Class	MySQLc
In:	mysql-cellophane.rb
Parent:	Object

MySQL-cellophane is a very thin wrapper around the Mysql Ruby implementation MySQc provides an API interface for those famaliar with SQL and wishes to have more fine grained control over their queries, but don't want to write each query manually.

Author: Chris Marshall (chris@chrismar035.com)

Methods

build_query connect count delete esc execute extract_fields_from extract_values_from insert inserted_id new quoted_string_from select set_attributes update

Attributes

extra_cond	[RW]
field_list	[RW]
id	[RW]
id_field	[RW]
query	[RW]
result	[RW]
table	[RW]

Public Class methods

new(options = {})

initializes common instance variables.

can setup connection, but does not create connection

Public Instance methods

build_query(update = false)

Creates the internal query string for processing. Shouldn't ever NEED to be used externally.

connect(options = {})

takes hash of connection ooptions, has default also sets @db as Mysql instance as separate method for lazy connection

count(*vals)

returns an integer of the totals returned by query

takes an optional last parameter of a hash. This hash has symbol keys of any/all of the following: table, field_list, id_field, id, extra_cond

delete(*vals)

deletes row(s) matching the instalce variables' criteria

takes an optional last parameter of a hash. This hash has symbol keys of any/all of the following: table, field_list, id_field, id, extra_cond

esc(raw_string)

takes string parameter and returns saftely escaped string

execute()

builds query from parts if neccessary

returns cached result if exists

saves result in @result

extract_fields_from(hash)

takes a hash and returns a comma separated list of keys

```
  { :key1 => 'val1', :key2 => 'key2' } #=> "key1, key2"
```

extract_values_from(hash)

takes a hash and returns a songle quoted string of the values

```
  { :key => 'val1', :key2 => "val'2" }  #=> "'val1','val\'2'"
```

insert(inserts, *vals)

inserts a new row into the table from the given hash

```
 { :column_one_field => 'column_one_value',
 :column_two_field => 'column_two_value' }
```

return the newly insertted row's ID

takes an optional last parameter of a hash. This hash has symbol keys of any/all of the following: table, field_list, id_field, id, extra_cond

inserted_id()

returns the last inserted ID

quoted_string_from(hash_or_array)

library function to convert values in a hash (or plain array) into an escaped single quoted comma delimited string

i.e.

```
  { :key => 'val1', :key2 => "val'2" }  #=> "'val1','val\'2'"
  [ 'val1', "val'2" ]                   #=> "'val1', 'val\'2'"
```

select(*vals)

creates a select query and returns the reqult as Mysql::Result

takes an optional last parameter of a hash. This hash has symbol keys of any/all of the following: table, field_list, id_field, id, extra_cond

set_attributes(*vals)

sets instance attributes from fucntion parameters called by each public API methods

update(updates, *vals)

takes a given hash:

```
 { :column_one_field => 'column_one_value', :column_two_field => 'column_two_value' }
```

and updates the correct row

takes an optional last parameter of a hash. This hash has symbol keys of any/all of the following: table, field_list, id_field, id, extra_cond

[Validate]