed: " (mysql-error db))))
  (mysql-query-foreach db "SHOW DATABASES" (lambda (row idx)
      (display (conc "Row " idx ": " (row "Database") "\n"))))
  (mysql-close db))
```

4 Data Type Conversion

All MySQL result data (except NULL) are returned as Scheme strings. The NULL value is represented by `#f`. Booleans are expressed as Scheme strings `"1"` and `"0"`. All remaining types, including numeric types and strings are returned as Scheme strings.

5 Bugs

This is alpha quality software. Only very basic functionality has been tested so far. I look forward to providing a more complete test suite (and probably a slew of bugfixes) with the next release.

Retrieval of field info isn't supported.

`mysql-escape-string` is broken when it's used for binary data.

Not yet sure how to handle `unsigned long *` for `foreign-mysql-fetch-lengths`.

`foreign-mysql-get-charset-info` isn't yet supported.

I need to nail down the supported `libmysqlclient` versions. Right now there are some functions I've put off because they may or may not be supported in my target range. We'll see, soon...

6 License

Copyright (c) 2005 Toby Butzon.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the Software), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED ASIS, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Index

mysql-close	3	mysql-num-rows	3
mysql-connect	2	mysql-query	2
mysql-fetch-row	2	mysql-query-foreach	3
mysql-foreach-row	3	mysql-rewind	3