

general\_ledger

Generated by Doxygen 1.8.1.2

Sat Jun 14 2014 02:38:15



# Contents

<b>1</b>	<b>General Ledger.</b>	<b>1</b>
<b>2</b>	<b>Module Index</b>	<b>3</b>
2.1	Modules . . . . .	3
<b>3</b>	<b>Class Index</b>	<b>5</b>
3.1	Class Hierarchy . . . . .	5
<b>4</b>	<b>Class Index</b>	<b>7</b>
4.1	Class List . . . . .	7
<b>5</b>	<b>File Index</b>	<b>9</b>
5.1	File List . . . . .	9
<b>6</b>	<b>Module Documentation</b>	<b>11</b>
6.1	Database interaction module . . . . .	11
6.1.1	Detailed Description . . . . .	12
6.1.2	Function Documentation . . . . .	12
6.1.2.1	get_connection . . . . .	12
6.1.2.2	get_database_type . . . . .	12
6.2	SQL statements module . . . . .	13
6.2.1	Detailed Description . . . . .	13
6.3	Program configuration module . . . . .	14
6.3.1	Detailed Description . . . . .	14
6.4	General purpose helpers. . . . .	15
6.4.1	Detailed Description . . . . .	15
6.4.2	Function Documentation . . . . .	15
6.4.2.1	split . . . . .	15
6.4.2.2	split . . . . .	15
6.4.2.3	trim . . . . .	15
6.4.2.4	trim_back . . . . .	16
6.4.2.5	trim_front . . . . .	16
6.5	Reporting program. . . . .	17

6.5.1	Detailed Description	17
6.5.2	Function Documentation	17
6.5.2.1	login	17
6.5.2.2	main	17
6.5.2.3	set_configuration	18
6.6	Database program.	19
6.6.1	Detailed Description	19
6.6.2	Function Documentation	19
6.6.2.1	login	19
6.6.2.2	main	19
6.6.2.3	set_configuration	20
<b>7</b>	<b>Class Documentation</b>	<b>21</b>
7.1	genleg::Config Class Reference	21
7.1.1	Detailed Description	21
7.1.2	Constructor & Destructor Documentation	21
7.1.2.1	Config	21
7.1.2.2	~Config	22
7.1.3	Member Function Documentation	22
7.1.3.1	add_cmdline_option	22
7.1.3.2	is_set	22
7.1.3.3	operator[]	22
7.1.3.4	populate_from_cmdline	22
7.1.3.5	populate_from_file	23
7.1.4	Member Data Documentation	23
7.1.4.1	m_opts_set	23
7.1.4.2	m_opts_supp	23
7.2	genleg::ConfigBadConfigFile Class Reference	23
7.2.1	Detailed Description	24
7.3	genleg::ConfigBadOption Class Reference	24
7.3.1	Detailed Description	25
7.4	genleg::ConfigCouldNotOpenFile Class Reference	25
7.4.1	Detailed Description	26
7.5	genleg::ConfigException Class Reference	26
7.5.1	Detailed Description	27
7.6	genleg::ConfigOptionNotSet Class Reference	27
7.6.1	Detailed Description	28
7.7	gldb::DBConn Class Reference	28
7.7.1	Detailed Description	29
7.7.2	Constructor & Destructor Documentation	29

7.7.2.1	DBConn	29
7.7.2.2	DBConn	29
7.7.3	Member Function Documentation	29
7.7.3.1	operator=	29
7.7.3.2	query	29
7.7.3.3	select	29
7.7.4	Member Data Documentation	30
7.7.4.1	m_imp	30
7.8	gldb::DBConnCouldNotConnect Class Reference	30
7.8.1	Detailed Description	31
7.8.2	Constructor & Destructor Documentation	31
7.8.2.1	DBConnCouldNotConnect	31
7.9	gldb::DBConnCouldNotQuery Class Reference	31
7.9.1	Detailed Description	32
7.9.2	Constructor & Destructor Documentation	32
7.9.2.1	DBConnCouldNotQuery	32
7.10	gldb::DBConnDummy Class Reference	32
7.10.1	Detailed Description	33
7.10.2	Constructor & Destructor Documentation	33
7.10.2.1	DBConnDummy	33
7.10.2.2	DBConnDummy	34
7.10.2.3	~DBConnDummy	34
7.10.3	Member Function Documentation	34
7.10.3.1	operator=	34
7.10.3.2	select	34
7.11	gldb::DBConnException Class Reference	34
7.11.1	Detailed Description	35
7.11.2	Constructor & Destructor Documentation	35
7.11.2.1	DBConnException	35
7.12	gldb::DBConnImp Class Reference	35
7.12.1	Detailed Description	36
7.12.2	Constructor & Destructor Documentation	36
7.12.2.1	DBConnImp	36
7.12.2.2	~DBConnImp	36
7.12.3	Member Function Documentation	36
7.12.3.1	query	36
7.12.3.2	select	37
7.13	gldb::DBConnMySQL Class Reference	37
7.13.1	Detailed Description	38
7.13.2	Constructor & Destructor Documentation	38

7.13.2.1	DBConnMySQL	38
7.13.2.2	DBConnMySQL	38
7.13.2.3	~DBConnMySQL	38
7.13.3	Member Function Documentation	38
7.13.3.1	operator=	39
7.13.3.2	query	39
7.13.3.3	select	39
7.13.4	Member Data Documentation	39
7.13.4.1	m_conn	39
7.14	genleg::DBSQLMySQL Class Reference	39
7.14.1	Detailed Description	40
7.15	genleg::DBSQLStatements Class Reference	40
7.15.1	Detailed Description	41
7.15.2	Constructor & Destructor Documentation	41
7.15.2.1	DBSQLStatements	41
7.15.2.2	~DBSQLStatements	41
7.15.3	Member Function Documentation	41
7.15.3.1	create_table	41
7.15.3.2	create_view	42
7.15.3.3	drop_table	42
7.15.3.4	drop_view	42
7.16	gldb::Table Class Reference	42
7.16.1	Detailed Description	43
7.16.2	Constructor & Destructor Documentation	44
7.16.2.1	Table	44
7.16.2.2	~Table	44
7.16.3	Member Function Documentation	44
7.16.3.1	append_record	44
7.16.3.2	create_from_file	44
7.16.3.3	get_headers	44
7.16.3.4	insert_query	45
7.16.3.5	num_fields	45
7.16.3.6	num_records	45
7.16.3.7	operator[]	45
7.16.3.8	set_quoted	45
7.16.4	Member Data Documentation	45
7.16.4.1	m_headers	46
7.16.4.2	m_quoted	46
7.16.4.3	m_records	46
7.17	gldb::TableBadInputFile Class Reference	46

7.17.1 Detailed Description . . . . .	47
7.18 glDb::TableCouldNotOpenInputFile Class Reference . . . . .	47
7.18.1 Detailed Description . . . . .	47
7.19 glDb::TableException Class Reference . . . . .	48
7.19.1 Detailed Description . . . . .	48
7.20 glDb::TableField Class Reference . . . . .	48
7.20.1 Detailed Description . . . . .	49
7.20.2 Constructor & Destructor Documentation . . . . .	49
7.20.2.1 TableField . . . . .	49
7.20.2.2 TableField . . . . .	49
7.20.2.3 ~TableField . . . . .	50
7.20.3 Member Function Documentation . . . . .	50
7.20.3.1 length . . . . .	50
7.20.3.2 operator std::string . . . . .	50
7.20.3.3 operator+= . . . . .	50
7.20.3.4 operator+= . . . . .	50
7.20.3.5 operator= . . . . .	50
7.20.3.6 operator= . . . . .	51
7.20.3.7 operator[] . . . . .	51
7.20.3.8 operator[] . . . . .	51
7.20.4 Friends And Related Function Documentation . . . . .	51
7.20.4.1 operator<< . . . . .	51
7.20.5 Member Data Documentation . . . . .	52
7.20.5.1 m_data . . . . .	52
7.21 glDb::TableRow Class Reference . . . . .	52
7.21.1 Detailed Description . . . . .	53
7.21.2 Constructor & Destructor Documentation . . . . .	53
7.21.2.1 TableRow . . . . .	53
7.21.2.2 TableRow . . . . .	53
7.21.2.3 TableRow . . . . .	53
7.21.2.4 ~TableRow . . . . .	53
7.21.3 Member Function Documentation . . . . .	53
7.21.3.1 append_field . . . . .	53
7.21.3.2 append_field . . . . .	53
7.21.3.3 append_field . . . . .	53
7.21.3.4 operator[] . . . . .	54
7.21.3.5 operator[] . . . . .	54
7.21.3.6 print . . . . .	54
7.21.3.7 record_string . . . . .	54
7.21.3.8 record_string . . . . .	54

7.21.3.9	size	55
7.21.4	Member Data Documentation	55
7.21.4.1	m_fields	55
<b>8</b>	<b>File Documentation</b>	<b>57</b>
8.1	lib/config/config.cpp File Reference	57
8.1.1	Detailed Description	57
8.2	lib/config/config.h File Reference	58
8.2.1	Detailed Description	59
8.3	lib/config/config_getopt.cpp File Reference	59
8.3.1	Detailed Description	59
8.3.2	Macro Definition Documentation	60
8.3.2.1	_XOPEN_SOURCE	60
8.4	lib/database/data_structures.h File Reference	60
8.4.1	Detailed Description	61
8.5	lib/database/database.h File Reference	61
8.5.1	Detailed Description	62
8.6	lib/database/dbconn.cpp File Reference	63
8.6.1	Detailed Description	64
8.7	lib/database/dbconn.h File Reference	64
8.7.1	Detailed Description	66
8.8	lib/database/dbconnimp.h File Reference	66
8.8.1	Detailed Description	68
8.9	lib/database/table.cpp File Reference	68
8.9.1	Detailed Description	68
8.10	lib/database/table.h File Reference	69
8.10.1	Detailed Description	70
8.11	lib/database/tablefield.cpp File Reference	70
8.11.1	Detailed Description	71
8.12	lib/database/tablefield.h File Reference	71
8.12.1	Detailed Description	73
8.13	lib/database/ablerow.cpp File Reference	73
8.13.1	Detailed Description	73
8.14	lib/database/ablerow.h File Reference	74
8.14.1	Detailed Description	75
8.15	lib/database_imp/database_imp.h File Reference	75
8.15.1	Detailed Description	77
8.16	lib/database_imp/dummy/dbconn_dummy_imp.cpp File Reference	77
8.16.1	Detailed Description	78
8.17	lib/database_imp/dummy/dbconn_dummy_imp.h File Reference	79



8.17.1 Detailed Description . . . . .	80
8.18 lib/database_imp/mysql/dbconn_mysql_imp.cpp File Reference . . . . .	81
8.18.1 Detailed Description . . . . .	81
8.19 lib/database_imp/mysql/dbconn_mysql_imp.h File Reference . . . . .	82
8.19.1 Detailed Description . . . . .	83
8.20 lib/dbsql/dbsql_mysql.h File Reference . . . . .	83
8.20.1 Detailed Description . . . . .	84
8.21 lib/dbsql/dbsqlstatements.cpp File Reference . . . . .	84
8.21.1 Detailed Description . . . . .	85
8.22 lib/dbsql/dbsqlstatements.h File Reference . . . . .	85
8.22.1 Detailed Description . . . . .	86
8.23 lib/stringhelp/stringhelp.cpp File Reference . . . . .	87
8.23.1 Detailed Description . . . . .	87
8.24 lib/stringhelp/stringhelp.h File Reference . . . . .	87
8.24.1 Detailed Description . . . . .	89
8.25 progs/gl_db/gl_db_main.cpp File Reference . . . . .	89
8.25.1 Detailed Description . . . . .	90
8.26 progs/gl_report/gl_report_main.cpp File Reference . . . . .	90
8.26.1 Detailed Description . . . . .	92



## Chapter 1

# General Ledger.

General Ledger will be a fully-featured, multi-user, open-source general ledger system. The project is in the early stages of development.



## Chapter 2

# Module Index

### 2.1 Modules

Here is a list of all modules:

Database interaction module . . . . .	11
SQL statements module . . . . .	13
Program configuration module . . . . .	14
General purpose helpers. . . . .	15
Reporting program. . . . .	17
Database program. . . . .	19



## Chapter 3

# Class Index

### 3.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

genleg::Config . . . . .	21
genleg::ConfigException . . . . .	26
genleg::ConfigBadConfigFile . . . . .	23
genleg::ConfigBadOption . . . . .	24
genleg::ConfigCouldNotOpenFile . . . . .	25
genleg::ConfigOptionNotSet . . . . .	27
gldb::DBConn . . . . .	28
gldb::DBConnException . . . . .	34
gldb::DBConnCouldNotConnect . . . . .	30
gldb::DBConnCouldNotQuery . . . . .	31
gldb::DBConnImp . . . . .	35
gldb::DBConnDummy . . . . .	32
gldb::DBConnMySQL . . . . .	37
genleg::DBSQLStatements . . . . .	40
genleg::DBSQLMySQL . . . . .	39
gldb::Table . . . . .	42
gldb::TableException . . . . .	48
gldb::TableBadInputFile . . . . .	46
gldb::TableCouldNotOpenInputFile . . . . .	47
gldb::TableField . . . . .	48
gldb::TableRow . . . . .	52





## Chapter 4

# Class Index

### 4.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

<a href="#">genleg::Config</a>	Configuration options class . . . . .	21
<a href="#">genleg::ConfigBadConfigFile</a>	Exception class for badly formed configuration file . . . . .	23
<a href="#">genleg::ConfigBadOption</a>	Exception class for bad provided option . . . . .	24
<a href="#">genleg::ConfigCouldNotOpenFile</a>	Exception class for when conf file cannot be opened . . . . .	25
<a href="#">genleg::ConfigException</a>	Configuration module exception base class . . . . .	26
<a href="#">genleg::ConfigOptionNotSet</a>	Exception class for option not set . . . . .	27
<a href="#">gldb::DBConn</a>	Database connection class . . . . .	28
<a href="#">gldb::DBConnCouldNotConnect</a>	Could not connect to database exception class . . . . .	30
<a href="#">gldb::DBConnCouldNotQuery</a>	Could not execute database query exception class . . . . .	31
<a href="#">gldb::DBConnDummy</a>	Dummy database implementation class . . . . .	32
<a href="#">gldb::DBConnException</a>	Base database connection exception class . . . . .	34
<a href="#">gldb::DBConnImp</a>	Abstract database implementation base class . . . . .	35
<a href="#">gldb::DBConnMySQL</a>	MySQL database implementation class . . . . .	37
<a href="#">genleg::DBSQLMySQL</a>	MySQL SQL statements class . . . . .	39
<a href="#">genleg::DBSQLStatements</a>	SQL statements class . . . . .	40
<a href="#">gldb::Table</a>	Database table class . . . . .	42
<a href="#">gldb::TableBadInputFile</a>	Could not connect to database exception class . . . . .	46
<a href="#">gldb::TableCouldNotOpenInputFile</a>	Could not connect to database exception class . . . . .	47
<a href="#">gldb::TableException</a>	Base database connection exception class . . . . .	48

<a href="#">gldb::TableField</a>	
Database table field class . . . . .	<a href="#">48</a>
<a href="#">gldb::TableRow</a>	
Database table row class . . . . .	<a href="#">52</a>

## Chapter 5

# File Index

### 5.1 File List

Here is a list of all documented files with brief descriptions:

lib/config/ <a href="#">config.cpp</a>	
Implementation of program configurations class . . . . .	57
lib/config/ <a href="#">config.h</a>	
Interface to program configurations class . . . . .	58
lib/config/ <a href="#">config_getopt.cpp</a>	
Implementation of command line functionality . . . . .	59
lib/database/ <a href="#">data_structures.h</a>	
Main interface to database data structures . . . . .	60
lib/database/ <a href="#">database.h</a>	
User interface to database functionality . . . . .	61
lib/database/ <a href="#">dbconn.cpp</a>	
Implementation of database connection class . . . . .	63
lib/database/ <a href="#">dbconn.h</a>	
Interface to database connection base class . . . . .	64
lib/database/ <a href="#">dbconnimp.h</a>	
Interface to abstract database implementation base class . . . . .	66
lib/database/ <a href="#">table.cpp</a>	
Implementation of database table data structure . . . . .	68
lib/database/ <a href="#">table.h</a>	
Interface to database table data structure . . . . .	69
lib/database/ <a href="#">tablefield.cpp</a>	
Implementation of database table field class . . . . .	70
lib/database/ <a href="#">tablefield.h</a>	
Interface to database table field class . . . . .	71
lib/database/ <a href="#">tablerow.cpp</a>	
Implementation of database table row data structure . . . . .	73
lib/database/ <a href="#">tablerow.h</a>	
Interface to database table row data structure . . . . .	74
lib/database_imp/ <a href="#">database_imp.h</a>	
Interface to database implementation factory function . . . . .	75
lib/database_imp/dummy/ <a href="#">dbconn_dummy_imp.cpp</a>	
Implementation of Dummy database connection implementation class . . . . .	77
lib/database_imp/dummy/ <a href="#">dbconn_dummy_imp.h</a>	
Interface to dummy database connection implementation class . . . . .	79
lib/database_imp/mysql/ <a href="#">dbconn_mysql_imp.cpp</a>	
Implementation of MySQL database connection implementation class . . . . .	81
lib/database_imp/mysql/ <a href="#">dbconn_mysql_imp.h</a>	
Interface to MySQL database connection implementation class . . . . .	82

lib/dbsql/ <b>dbsql.h</b> . . . . .	??
lib/dbsql/ <b>dbsql_functions.h</b> . . . . .	??
lib/dbsql/ <b>dbsql_implementations.h</b> . . . . .	??
lib/dbsql/ <a href="#">dbsql_mysql.h</a>	
Interface to MySQL SQL statement class . . . . .	83
lib/dbsql/ <a href="#">dbsqlstatements.cpp</a>	
Implementation of SQL statement class . . . . .	84
lib/dbsql/ <a href="#">dbsqlstatements.h</a>	
Interface to SQL statement class . . . . .	85
lib/stringhelp/ <a href="#">stringhelp.cpp</a>	
Implementation of string helper functions . . . . .	87
lib/stringhelp/ <a href="#">stringhelp.h</a>	
Interface to string helper functions . . . . .	87
progs/gl_db/ <a href="#">gl_db_main.cpp</a>	
Main functionality for gl_db program . . . . .	89
progs/gl_report/ <a href="#">gl_report_main.cpp</a>	
Main functionality for gl_report program . . . . .	90

## Chapter 6

# Module Documentation

### 6.1 Database interaction module

#### Classes

- class [gldb::DBConnException](#)  
*Base database connection exception class.*
- class [gldb::DBConnCouldNotConnect](#)  
*Could not connect to database exception class.*
- class [gldb::DBConnCouldNotQuery](#)  
*Could not execute database query exception class.*
- class [gldb::DBConn](#)  
*Database connection class.*
- class [gldb::DBConnImp](#)  
*Abstract database implementation base class.*
- class [gldb::TableException](#)  
*Base database connection exception class.*
- class [gldb::TableBadInputFile](#)  
*Could not connect to database exception class.*
- class [gldb::TableCouldNotOpenInputFile](#)  
*Could not connect to database exception class.*
- class [gldb::Table](#)  
*Database table class.*
- class [gldb::TableField](#)  
*Database table field class.*
- class [gldb::TableRow](#)  
*Database table row class.*
- class [gldb::DBConnDummy](#)  
*Dummy database implementation class.*
- class [gldb::DBConnMySQL](#)  
*MySQL database implementation class.*

#### Functions

- [DBConnImp \\* gldb::get\\_connection](#) (const std::string database, const std::string hostname, const std::string username, const std::string password)  
*Creates and returns a pointer to a database implementation.*
- std::string [gldb::get\\_database\\_type](#) ()  
*Returns the name of the compiled-in database type.*

### 6.1.1 Detailed Description

Module for interacting with the database.

### 6.1.2 Function Documentation

#### 6.1.2.1 DBConnImp \* gldb::get\_connection ( const std::string *database*, const std::string *hostname*, const std::string *username*, const std::string *password* )

Creates and returns a pointer to a database implementation.

The implementation of this function is provided by the individual database implementations. One database implementation is compiled into the program at any one time. Multiple database systems are, or will be, supported, and not every system will possess the libraries and headers to compile every implementation. Therefore, only one implementation is compiled in at a time. The fact that each database implementation will implement this function to return the correct derived class prevents any attempt to compile unsupported library code. This would not be feasible if we were to simply provide each implementation as a subclass.

#### Parameters

<i>database</i>	The name of the database to which to connect.
<i>hostname</i>	The hostname of the computer running the database.
<i>username</i>	The username with which to log into the database.
<i>password</i>	The password with which to log into the database.

#### Returns

A pointer to the database implementation.

#### 6.1.2.2 std::string gldb::get\_database\_type ( )

Returns the name of the compiled-in database type.

#### Returns

The name of the compiled-in database type.

## 6.2 SQL statements module

### Classes

- class [genleg::DBSQLMySQL](#)  
*MySQL SQL statements class.*
- class [genleg::DBSQLStatements](#)  
*SQL statements class.*

### 6.2.1 Detailed Description

Module for producing SQL statements used by program.

## 6.3 Program configuration module

### Classes

- class [genleg::ConfigException](#)  
*Configuration module exception base class.*
- class [genleg::ConfigOptionNotSet](#)  
*Exception class for option not set.*
- class [genleg::ConfigBadOption](#)  
*Exception class for bad provided option.*
- class [genleg::ConfigCouldNotOpenFile](#)  
*Exception class for when conf file cannot be opened.*
- class [genleg::ConfigBadConfigFile](#)  
*Exception class for badly formed configuration file.*
- class [genleg::Config](#)  
*Configuration options class.*

### Enumerations

- enum [genleg::Argument](#)  
*Enumeration class for option argument specifications.*

#### 6.3.1 Detailed Description

Module for getting options from the command line and configuration files.



## 6.4 General purpose helpers.

### Functions

- `std::string & pgstring::trim_front (std::string &s)`  
*Trims leading whitespace from a string.*
- `std::string & pgstring::trim_back (std::string &s)`  
*Trims trailing whitespace from a string.*
- `std::string & pgstring::trim (std::string &s)`  
*Trims leading and trailing whitespace from a string.*
- `std::vector< std::string > pgstring::split (const std::string &s, const char delim)`  
*Splits a delimited string into tokens.*
- `std::vector< std::string > & pgstring::split (std::vector< std::string > &vec, const std::string &s, const char delim)`  
*Splits a delimited string into tokens.*

#### 6.4.1 Detailed Description

General purpose helper classes and functions.

#### 6.4.2 Function Documentation

##### 6.4.2.1 `std::vector< std::string > pgstring::split ( const std::string & s, const char delim )`

Splits a delimited string into tokens.

##### Parameters

<code>s</code>	The string to split.
<code>delim</code>	The delimiter character on which to split.

##### Returns

A vector of tokens.

##### 6.4.2.2 `std::vector< std::string > & pgstring::split ( std::vector< std::string > & vec, const std::string & s, const char delim )`

Splits a delimited string into tokens.

##### Parameters

<code>vec</code>	The vector into which to add the tokens.
<code>s</code>	The string to split.
<code>delim</code>	The delimiter character on which to split.

##### Returns

A reference to `vec`.

##### 6.4.2.3 `std::string & pgstring::trim ( std::string & s )`

Trims leading and trailing whitespace from a string.

**Parameters**

<i>s</i>	The string to trim.
----------	---------------------

**Returns**

The trimmed string.

**6.4.2.4 `std::string & pgstring::trim_back ( std::string & s )`**

Trims trailing whitespace from a string.

**Parameters**

<i>s</i>	The string to trim.
----------	---------------------

**Returns**

The trimmed string.

**6.4.2.5 `std::string & pgstring::trim_front ( std::string & s )`**

Trims leading whitespace from a string.

**Parameters**

<i>s</i>	The string to trim.
----------	---------------------

**Returns**

The trimmed string.

## 6.5 Reporting program.

### Functions

- static void `set_configuration` (`genleg::Config` &config, int argc, char \*argv[])  
*Sets program configuration options.*
- static void `print_usage_message` ()  
*Prints a program usage message.*
- static void `print_version_message` ()  
*Prints a program version message.*
- static void `print_help_message` ()  
*Prints a program help message.*
- static std::string `login` (void)  
*Gets a password from the terminal.*
- int `main` (int argc, char \*argv[])  
*Main function.*

### Variables

- static const char \* `progrname` = "gl\_report"  
*Static variable for program name.*

#### 6.5.1 Detailed Description

Administrative reporting program.

#### 6.5.2 Function Documentation

##### 6.5.2.1 static std::string login ( void ) [static]

Gets a password from the terminal.

##### Returns

The password.

##### 6.5.2.2 int main ( int argc, char \* argv[] )

Main function.

##### Parameters

<code>argc</code>	Number of command line arguments.
<code>argv</code>	Command line arguments.

#### Returns

Exit status code.

6.5.2.3 `static void set_configuration ( genleg::Config & config, int argc, char * argv[] )` [static]

Sets program configuration options.

#### Parameters

<i>config</i>	Reference to a Config object.
<i>argc</i>	argc passed to <code>main()</code> .
<i>argv</i>	argv passed to <code>main()</code> .

## 6.6 Database program.

### Functions

- static void `set_configuration` (`Config` &config, int argc, char \*argv[])  
*Sets program configuration options.*
- static void `print_usage_message` ()  
*Prints a program usage message.*
- static void `print_version_message` ()  
*Prints a program version message.*
- static void `print_help_message` ()  
*Prints a program help message.*
- static std::string `login` (void)  
*Gets a password from the terminal.*
- int `main` (int argc, char \*argv[])  
*Main function.*

### Variables

- static const char \* `progrname` = "gl\_db"  
*Static variable for program name.*

#### 6.6.1 Detailed Description

Administrative database management program.

#### 6.6.2 Function Documentation

##### 6.6.2.1 static std::string login ( void ) [static]

Gets a password from the terminal.

##### Returns

The password.

##### 6.6.2.2 int main ( int argc, char \* argv[] )

Main function.

##### Parameters

<code>argc</code>	Number of command line arguments.
<code>argv</code>	Command line arguments.

### Returns

Exit status code.

6.6.2.3 `static void set_configuration ( Config & config, int argc, char * argv[] )` `[static]`

Sets program configuration options.

### Parameters

<i>config</i>	Reference to a Config object.
<i>argc</i>	<code>argc</code> passed to <code>main()</code> .
<i>argv</i>	<code>argv</code> passed to <code>main()</code> .

## Chapter 7

# Class Documentation

### 7.1 genleg::Config Class Reference

Configuration options class.

```
#include <config.h>
```

#### Public Member Functions

- [Config](#) ()
- [~Config](#) ()
- void [add\\_cmdline\\_option](#) (const std::string option, const enum [Argument](#) arg)  
*Adds a supported command line option.*
- void [populate\\_from\\_cmdline](#) (const int argc, char \*const \*argv)  
*Populates options from the command line.*
- void [populate\\_from\\_file](#) (const std::string filename)  
*Populates options from a configuration file.*
- bool [is\\_set](#) (const std::string option) const  
*Checks if an option is set.*
- const std::string & [operator\[\]](#) (const std::string &option) const  
*operator[] overload.*

#### Private Attributes

- std::map< std::string, std::string > [m\\_opts\\_set](#)
- std::list< std::pair< std::string, enum [Argument](#) > > [m\\_opts\\_supp](#)

#### 7.1.1 Detailed Description

Configuration options class.

#### 7.1.2 Constructor & Destructor Documentation

##### 7.1.2.1 Config::Config ( )

Constructor

### 7.1.2.2 Config::~Config ( )

Destructor

## 7.1.3 Member Function Documentation

### 7.1.3.1 void Config::add\_cmdline\_option ( const std::string *option*, const enum Argument *arg* )

Adds a supported command line option.

#### Parameters

<i>option</i>	The name of the option.
<i>arg</i>	The argument specification for the option.

### 7.1.3.2 bool Config::is\_set ( const std::string *option* ) const

Checks is an option is set.

#### Parameters

<i>option</i>	The name of the option to check.
---------------	----------------------------------

#### Returns

`true` if the option has been set, `false` if it has not.

### 7.1.3.3 const std::string & Config::operator[] ( const std::string & *option* ) const

operator[] overload.

Retrieves the value of a set option.

#### Parameters

<i>option</i>	The name of the option.
---------------	-------------------------

#### Returns

The value of the option.

#### Exceptions

<a href="#"><i>ConfigOptionNotSet</i></a>	If the named option has not been set.
---	---------------------------------------

### 7.1.3.4 void Config::populate\_from\_cmdline ( const int *argc*, char \*const \* *argv* )

Populates options from the command line.

#### Parameters

<i>argc</i>	<i>argc</i> supplied to <code>main()</code> .
<i>argv</i>	<i>argv</i> supplied to <code>main()</code> .



## Exceptions

<a href="#"><i>ConfigBadOption</i></a>	If an unsupported option is specified, or if a required argument is missing, or if an unexpected argument is found.
--	---

## 7.1.3.5 void Config::populate\_from\_file ( const std::string filename )

Populates options from a configuration file.

## Parameters

<i>filename</i>	The name of the configuration file.
-----------------	-------------------------------------

## Exceptions

<a href="#"><i>ConfigCouldNotOpenFile</i></a>	If the configuration file cannot be opened.
<a href="#"><i>ConfigBadConfigFile</i></a>	If the configuration file is badly formed.

## 7.1.4 Member Data Documentation

## 7.1.4.1 std::map&lt;std::string, std::string&gt; genleg::Config::m\_opts\_set [private]

Map of options which have been set

## 7.1.4.2 std::list&lt;std::pair&lt;std::string, enum Argument&gt; &gt; genleg::Config::m\_opts\_supp [private]

List of options which are supported

The documentation for this class was generated from the following files:

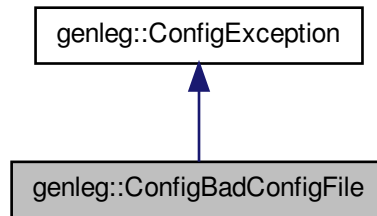
- lib/config/[config.h](#)
- lib/config/[config.cpp](#)
- lib/config/[config\\_getopt.cpp](#)

## 7.2 genleg::ConfigBadConfigFile Class Reference

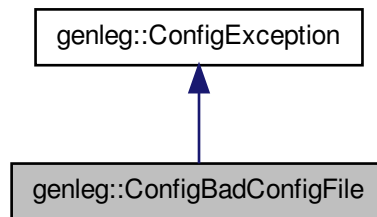
Exception class for badly formed configuration file.

```
#include <config.h>
```

Inheritance diagram for `genleg::ConfigBadConfigFile`:



Collaboration diagram for `genleg::ConfigBadConfigFile`:



### 7.2.1 Detailed Description

Exception class for badly formed configuration file.

The documentation for this class was generated from the following file:

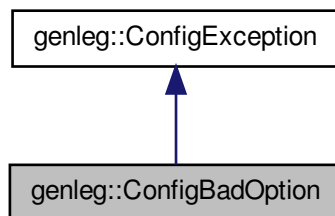
- `lib/config/config.h`

## 7.3 `genleg::ConfigBadOption` Class Reference

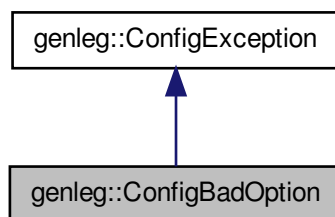
Exception class for bad provided option.

```
#include <config.h>
```

Inheritance diagram for genleg::ConfigBadOption:



Collaboration diagram for genleg::ConfigBadOption:



### 7.3.1 Detailed Description

Exception class for bad provided option.

The documentation for this class was generated from the following file:

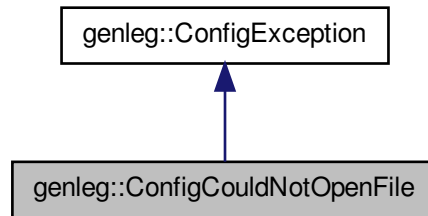
- lib/config/[config.h](#)

## 7.4 genleg::ConfigCouldNotOpenFile Class Reference

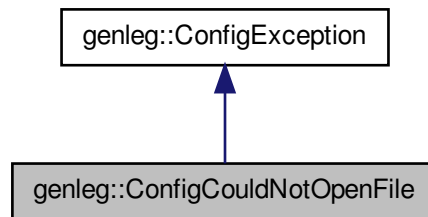
Exception class for when conf file cannot be opened.

```
#include <config.h>
```

Inheritance diagram for `genleg::ConfigCouldNotOpenFile`:



Collaboration diagram for `genleg::ConfigCouldNotOpenFile`:



### 7.4.1 Detailed Description

Exception class for when conf file cannot be opened.

The documentation for this class was generated from the following file:

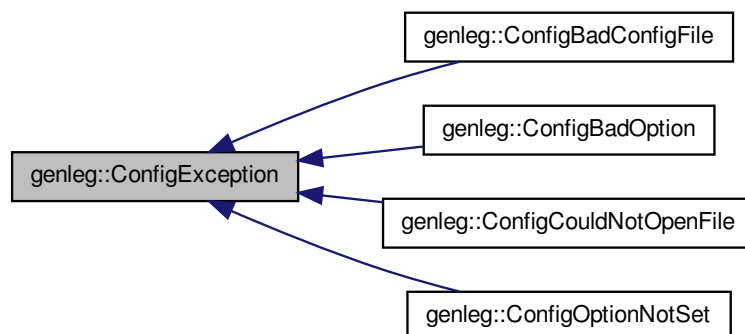
- `lib/config/config.h`

## 7.5 `genleg::ConfigException` Class Reference

Configuration module exception base class.

```
#include <config.h>
```

Inheritance diagram for genleg::ConfigException:



### 7.5.1 Detailed Description

Configuration module exception base class.

The documentation for this class was generated from the following file:

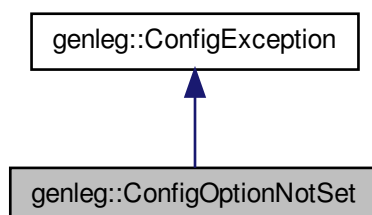
- `lib/config/config.h`

## 7.6 genleg::ConfigOptionNotSet Class Reference

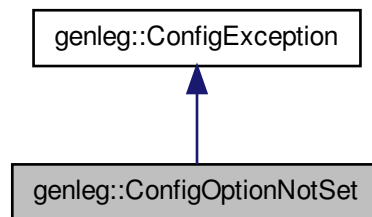
Exception class for option not set.

```
#include <config.h>
```

Inheritance diagram for genleg::ConfigOptionNotSet:



Collaboration diagram for genleg::ConfigOptionNotSet:



### 7.6.1 Detailed Description

Exception class for option not set.

The documentation for this class was generated from the following file:

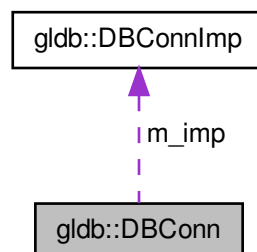
- `lib/config/config.h`

## 7.7 glldb::DBConn Class Reference

Database connection class.

```
#include <dbconn.h>
```

Collaboration diagram for glldb::DBConn:



### Public Member Functions

- `DBConn (DBConnImp *imp)`  
*Constructor.*
- `~DBConn ()`  
*Destructor..*

- void [query](#) (std::string *sql\_query*)  
*Runs an SQL query.*
- [Table select](#) (std::string *query*)  
*Runs an SQL SELECT query.*
- [DBConn](#) (const [DBConn](#) &)
- [DBConn](#) & [operator=](#) (const [DBConn](#) &)

### Private Attributes

- [DBConnImp](#) \* *m\_imp*

### 7.7.1 Detailed Description

Database connection class.

### 7.7.2 Constructor & Destructor Documentation

#### 7.7.2.1 DBConn::DBConn ( [DBConnImp](#) \* *imp* ) [explicit]

Constructor.

##### Parameters

<i>imp</i>	Pointer to database implementation object.
------------	--

#### 7.7.2.2 glldb::DBConn::DBConn ( const [DBConn](#) & )

Deleted copy constructor

### 7.7.3 Member Function Documentation

#### 7.7.3.1 [DBConn](#)& glldb::DBConn::operator= ( const [DBConn](#) & )

Deleted assignment operator

#### 7.7.3.2 void [DBConn](#)::query ( std::string *sql\_query* )

Runs an SQL query.

##### Parameters

<i>sql_query</i>	The query.
------------------	------------

##### Returns

A [Table](#) object containing the results.

#### 7.7.3.3 [Table](#) [DBConn](#)::select ( std::string *query* )

Runs an SQL SELECT query.

## Parameters

<i>query</i>	The query.
--------------	------------

## Returns

A [Table](#) object containing the results.

## 7.7.4 Member Data Documentation

### 7.7.4.1 DBConnImp\* glldb::DBConn::m\_imp [private]

Pointer to database implementation object.

The documentation for this class was generated from the following files:

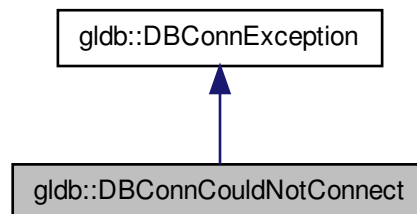
- lib/database/[dbconn.h](#)
- lib/database/[dbconn.cpp](#)

## 7.8 glldb::DBConnCouldNotConnect Class Reference

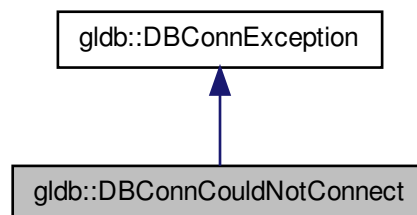
Could not connect to database exception class.

```
#include <dbconn.h>
```

Inheritance diagram for glldb::DBConnCouldNotConnect:



Collaboration diagram for glldb::DBConnCouldNotConnect:





## Public Member Functions

- [DBConnCouldNotConnect](#) (const std::string &msg)  
*Constructor.*

### 7.8.1 Detailed Description

Could not connect to database exception class.

### 7.8.2 Constructor & Destructor Documentation

**7.8.2.1** `glldb::DBConnCouldNotConnect::DBConnCouldNotConnect ( const std::string & msg ) [inline], [explicit]`

Constructor.

#### Parameters

<i>msg</i>	Database error message
------------	------------------------

The documentation for this class was generated from the following file:

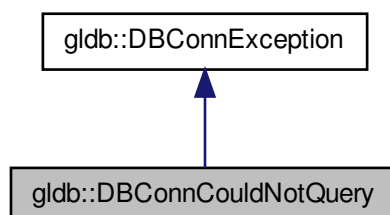
- lib/database/[dbconn.h](#)

## 7.9 glldb::DBConnCouldNotQuery Class Reference

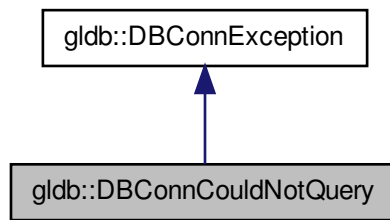
Could not execute database query exception class.

```
#include <dbconn.h>
```

Inheritance diagram for glldb::DBConnCouldNotQuery:



Collaboration diagram for `gldb::DBConnCouldNotQuery`:



## Public Member Functions

- [`DBConnCouldNotQuery`](#) (`const std::string &msg`)  
*Constructor.*

### 7.9.1 Detailed Description

Could not execute database query exception class.

### 7.9.2 Constructor & Destructor Documentation

7.9.2.1 `gldb::DBConnCouldNotQuery::DBConnCouldNotQuery ( const std::string & msg ) [inline], [explicit]`

Constructor.

#### Parameters

<code>msg</code>	Database error message
------------------	------------------------

The documentation for this class was generated from the following file:

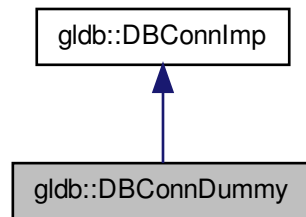
- `lib/database/dbconn.h`

## 7.10 gldb::DBConnDummy Class Reference

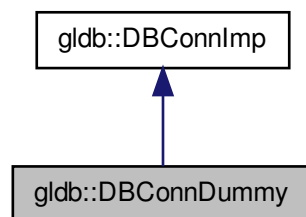
Dummy database implementation class.

```
#include <dbconn_dummy_imp.h>
```

Inheritance diagram for glldb::DBConnDummy:



Collaboration diagram for glldb::DBConnDummy:



## Public Member Functions

- [DBConnDummy](#) (const std::string database, const std::string hostname, const std::string username, const std::string password)  
*Constructor.*
- [DBConnDummy](#) (const [DBConnDummy](#) &)
- virtual [~DBConnDummy](#) ()
- [DBConnDummy](#) & [operator=](#) (const [DBConnDummy](#) &)
- [Table select](#) (std::string query)  
*Fakes running of an SQL SELECT query.*

### 7.10.1 Detailed Description

Dummy database implementation class.

### 7.10.2 Constructor & Destructor Documentation

7.10.2.1 [DBConnDummy::DBConnDummy](#) ( const std::string *database*, const std::string *hostname*, const std::string *username*, const std::string *password* )

Constructor.

## Parameters

<i>database</i>	The name of the Dummy database.
<i>hostname</i>	The hostname of the server.
<i>username</i>	The username to log into the database.
<i>password</i>	The password to log into the database.

7.10.2.2 `gldb::DBConnDummy::DBConnDummy ( const DBConnDummy & )`

Deleted copy constructor

7.10.2.3 `DBConnDummy::~~DBConnDummy ( ) [virtual]`

Destructor

## 7.10.3 Member Function Documentation

7.10.3.1 `DBConnDummy& gldb::DBConnDummy::operator= ( const DBConnDummy & )`

Deleted assignment operator

7.10.3.2 `Table DBConnDummy::select ( std::string query ) [virtual]`

Fakes running of an SQL SELECT query.

## Parameters

<i>query</i>	Any query.
--------------	------------

## Returns

A [Table](#) object containing dummy results.

Implements [gldb::DBConnImp](#).

The documentation for this class was generated from the following files:

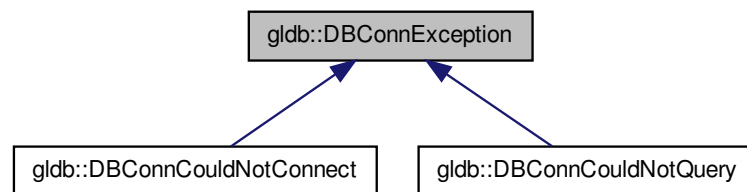
- [lib/database\\_imp/dummy/dbconn\\_dummy\\_imp.h](#)
- [lib/database\\_imp/dummy/dbconn\\_dummy\\_imp.cpp](#)

7.11 `gldb::DBConnException` Class Reference

Base database connection exception class.

```
#include <dbconn.h>
```

Inheritance diagram for glldb::DBConnException:



## Public Member Functions

- [DBConnException](#) (const std::string &msg)  
*Constructor.*

### 7.11.1 Detailed Description

Base database connection exception class.

### 7.11.2 Constructor & Destructor Documentation

7.11.2.1 `glldb::DBConnException::DBConnException ( const std::string & msg )` `[inline], [explicit]`

Constructor.

#### Parameters

<i>msg</i>	Database error message
------------	------------------------

The documentation for this class was generated from the following file:

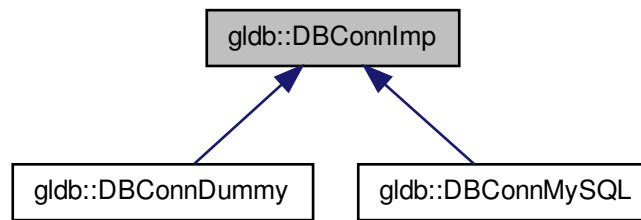
- lib/database/[dbconn.h](#)

## 7.12 glldb::DBConnImp Class Reference

Abstract database implementation base class.

```
#include <dbconnimp.h>
```

Inheritance diagram for `gldb::DBConnImp`:



## Public Member Functions

- [DBConnImp](#) ()
- virtual [~DBConnImp](#) ()
- virtual void [query](#) (std::string sql\_query)=0  
*Runs an SQL query.*
- virtual [Table select](#) (std::string query)=0  
*Runs an SQL SELECT query.*

### 7.12.1 Detailed Description

Abstract database implementation base class.

### 7.12.2 Constructor & Destructor Documentation

7.12.2.1 `gldb::DBConnImp::DBConnImp ( )` `[inline]`

Constructor

7.12.2.2 `virtual gldb::DBConnImp::~~DBConnImp ( )` `[inline],[virtual]`

Destructor

### 7.12.3 Member Function Documentation

7.12.3.1 `virtual void gldb::DBConnImp::query ( std::string sql_query )` `[pure virtual]`

Runs an SQL query.

Parameters

<i>sql_query</i>	The query.
------------------	------------

Implemented in [gldb::DBConnMySQL](#).

7.12.3.2 virtual Table glldb::DBConnImp::select ( std::string *query* ) [pure virtual]

Runs an SQL SELECT query.

#### Parameters

<i>query</i>	The query.
--------------	------------

#### Returns

A [Table](#) object containing the results.

Implemented in [glldb::DBConnMySQL](#), and [glldb::DBConnDummy](#).

The documentation for this class was generated from the following file:

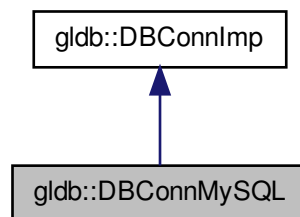
- lib/database/[dbconnimp.h](#)

## 7.13 glldb::DBConnMySQL Class Reference

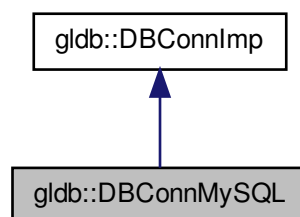
MySQL database implementation class.

```
#include <dbconn_mysql_imp.h>
```

Inheritance diagram for glldb::DBConnMySQL:



Collaboration diagram for glldb::DBConnMySQL:



## Public Member Functions

- [DBConnMySQL](#) (const std::string database, const std::string hostname, const std::string username, const std::string password)  
*Constructor.*
- [DBConnMySQL](#) (const [DBConnMySQL](#) &)
- virtual [~DBConnMySQL](#) ()
- [DBConnMySQL](#) & [operator=](#) (const [DBConnMySQL](#) &)
- virtual void [query](#) (std::string sql\_query)  
*Runs an SQL query.*
- virtual [Table select](#) (std::string query)  
*Runs an SQL SELECT query.*

## Private Attributes

- MySQL \* [m\\_conn](#)

### 7.13.1 Detailed Description

MySQL database implementation class.

### 7.13.2 Constructor & Destructor Documentation

- 7.13.2.1 [DBConnMySQL::DBConnMySQL](#) ( const std::string *database*, const std::string *hostname*, const std::string *username*, const std::string *password* )

Constructor.

#### Parameters

<i>database</i>	The name of the MySQL database.
<i>hostname</i>	The hostname of the server.
<i>username</i>	The username to log into the database.
<i>password</i>	The password to log into the database.

#### Exceptions

<a href="#">DBConnCouldNotConnect</a>	If could not connect to database.
---------------------------------------	-----------------------------------

- 7.13.2.2 [gldb::DBConnMySQL::DBConnMySQL](#) ( const [DBConnMySQL](#) & )

Deleted copy constructor

- 7.13.2.3 [DBConnMySQL::~~DBConnMySQL](#) ( ) [virtual]

Destructor

### 7.13.3 Member Function Documentation



## 7.13.3.1 DBConnMySQL&amp; glldb::DBConnMySQL::operator= ( const DBConnMySQL &amp; )

Deleted assignment operator

7.13.3.2 void DBConnMySQL::query ( std::string *sql\_query* ) [virtual]

Runs an SQL query.

## Parameters

<i>sql_query</i>	The query.
------------------	------------

## Exceptions

<a href="#">DBConnCouldNotQuery</a>	If could not successfully execute query.
-------------------------------------	--

Implements [glldb::DBConnImp](#).

7.13.3.3 Table DBConnMySQL::select ( std::string *query* ) [virtual]

Runs an SQL SELECT query.

## Parameters

<i>query</i>	The query.
--------------	------------

## Returns

A [Table](#) object containing the results.

## Exceptions

<a href="#">DBConnCouldNotQuery</a>	If could not successfully execute query.
-------------------------------------	--

Implements [glldb::DBConnImp](#).

## 7.13.4 Member Data Documentation

## 7.13.4.1 MYSQL\* glldb::DBConnMySQL::m\_conn [private]

The initialized MySQL handle.

The documentation for this class was generated from the following files:

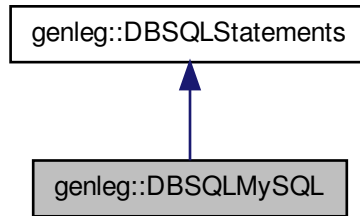
- lib/database\_imp/mysql/[dbconn\\_mysql\\_imp.h](#)
- lib/database\_imp/mysql/[dbconn\\_mysql\\_imp.cpp](#)

## 7.14 genleg::DBSQLMySQL Class Reference

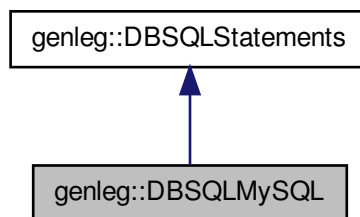
MySQL SQL statements class.

```
#include <dbsql_mysql.h>
```

Inheritance diagram for genleg::DBSQLMySQL:



Collaboration diagram for genleg::DBSQLMySQL:



## Additional Inherited Members

### 7.14.1 Detailed Description

MySQL SQL statements class.

The documentation for this class was generated from the following file:

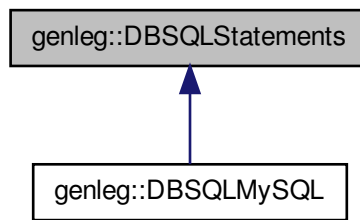
- lib/dbsql/[dbsql\\_mysql.h](#)

## 7.15 genleg::DBSQLStatements Class Reference

SQL statements class.

```
#include <dbsqlstatements.h>
```

Inheritance diagram for genleg::DBSQLStatements:



## Public Member Functions

- [DBSQLStatements](#) ()
- virtual [~DBSQLStatements](#) ()
- virtual std::string [create\\_table](#) (const std::string table\_name) const  
*Returns a SQL statement for creating a table.*
- virtual std::string [drop\\_table](#) (const std::string table\_name) const  
*Returns a SQL statement for dropping a table.*
- virtual std::string [create\\_view](#) (const std::string view\_name) const  
*Returns a SQL statement for creating a view.*
- virtual std::string [drop\\_view](#) (const std::string view\_name) const  
*Returns a SQL statement for dropping a view.*

### 7.15.1 Detailed Description

SQL statements class.

### 7.15.2 Constructor & Destructor Documentation

#### 7.15.2.1 DBSQLStatements::DBSQLStatements ( )

Constructor

#### 7.15.2.2 DBSQLStatements::~~DBSQLStatements ( ) [virtual]

Destructor

### 7.15.3 Member Function Documentation

#### 7.15.3.1 std::string DBSQLStatements::create\_table ( const std::string table\_name ) const [virtual]

Returns a SQL statement for creating a table.

Parameters

<i>table_name</i>	The table to create.
-------------------	----------------------

**Returns**

The SQL statement to create the table.

**7.15.3.2** `std::string DBSQLStatements::create_view ( const std::string view_name ) const` [virtual]

Returns a SQL statement for creating a view.

**Parameters**

<i>view_name</i>	The view to create.
------------------	---------------------

**Returns**

The SQL statement to create the view.

**7.15.3.3** `std::string DBSQLStatements::drop_table ( const std::string table_name ) const` [virtual]

Returns a SQL statement for dropping a table.

**Parameters**

<i>table_name</i>	The table to drop.
-------------------	--------------------

**Returns**

The SQL statement to drop the table.

**7.15.3.4** `std::string DBSQLStatements::drop_view ( const std::string view_name ) const` [virtual]

Returns a SQL statement for dropping a view.

**Parameters**

<i>view_name</i>	The view to drop.
------------------	-------------------

**Returns**

The SQL statement to drop the view.

The documentation for this class was generated from the following files:

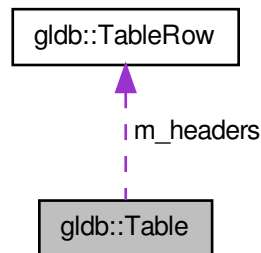
- [lib/dbsql/dbsqlstatements.h](#)
- [lib/dbsql/dbsqlstatements.cpp](#)

## 7.16 glldb::Table Class Reference

Database table class.

```
#include <table.h>
```

Collaboration diagram for glldb::Table:



## Public Member Functions

- [Table](#) (const [TableRow](#) &headers)  
*Constructor.*
- [~Table](#) ()
- [size\\_t num\\_fields](#) () const  
*Returns the number of fields in each row.*
- [size\\_t num\\_records](#) () const  
*Returns the number of record in the table.*
- void [set\\_quoted](#) (std::vector< bool > &vec)  
*Sets the quote flags for the records.*
- const [TableRow](#) & [get\\_headers](#) () const  
*Returns the field names.*
- const [TableRow](#) & [operator\[\]](#) (const [size\\_t](#) idx) const  
*Overloaded index operator.*
- void [append\\_record](#) (const [TableRow](#) &new\_record)  
*Appends a record to the table.*
- std::string [insert\\_query](#) (const std::string table\_name, const [size\\_t](#) idx)  
*Creates an SQL INSERT query from a table record.*

## Static Public Member Functions

- static [Table create\\_from\\_file](#) (const std::string filename, const char delim)  
*Creates a table from an input file.*

## Private Attributes

- [TableRow](#) m\_headers
- std::vector< [TableRow](#) > m\_records
- std::vector< bool > m\_quoted

### 7.16.1 Detailed Description

Database table class.

## 7.16.2 Constructor & Destructor Documentation

### 7.16.2.1 `Table::Table ( const TableRow & headers ) [explicit]`

Constructor.

#### Parameters

<i>headers</i>	<a href="#">Table</a> row containing field names.
----------------	---

### 7.16.2.2 `Table::~~Table ( )`

Destructor

## 7.16.3 Member Function Documentation

### 7.16.3.1 `void Table::append_record ( const TableRow & new_record )`

Appends a record to the table.

#### Parameters

<i>new_record</i>	The record to append.
-------------------	-----------------------

### 7.16.3.2 `Table Table::create_from_file ( const std::string filename, const char delim ) [static]`

Creates a table from an input file.

#### Parameters

<i>filename</i>	The name of the input file.
<i>delim</i>	The delimiting character.

#### Returns

The table.

#### Exceptions

<a href="#">TableBadInputFile</a>	on badly formed input file.
<a href="#">TableCouldNotOpenInputFile</a>	on bad filename.

### 7.16.3.3 `const TableRow & Table::get_headers ( ) const`

Returns the field names.

#### Returns

The field names.

**7.16.3.4** `std::string Table::insert_query ( const std::string table_name, const size_t idx )`

Creates an SQL INSERT query from a table record.

**Parameters**

<i>table_name</i>	The name of the table into which to INSERT.
<i>idx</i>	The index of the record.

**Returns**

A string containing the query.

**7.16.3.5** `size_t Table::num_fields ( ) const`

Returns the number of fields in each row.

**Returns**

The number of fields in each row.

**7.16.3.6** `size_t Table::num_records ( ) const`

Returns the number of record in the table.

**Returns**

The number of records in the table.

**7.16.3.7** `const TableRow & Table::operator[] ( const size_t idx ) const`

Overloaded index operator.

**Parameters**

<i>idx</i>	The zero-based index of the record.
------------	-------------------------------------

**Returns**

The selected record.

**7.16.3.8** `void Table::set_quoted ( std::vector< bool > & vec )`

Sets the quote flags for the records.

**Parameters**

<i>vec</i>	A vector of bools. The size must match the size of the records.
------------	---

**7.16.4 Member Data Documentation**

#### 7.16.4.1 TableRow gldb::Table::m\_headers [private]

The names of the fields

#### 7.16.4.2 std::vector<bool> gldb::Table::m\_quoted [private]

A vector to show if fields should be quoted for INSERT

#### 7.16.4.3 std::vector<TableRow> gldb::Table::m\_records [private]

A vector of the records

The documentation for this class was generated from the following files:

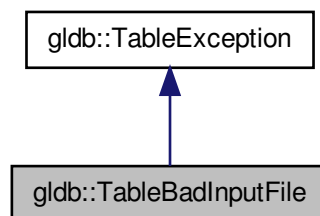
- lib/database/[table.h](#)
- lib/database/[table.cpp](#)

## 7.17 gldb::TableBadInputFile Class Reference

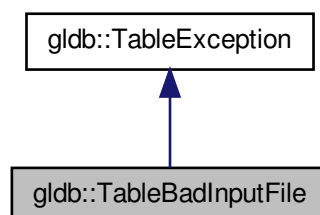
Could not connect to database exception class.

```
#include <table.h>
```

Inheritance diagram for gldb::TableBadInputFile:



Collaboration diagram for gldb::TableBadInputFile:





### 7.17.1 Detailed Description

Could not connect to database exception class.

The documentation for this class was generated from the following file:

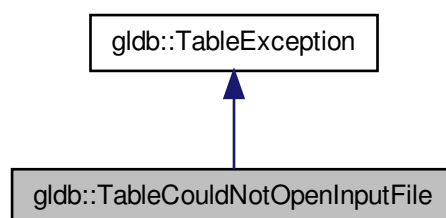
- lib/database/[table.h](#)

## 7.18 glldb::TableCouldNotOpenInputFile Class Reference

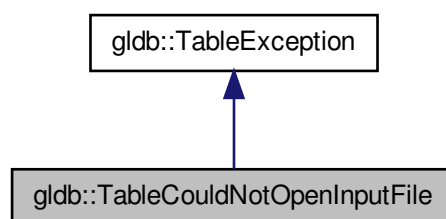
Could not connect to database exception class.

```
#include <table.h>
```

Inheritance diagram for glldb::TableCouldNotOpenInputFile:



Collaboration diagram for glldb::TableCouldNotOpenInputFile:



### 7.18.1 Detailed Description

Could not connect to database exception class.

The documentation for this class was generated from the following file:

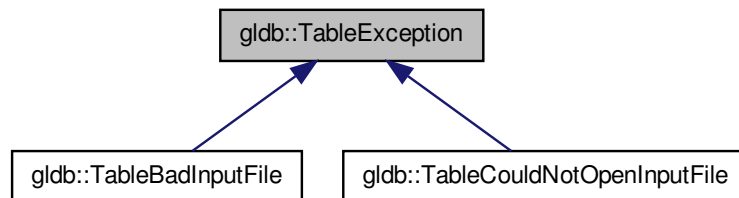
- lib/database/[table.h](#)

## 7.19 gldb::TableException Class Reference

Base database connection exception class.

```
#include <table.h>
```

Inheritance diagram for gldb::TableException:



### 7.19.1 Detailed Description

Base database connection exception class.

The documentation for this class was generated from the following file:

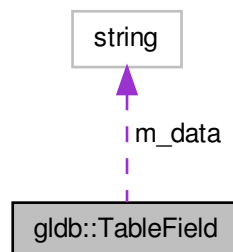
- lib/database/[table.h](#)

## 7.20 gldb::TableField Class Reference

Database table field class.

```
#include <tablefield.h>
```

Collaboration diagram for gldb::TableField:



### Public Member Functions

- [TableField](#) (const char \*data)

- Constructor accepting `const char * data`.*
- [TableField](#) (const std::string &data)
  - Constructor accepting `std::string data`.*
- [~TableField](#) ()
- [size\\_t length](#) () const
  - Returns the length of the field.*
- [operator std::string](#) () const
  - Overridden conversion operator.*
- [TableField](#) & [operator=](#) (const char \*data)
  - Overridden assignment operator for `const char *`.*
- [TableField](#) & [operator=](#) (const std::string &data)
  - Overridden assignment operator for `std::string`.*
- [char & operator\[\]](#) (const [size\\_t](#) idx)
  - Overridden index operator.*
- [const char & operator\[\]](#) (const [size\\_t](#) idx) const
  - Overridden index operator.*
- [TableField](#) & [operator+=](#) (const char &c)
  - Overridden compound assignment operator.*
- [TableField](#) & [operator+=](#) (const std::string &data)
  - Overridden compound assignment operator.*

## Private Attributes

- std::string [m\\_data](#)

## Friends

- std::ostream & [operator<<](#) (std::ostream &out, const [TableField](#) &field)
  - Overridden << operator for printing a field.*

## 7.20.1 Detailed Description

Database table field class.

## 7.20.2 Constructor & Destructor Documentation

### 7.20.2.1 [TableField::TableField](#) ( `const char * data` ) [explicit]

Constructor accepting `const char * data`.

#### Parameters

<i>data</i>	The initial contents of the field.
-------------	------------------------------------

### 7.20.2.2 [TableField::TableField](#) ( `const std::string & data` ) [explicit]

Constructor accepting `std::string data`.

## Parameters

<i>data</i>	The initial contents of the field.
-------------	------------------------------------

7.20.2.3 `TableField::~~TableField ( )`

Destructor

## 7.20.3 Member Function Documentation

7.20.3.1 `size_t TableField::length ( ) const`

Returns the length of the field.

## Returns

The length of the field.

7.20.3.2 `TableField::operator std::string ( ) const`

Overridden conversion operator.

Returns the field contents as a string.

7.20.3.3 `TableField & TableField::operator+= ( const char & c )`

Overridden compound assignment operator.

## Parameters

<i>c</i>	The character to append to the field.
----------	---------------------------------------

## Returns

A reference to the same field.

7.20.3.4 `TableField & TableField::operator+= ( const std::string & data )`

Overridden compound assignment operator.

## Parameters

<i>data</i>	The string to append to the field.
-------------	------------------------------------

## Returns

A reference to the same field.

7.20.3.5 `TableField & TableField::operator= ( const char * data )`

Overridden assignment operator for `const char *`.

## Parameters

<i>data</i>	The new contents of the field.
-------------	--------------------------------

## Returns

A reference to the same field.

7.20.3.6 TableField & TableField::operator= ( const std::string & *data* )

Overridden assignment operator for `std::string`.

## Parameters

<i>data</i>	The new contents of the field.
-------------	--------------------------------

## Returns

A reference to the same field.

7.20.3.7 char & TableField::operator[] ( const size\_t *idx* )

Overridden index operator.

## Parameters

<i>idx</i>	The desired index.
------------	--------------------

## Returns

A reference to the character at the specified index.

7.20.3.8 const char & TableField::operator[] ( const size\_t *idx* ) const

Overridden index operator.

## Parameters

<i>idx</i>	The desired index.
------------	--------------------

## Returns

A const reference to the character at the specified index.

## 7.20.4 Friends And Related Function Documentation

7.20.4.1 std::ostream& operator<< ( std::ostream & *out*, const TableField & *field* ) [friend]

Overridden << operator for printing a field.

## Parameters

<i>out</i>	The ostream to which to print.
<i>field</i>	A reference to the field.

## Returns

A reference to `out`.

## 7.20.5 Member Data Documentation

### 7.20.5.1 `std::string gldb::TableField::m_data` `[private]`

The field contents

The documentation for this class was generated from the following files:

- [lib/database/tablefield.h](#)
- [lib/database/tablefield.cpp](#)

## 7.21 `gldb::TableRow` Class Reference

Database table row class.

```
#include <tablerow.h>
```

### Public Member Functions

- [TableRow](#) ()  
*Constructor with initial number of fields.*
- [TableRow](#) (const size\_t size)  
*Constructor with string vector.*
- [TableRow](#) (std::vector< std::string > &vec)  
*Constructor with string vector.*
- [~TableRow](#) ()
- size\_t [size](#) () const  
*Returns the number of fields.*
- [TableField](#) & [operator\[\]](#) (const size\_t idx)  
*Overridden index operator.*
- const [TableField](#) & [operator\[\]](#) (const size\_t idx) const  
*Overridden index operator.*
- void [append\\_field](#) (const char \*new\_field)  
*Appends a field to the row.*
- void [append\\_field](#) (const std::string &new\_field)  
*Appends a field to the row.*
- void [append\\_field](#) (const [TableField](#) &new\_field)  
*Appends a field to the row.*
- void [print](#) (std::ostream &stream) const  
*Prints a row.*
- std::string [record\\_string](#) (const std::vector< bool > &quoted)  
*Creates a comma separated string of fields.*
- std::string [record\\_string](#) ()  
*Creates an unquoted comma separated string of fields.*

### Private Attributes

- std::vector< [TableField](#) > [m\\_fields](#)

### 7.21.1 Detailed Description

Database table row class.

### 7.21.2 Constructor & Destructor Documentation

#### 7.21.2.1 TableRow::TableRow ( )

Default constructor

#### 7.21.2.2 TableRow::TableRow ( const size\_t *size* ) [explicit]

Constructor with initial number of fields.

##### Parameters

<i>size</i>	The initial number of fields.
-------------	-------------------------------

#### 7.21.2.3 TableRow::TableRow ( std::vector< std::string > & *vec* ) [explicit]

Constructor with string vector.

##### Parameters

<i>vec</i>	The vector.
------------	-------------

#### 7.21.2.4 TableRow::~~TableRow ( )

Destructor

### 7.21.3 Member Function Documentation

#### 7.21.3.1 void TableRow::append\_field ( const char \* *new\_field* )

Appends a field to the row.

##### Parameters

<i>new_field</i>	The contents of the new field.
------------------	--------------------------------

#### 7.21.3.2 void TableRow::append\_field ( const std::string & *new\_field* )

Appends a field to the row.

##### Parameters

<i>new_field</i>	The contents of the new field.
------------------	--------------------------------

#### 7.21.3.3 void TableRow::append\_field ( const TableField & *new\_field* )

Appends a field to the row.

## Parameters

<i>new_field</i>	A field from which to copy.
------------------	-----------------------------

7.21.3.4 **TableField & TableRow::operator[]** ( `const size_t idx` )

Overridden index operator.

## Parameters

<i>idx</i>	The zero-based index of the field.
------------	------------------------------------

## Returns

A reference to the field at the specified index.

7.21.3.5 **const TableField & TableRow::operator[]** ( `const size_t idx` ) **const**

Overridden index operator.

## Parameters

<i>idx</i>	The zero-based index of the field.
------------	------------------------------------

## Returns

A const reference to the field at the specified index.

7.21.3.6 **void TableRow::print** ( `std::ostream & stream` ) **const**

Prints a row.

## Parameters

<i>stream</i>	The ostream to which to print.
---------------	--------------------------------

7.21.3.7 **std::string TableRow::record\_string** ( `const std::vector< bool > & quoted` )

Creates a comma separated string of fields.

## Parameters

<i>quoted</i>	A vector of <code>bool</code> , for each field <code>true</code> means that field will be enclosed in single quotes in the comma separated string, <code>false</code> means it will not be.
---------------	---

## Returns

The comma separated string.

7.21.3.8 **std::string TableRow::record\_string** ( )

Creates an unquoted comma separated string of fields.



**Returns**

The unquoted comma separated string.

**7.21.3.9** `size_t TableRow::size ( ) const`

Returns the number of fields.

**Returns**

The number of fields.

**7.21.4 Member Data Documentation****7.21.4.1** `std::vector<TableField> glldb::TableRow::m_fields` `[private]`

A vector of fields

The documentation for this class was generated from the following files:

- lib/database/[tablerow.h](#)
- lib/database/[tablerow.cpp](#)



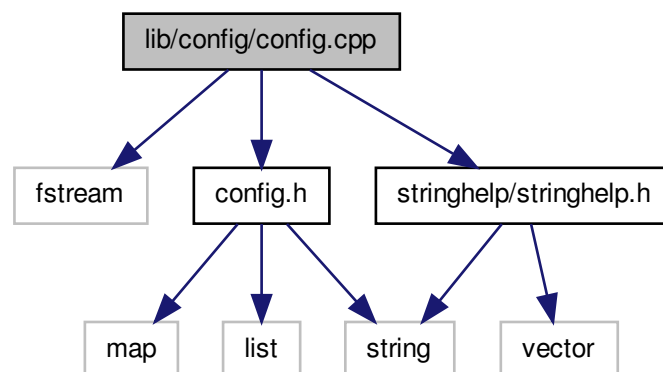
## Chapter 8

# File Documentation

### 8.1 lib/config/config.cpp File Reference

Implementation of program configurations class.

```
#include <fstream>
#include "config.h"
#include "stringhelp/stringhelp.h"
Include dependency graph for config.cpp:
```



#### 8.1.1 Detailed Description

Implementation of program configurations class.

##### Author

Paul Griffiths

##### Copyright

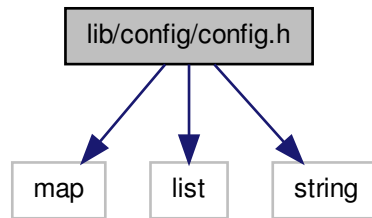
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

## 8.2 lib/config/config.h File Reference

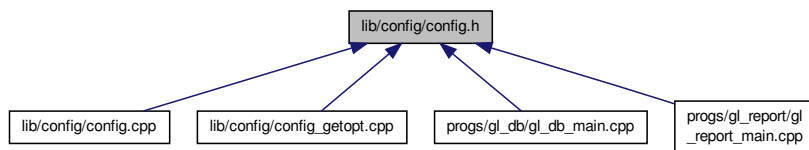
Interface to program configurations class.

```
#include <map>
#include <list>
#include <string>
```

Include dependency graph for config.h:



This graph shows which files directly or indirectly include this file:



### Classes

- class [genleg::ConfigException](#)  
*Configuration module exception base class.*
- class [genleg::ConfigOptionNotSet](#)  
*Exception class for option not set.*
- class [genleg::ConfigBadOption](#)  
*Exception class for bad provided option.*
- class [genleg::ConfigCouldNotOpenFile](#)  
*Exception class for when conf file cannot be opened.*
- class [genleg::ConfigBadConfigFile](#)  
*Exception class for badly formed configuration file.*
- class [genleg::Config](#)  
*Configuration options class.*

### Enumerations

- enum [genleg::Argument](#)  
*Enumeration class for option argument specifications.*

### 8.2.1 Detailed Description

Interface to program configurations class.

#### Author

Paul Griffiths

#### Copyright

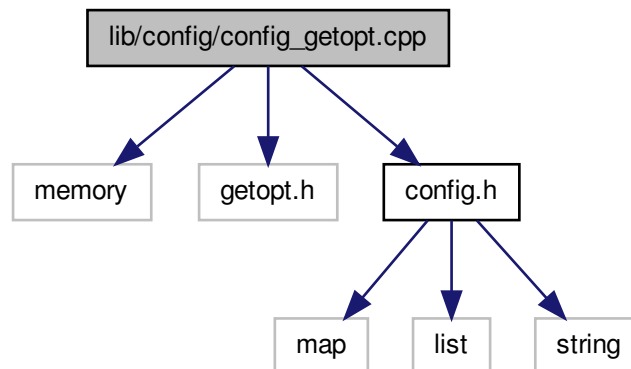
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

## 8.3 lib/config/config\_getopt.cpp File Reference

Implementation of command line functionality.

```
#include <memory>
#include <getopt.h>
#include "config.h"
```

Include dependency graph for config\_getopt.cpp:



#### Macros

- `#define _XOPEN_SOURCE 600`

### 8.3.1 Detailed Description

Implementation of command line functionality. Included in separate file to isolate usage of non-standard getopt library.

#### Author

Paul Griffiths

## Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

## 8.3.2 Macro Definition Documentation

### 8.3.2.1 `#define _XOPEN_SOURCE 600`

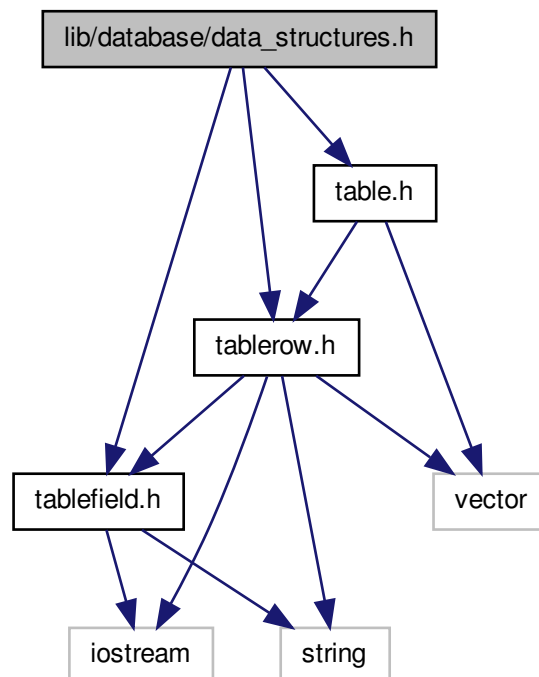
UNIX feature test macro for getopt library

## 8.4 `lib/database/data_structures.h` File Reference

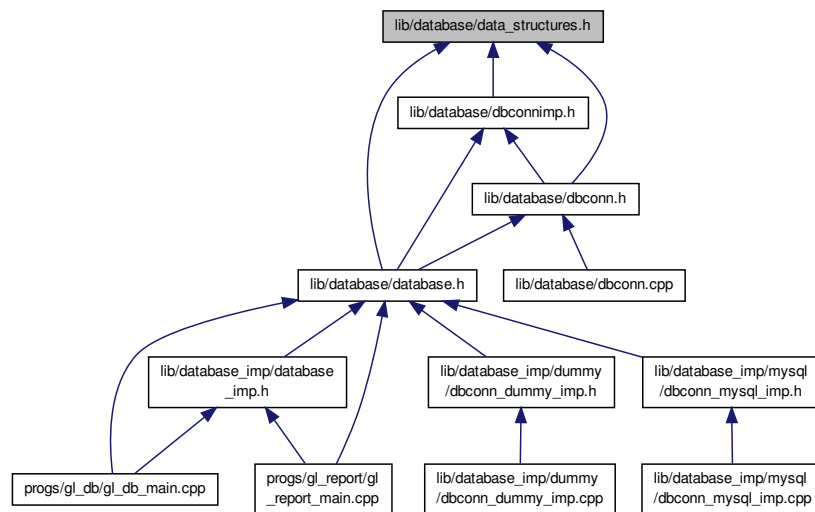
Main interface to database data structures.

```
#include "tablefield.h"  
#include "tablerow.h"  
#include "table.h"
```

Include dependency graph for `data_structures.h`:



This graph shows which files directly or indirectly include this file:



### 8.4.1 Detailed Description

Main interface to database data structures.

#### Author

Paul Griffiths

#### Copyright

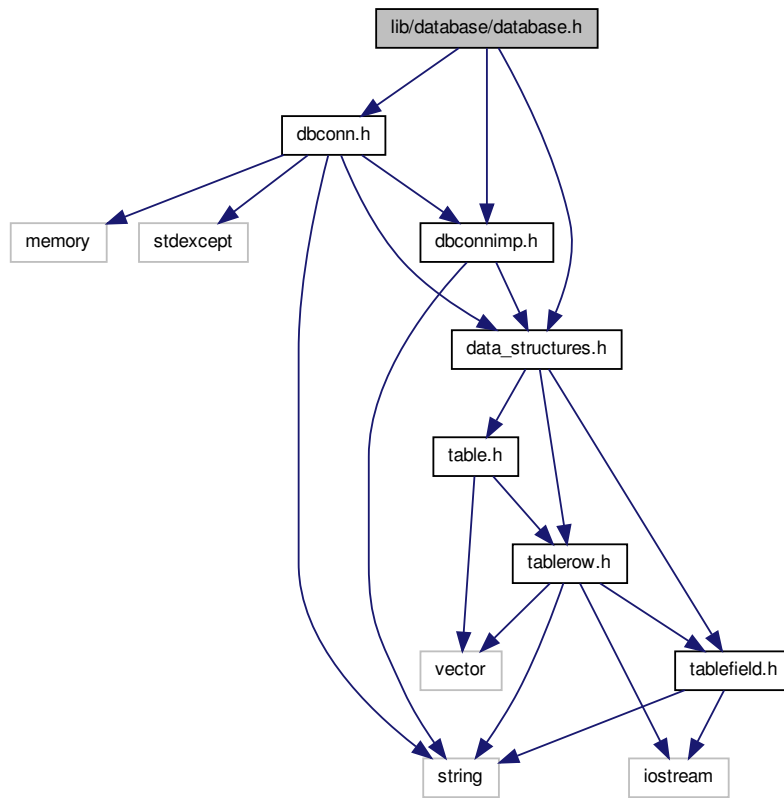
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

## 8.5 lib/database/database.h File Reference

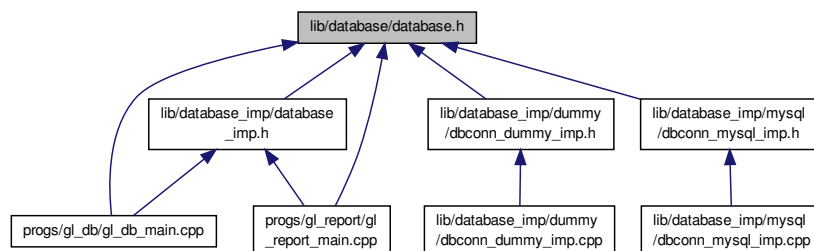
User interface to database functionality.

```
#include "data_structures.h"
#include "dbconnimp.h"
#include "dbconn.h"
```

Include dependency graph for database.h:



This graph shows which files directly or indirectly include this file:



### 8.5.1 Detailed Description

User interface to database functionality.

Author

Paul Griffiths



## Copyright

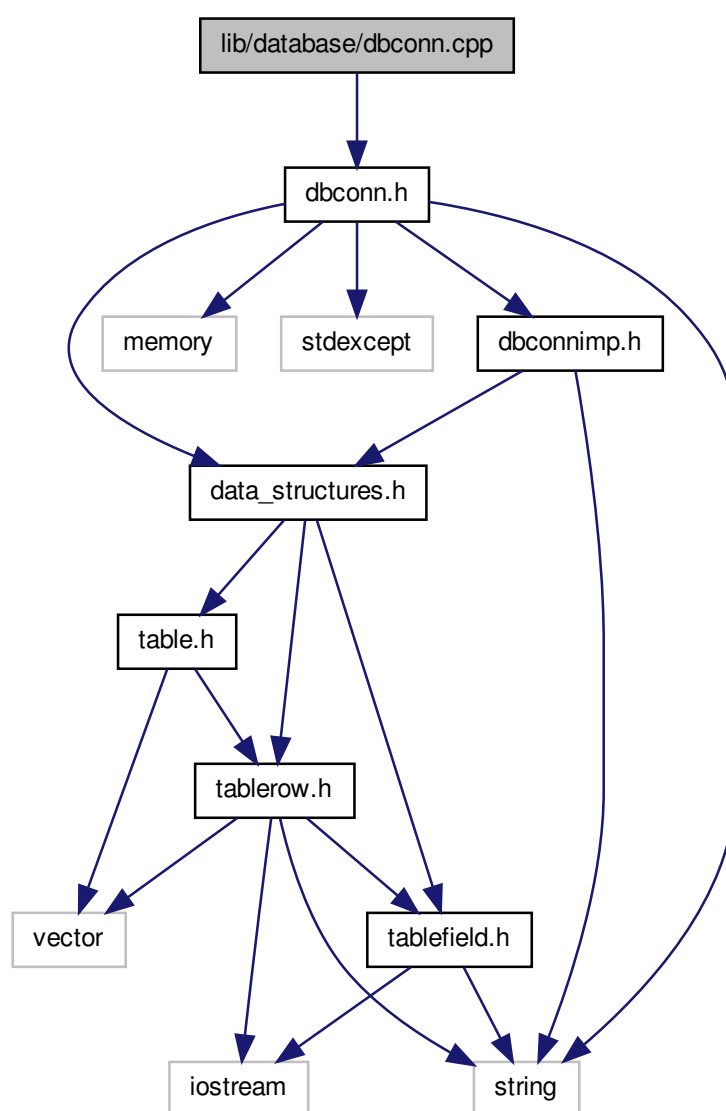
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

## 8.6 lib/database/dbconn.cpp File Reference

Implementation of database connection class.

```
#include "dbconn.h"
```

Include dependency graph for dbconn.cpp:



### 8.6.1 Detailed Description

Implementation of database connection class.

#### Author

Paul Griffiths

#### Copyright

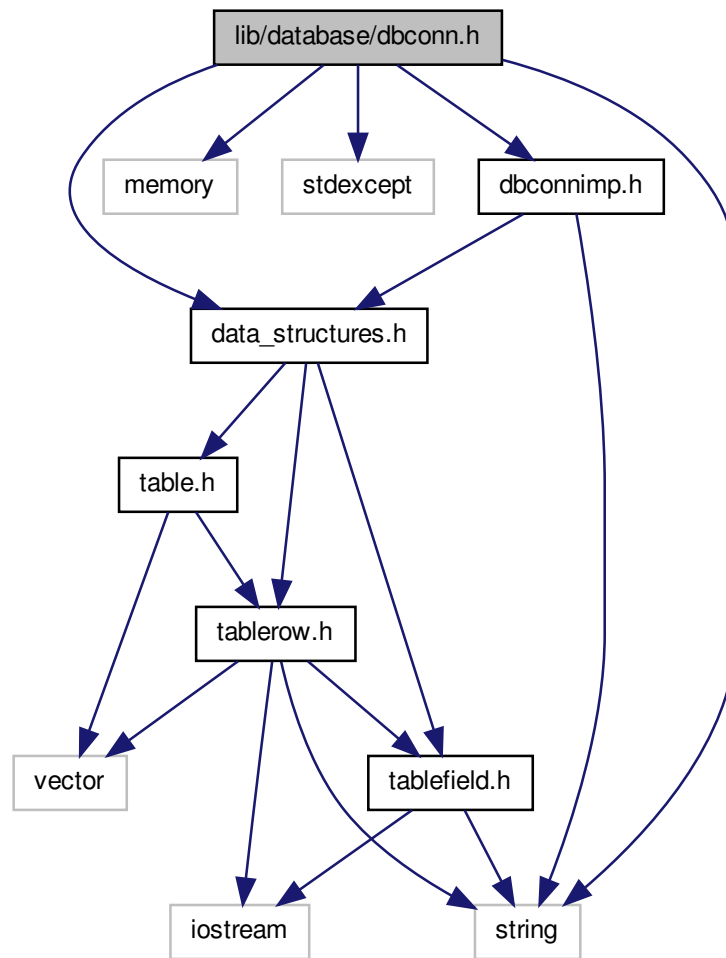
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

## 8.7 lib/database/dbconn.h File Reference

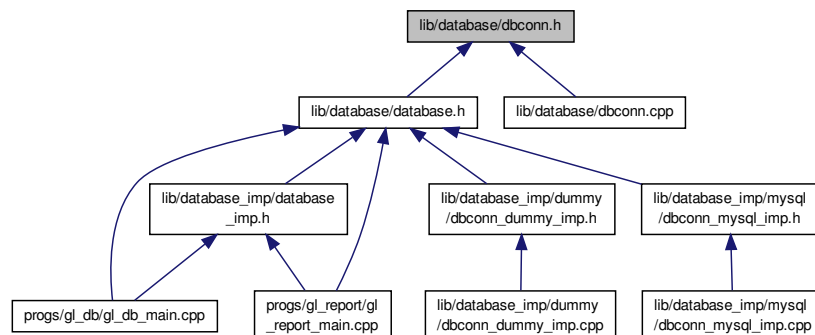
Interface to database connection base class.

```
#include <string>
#include <memory>
#include <stdexcept>
#include "data_structures.h"
#include "dbconnimp.h"
```

Include dependency graph for dbconn.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [gldb::DBConnException](#)

*Base database connection exception class.*

- class [gldb::DBConnCouldNotConnect](#)

*Could not connect to database exception class.*

- class [gldb::DBConnCouldNotQuery](#)

*Could not execute database query exception class.*

- class [gldb::DBConn](#)

*Database connection class.*

### 8.7.1 Detailed Description

Interface to database connection base class.

#### Author

Paul Griffiths

#### Copyright

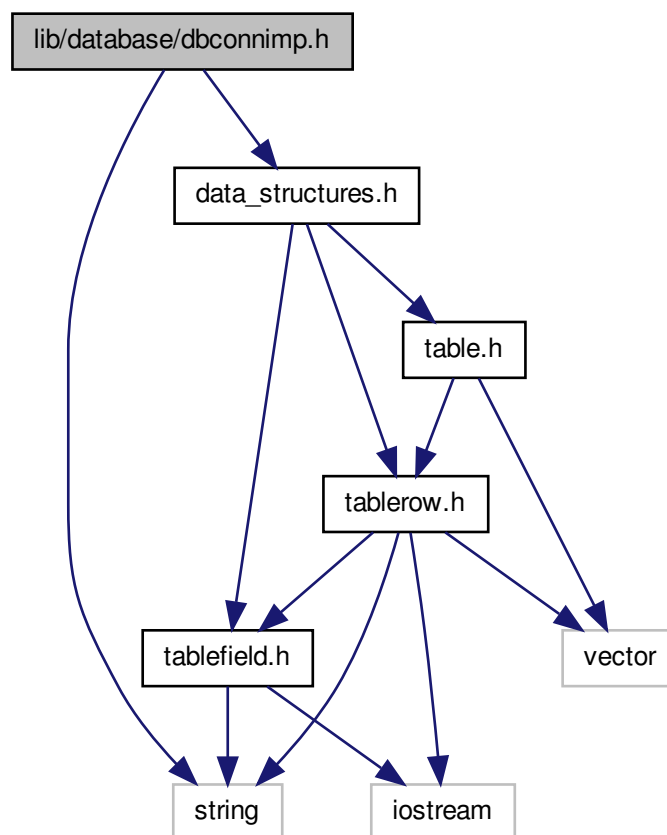
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

## 8.8 lib/database/dbconnimp.h File Reference

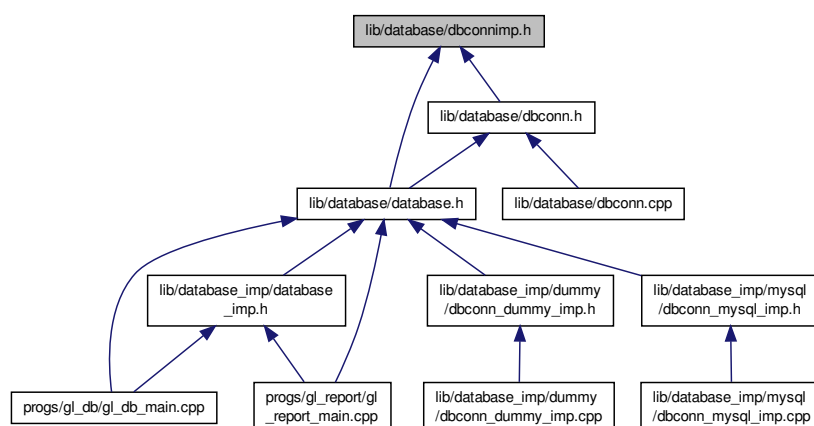
Interface to abstract database implementation base class.

```
#include <string>
#include "data_structures.h"
```

Include dependency graph for dbconnimp.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class `gldb::DBConnImp`  
*Abstract database implementation base class.*

### 8.8.1 Detailed Description

Interface to abstract database implementation base class.

#### Author

Paul Griffiths

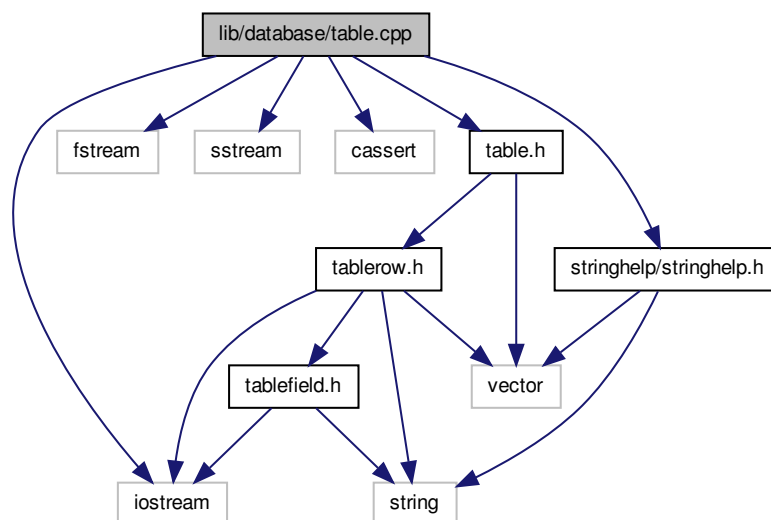
#### Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

## 8.9 lib/database/table.cpp File Reference

Implementation of database table data structure.

```
#include <iostream>
#include <fstream>
#include <sstream>
#include <cassert>
#include "table.h"
#include "stringhelp/stringhelp.h"
Include dependency graph for table.cpp:
```



### 8.9.1 Detailed Description

Implementation of database table data structure.

## Author

Paul Griffiths

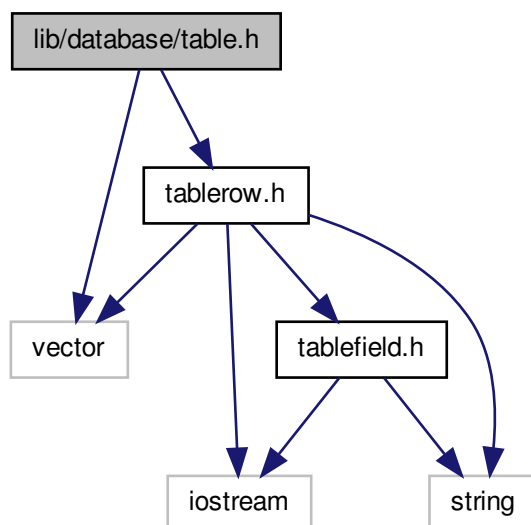
## Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

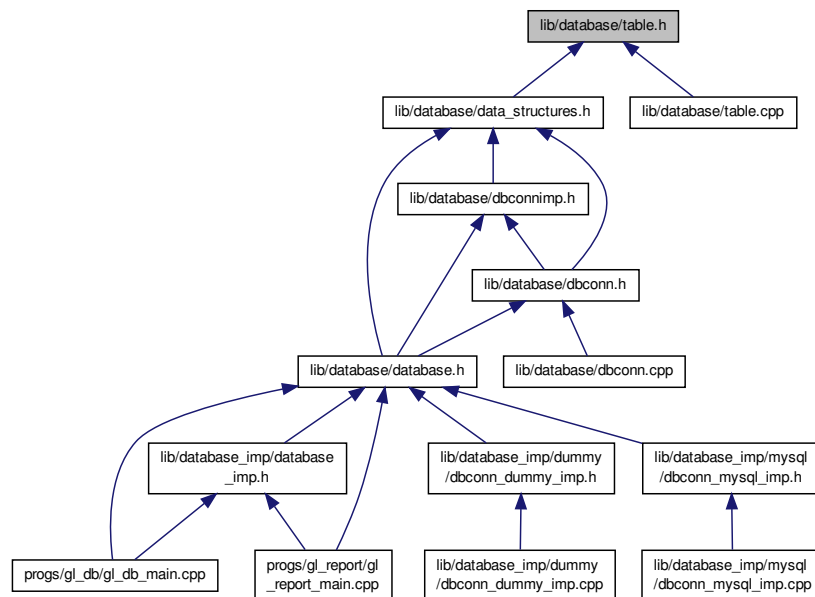
## 8.10 lib/database/table.h File Reference

Interface to database table data structure.

```
#include <vector>
#include "tablerow.h"
Include dependency graph for table.h:
```



This graph shows which files directly or indirectly include this file:



## Classes

- class [gldb::TableException](#)  
*Base database connection exception class.*
- class [gldb::TableBadInputFile](#)  
*Could not connect to database exception class.*
- class [gldb::TableCouldNotOpenInputFile](#)  
*Could not connect to database exception class.*
- class [gldb::Table](#)  
*Database table class.*

### 8.10.1 Detailed Description

Interface to database table data structure.

#### Author

Paul Griffiths

#### Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

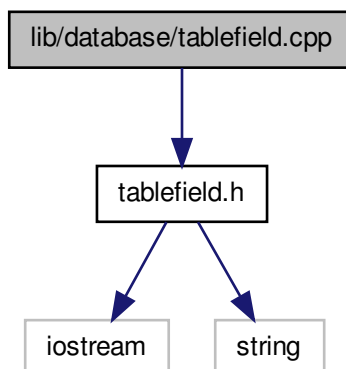
### 8.11 lib/database/tablefield.cpp File Reference

Implementation of database table field class.



```
#include "tablefield.h"
```

Include dependency graph for tablefield.cpp:



### 8.11.1 Detailed Description

Implementation of database table field class.

#### Author

Paul Griffiths

#### Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

## 8.12 lib/database/tablefield.h File Reference

Interface to database table field class.

```
#include <iostream>
#include <string>
```



### 8.12.1 Detailed Description

Interface to database table field class.

#### Author

Paul Griffiths

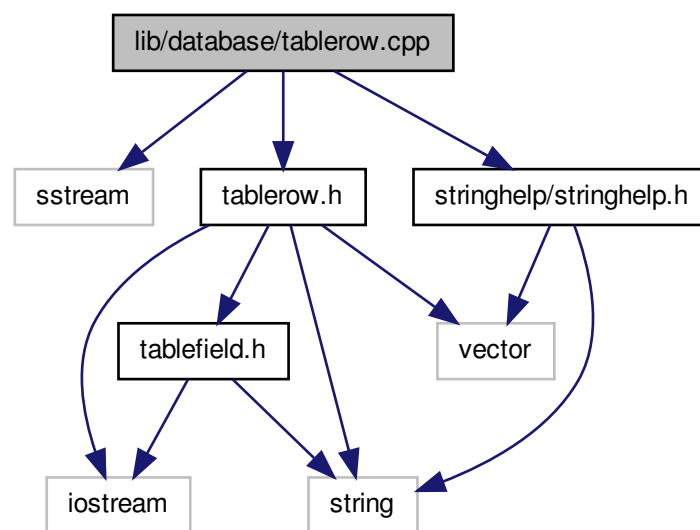
#### Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

## 8.13 lib/database/ablerow.cpp File Reference

Implementation of database table row data structure.

```
#include <sstream>
#include "ablerow.h"
#include "stringhelp/stringhelp.h"
Include dependency graph for ablerow.cpp:
```



### 8.13.1 Detailed Description

Implementation of database table row data structure.

#### Author

Paul Griffiths

#### Copyright

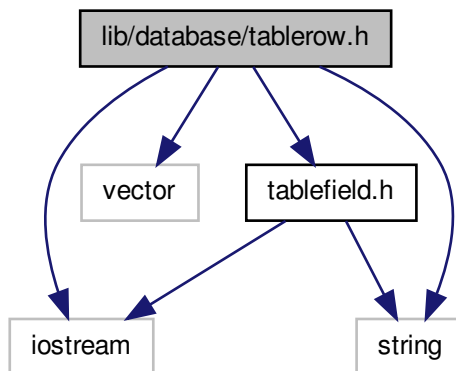
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

## 8.14 lib/database/tablerow.h File Reference

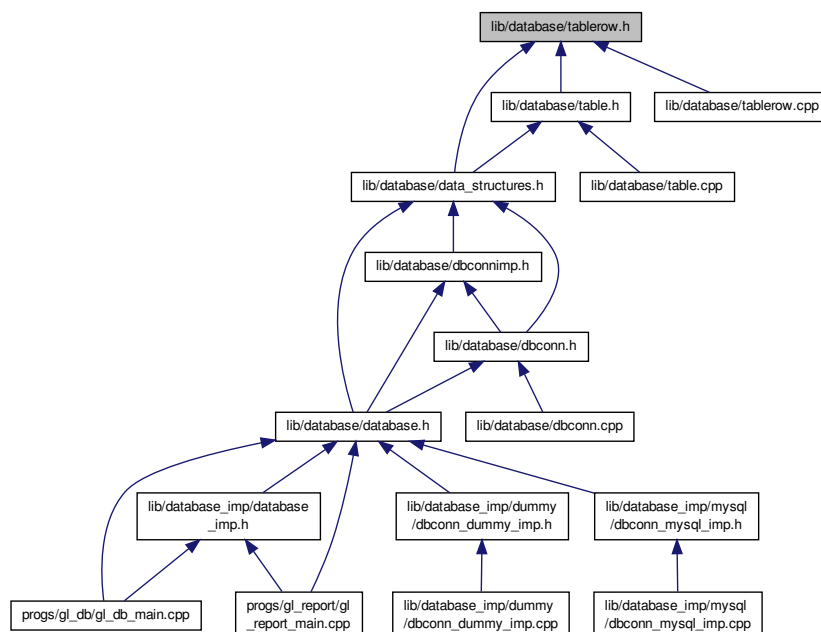
Interface to database table row data structure.

```
#include <iostream>
#include <vector>
#include <string>
#include "tablefield.h"
```

Include dependency graph for tablerow.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [gldb::TableRow](#)

*Database table row class.*

### 8.14.1 Detailed Description

Interface to database table row data structure.

#### Author

Paul Griffiths

#### Copyright

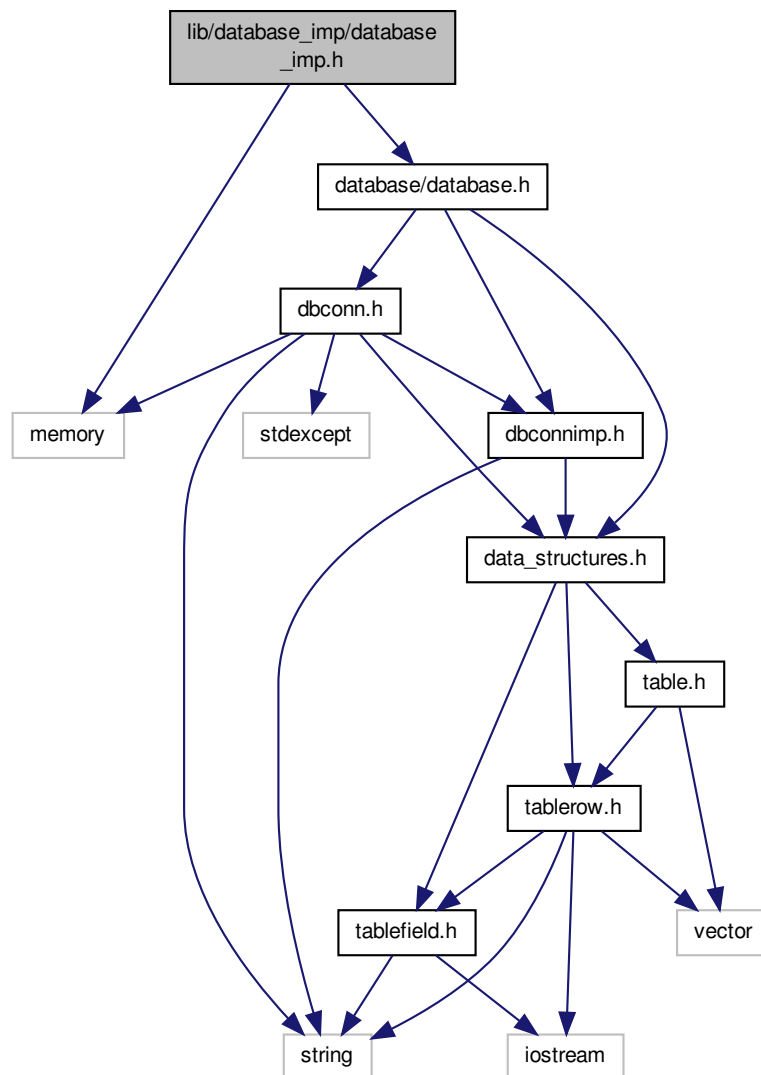
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

## 8.15 lib/database\_imp/database\_imp.h File Reference

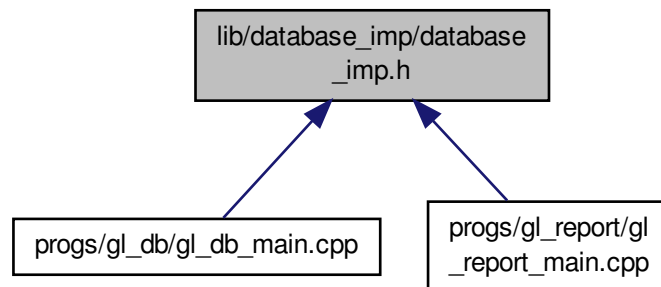
Interface to database implementation factory function.

```
#include <memory>
#include "database/database.h"
```

Include dependency graph for database\_imp.h:



This graph shows which files directly or indirectly include this file:



## Functions

- `DBConnImp * glldb::get_connection` (const std::string database, const std::string hostname, const std::string username, const std::string password)

*Creates and returns a pointer to a database implementation.*

- `std::string glldb::get_database_type` ()

*Returns the name of the compiled-in database type.*

### 8.15.1 Detailed Description

Interface to database implementation factory function.

#### Author

Paul Griffiths

#### Copyright

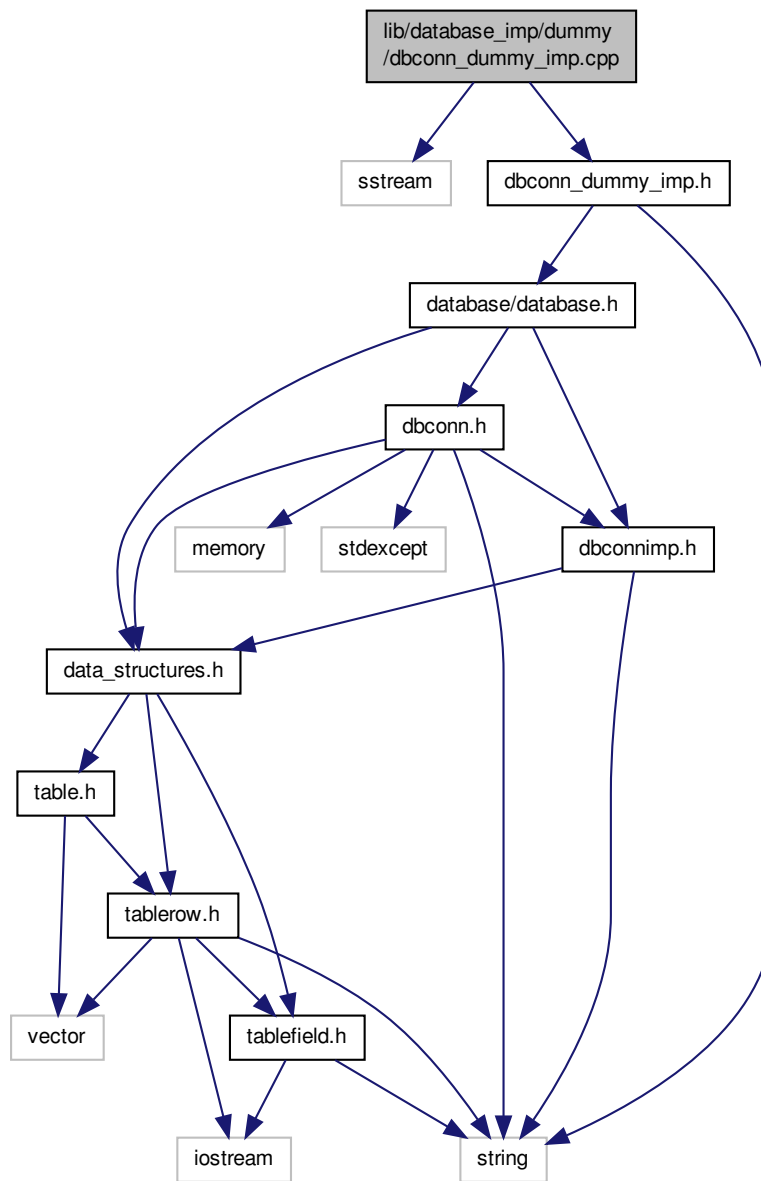
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

## 8.16 lib/database\_imp/dummy/dbconn\_dummy\_imp.cpp File Reference

Implementation of Dummy database connection implementation class.

```
#include <sstream>
#include "dbconn_dummy_imp.h"
```

Include dependency graph for `dbconn_dummy_imp.cpp`:



### 8.16.1 Detailed Description

Implementation of Dummy database connection implementation class.

Author

Paul Griffiths

Copyright

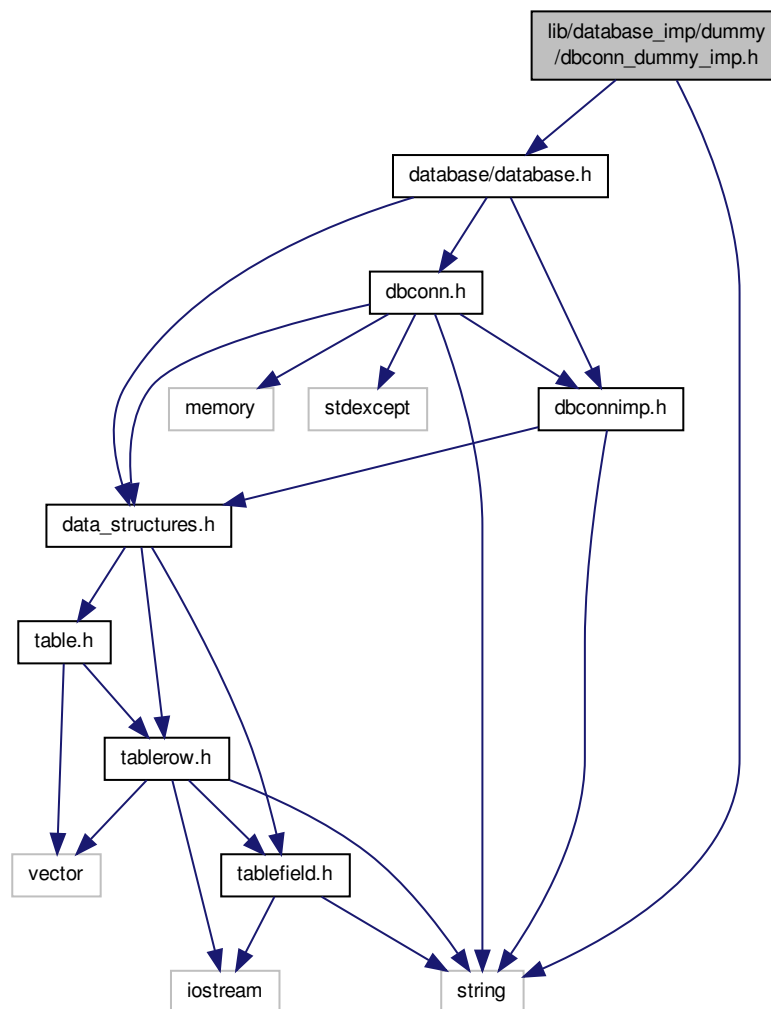
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>



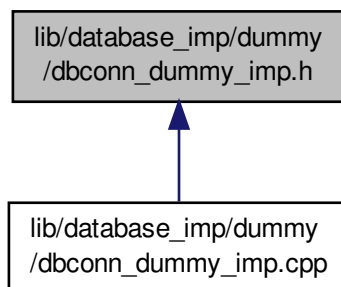
## 8.17 lib/database\_imp/dummy/dbconn\_dummy\_imp.h File Reference

Interface to dummy database connection implementation class.

```
#include <string>
#include "database/database.h"
Include dependency graph for dbconn_dummy_imp.h:
```



This graph shows which files directly or indirectly include this file:



## Classes

- class [gldb::DBConnDummy](#)

*Dummy database implementation class.*

### 8.17.1 Detailed Description

Interface to dummy database connection implementation class.

#### Author

Paul Griffiths

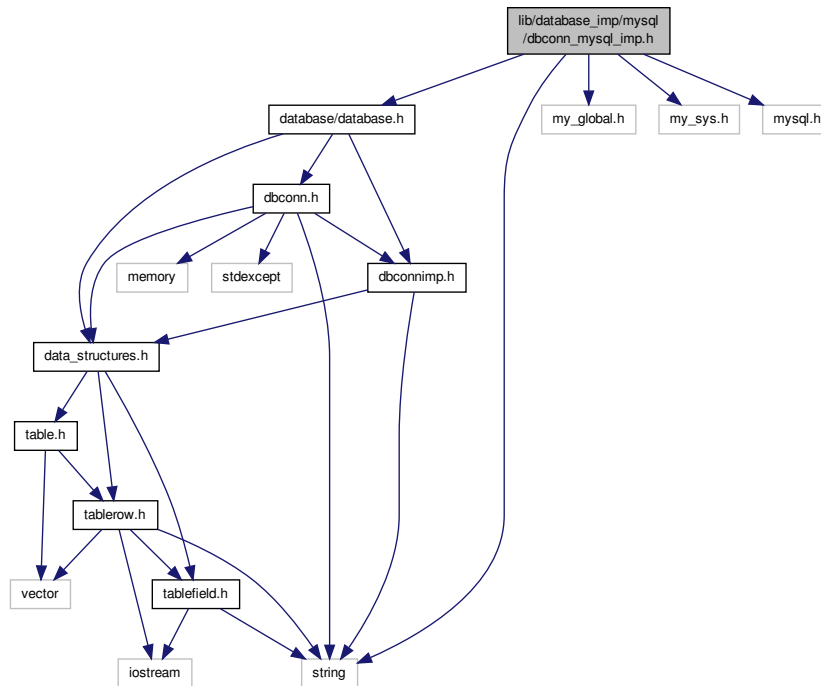


## 8.19 lib/database\_imp/mysql/dbconn\_mysql\_imp.h File Reference

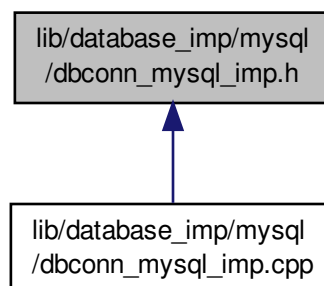
Interface to MySQL database connection implementation class.

```
#include <string>
#include "database/database.h"
#include <my_global.h>
#include <my_sys.h>
#include <mysql.h>
```

Include dependency graph for dbconn\_mysql\_imp.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [gldb::DBConnMySQL](#)

*MySQL database implementation class.*

### 8.19.1 Detailed Description

Interface to MySQL database connection implementation class.

#### Author

Paul Griffiths

#### Copyright

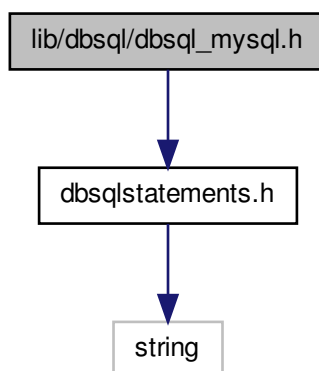
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

## 8.20 lib/dbsql/dbsql\_mysql.h File Reference

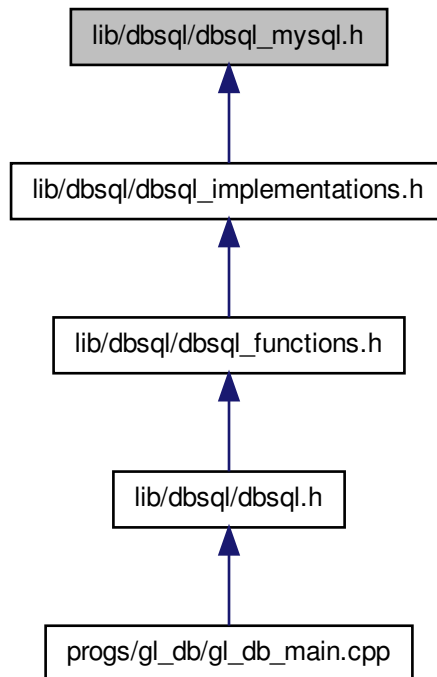
Interface to MySQL SQL statement class.

```
#include "dbsqlstatements.h"
```

Include dependency graph for `dbsql_mysql.h`:



This graph shows which files directly or indirectly include this file:



## Classes

- class [genleg::DBSQLMySQL](#)  
*MySQL SQL statements class.*

### 8.20.1 Detailed Description

Interface to MySQL SQL statement class. Interface to MySQL SQL statement class

#### Author

Paul Griffiths

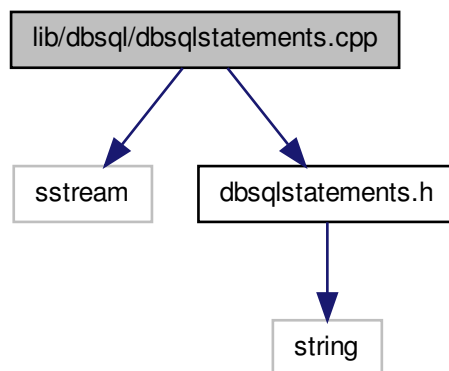
#### Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

## 8.21 lib/dbsql/dbsqlstatements.cpp File Reference

Implementation of SQL statement class.

```
#include <sstream>
#include "dbsqlstatements.h"
Include dependency graph for dbsqlstatements.cpp:
```



### 8.21.1 Detailed Description

Implementation of SQL statement class. Implementation of SQL statement class

#### Author

Paul Griffiths

#### Copyright

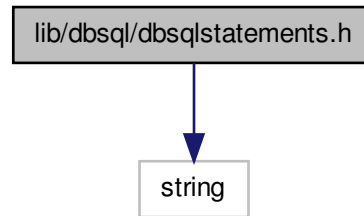
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

## 8.22 lib/dbsql/dbsqlstatements.h File Reference

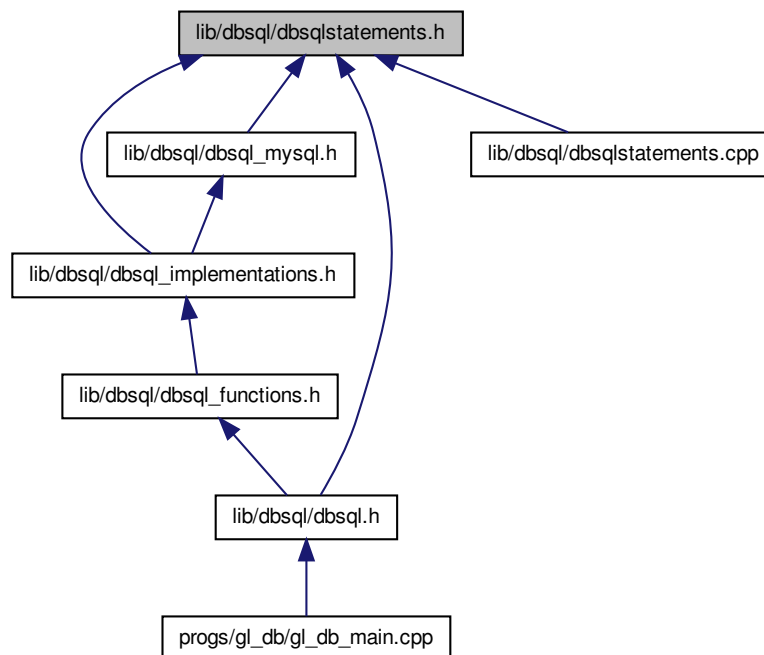
Interface to SQL statement class.

```
#include <string>
```

Include dependency graph for dbsqlstatements.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [genleg::DBSQLStatements](#)  
*SQL statements class.*

### 8.22.1 Detailed Description

Interface to SQL statement class.



**Author**

Paul Griffiths

**Copyright**

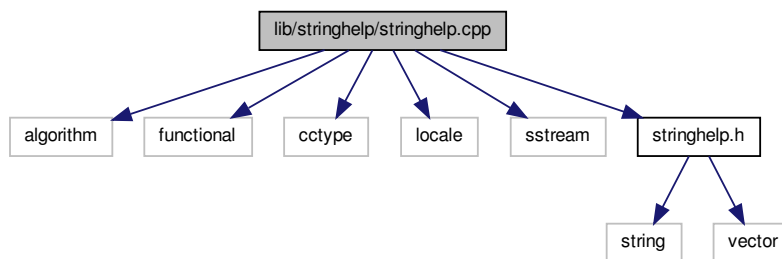
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

## 8.23 lib/stringhelp/stringhelp.cpp File Reference

Implementation of string helper functions.

```
#include <algorithm>
#include <functional>
#include <cctype>
#include <locale>
#include <sstream>
#include "stringhelp.h"
```

Include dependency graph for stringhelp.cpp:



### 8.23.1 Detailed Description

Implementation of string helper functions.

**Author**

Paul Griffiths

**Copyright**

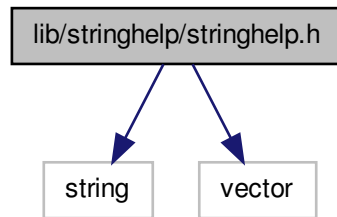
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

## 8.24 lib/stringhelp/stringhelp.h File Reference

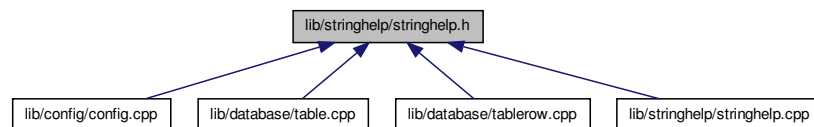
Interface to string helper functions.

```
#include <string>
#include <vector>
```

Include dependency graph for stringhelp.h:



This graph shows which files directly or indirectly include this file:



## Functions

- `std::string & pgstring::trim_front (std::string &s)`  
*Trims leading whitespace from a string.*
- `std::string & pgstring::trim_back (std::string &s)`  
*Trims trailing whitespace from a string.*
- `std::string & pgstring::trim (std::string &s)`  
*Trims leading and trailing whitespace from a string.*
- `std::vector< std::string > pgstring::split (const std::string &s, const char delim)`  
*Splits a delimited string into tokens.*
- `std::vector< std::string > & pgstring::split (std::vector< std::string > &vec, const std::string &s, const char delim)`  
*Splits a delimited string into tokens.*
- `bool pgstring::next_content_line (std::istream &if, std::string &s)`  
*Gets the next content line from a stream.*
- `std::vector< std::string > & pgstring::content_lines (std::vector< std::string > &vec, std::istream &if)`  
*Populates a vector of content lines from a stream.*
- `std::vector< std::vector< std::string > > & pgstring::split_lines (std::vector< std::vector< std::string > > &vec, std::istream &if, const char delim)`  
*Populates a vector of vectors of fields from a stream.*
- `std::string & pgstring::join (std::vector< std::string > &vec, std::string &s, const char delim)`  
*Joins a vector of strings into a delimited line.*

### 8.24.1 Detailed Description

Interface to string helper functions.

#### Author

Paul Griffiths

#### Copyright

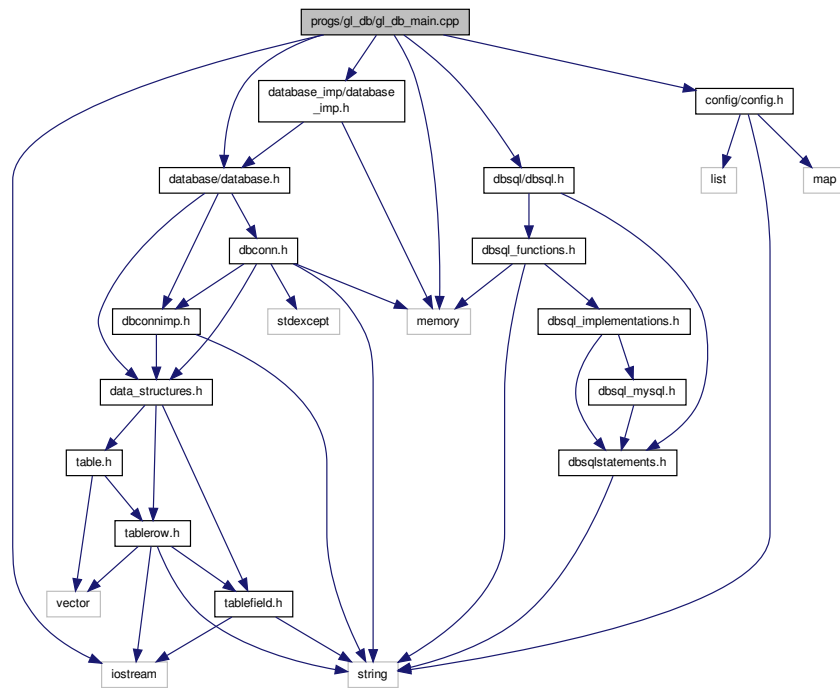
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

## 8.25 progs/gl\_db/gl\_db\_main.cpp File Reference

Main functionality for gl\_db program.

```
#include <iostream>
#include <memory>
#include "database/database.h"
#include "database_imp/database_imp.h"
#include "dbsql/dbsql.h"
#include "config/config.h"
```

Include dependency graph for gl\_db\_main.cpp:



### Functions

- static void [set\\_configuration](#) ([Config](#) &config, int argc, char \*argv[])  
*Sets program configuration options.*
- static void [print\\_usage\\_message](#) ()

*Prints a program usage message.*

- static void `print_version_message ()`

*Prints a program version message.*

- static void `print_help_message ()`

*Prints a program help message.*

- static std::string `login (void)`

*Gets a password from the terminal.*

- int `main (int argc, char *argv[])`

*Main function.*

## Variables

- static const char \* `progrname = "gl_db"`

*Static variable for program name.*

### 8.25.1 Detailed Description

Main functionality for gl\_db program.

#### Author

Paul Griffiths

#### Copyright

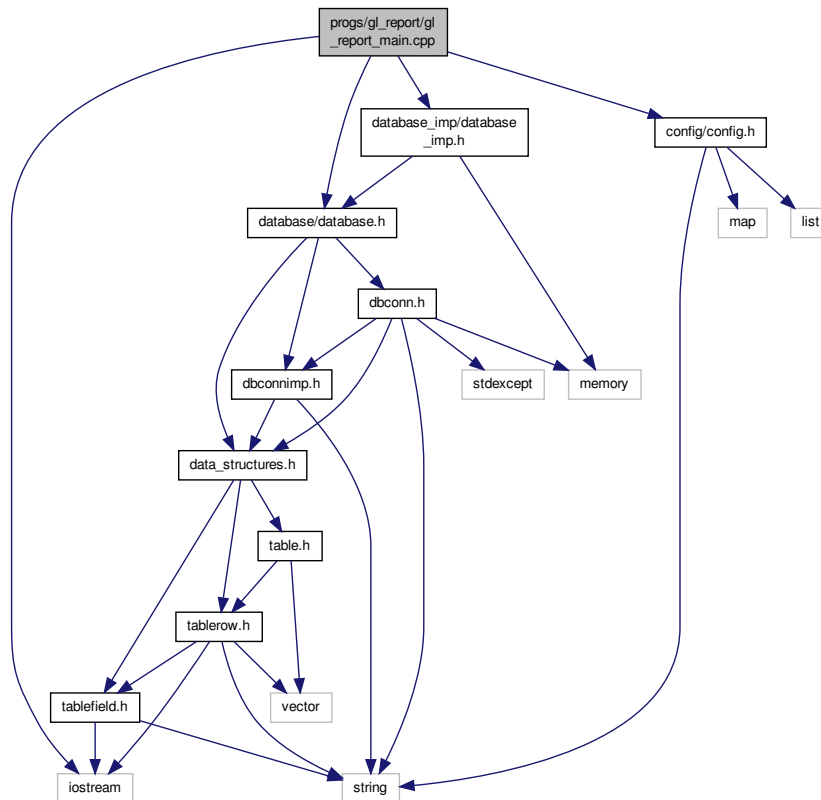
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

## 8.26 progs/gl\_report/gl\_report\_main.cpp File Reference

Main functionality for gl\_report program.

```
#include <iostream>
#include "database/database.h"
#include "database_imp/database_imp.h"
#include "config/config.h"
```

Include dependency graph for gl\_report\_main.cpp:



## Functions

- static void `set_configuration` (`genleg::Config` &config, int argc, char \*argv[])  
*Sets program configuration options.*
- static void `print_usage_message` ()  
*Prints a program usage message.*
- static void `print_version_message` ()  
*Prints a program version message.*
- static void `print_help_message` ()  
*Prints a program help message.*
- static std::string `login` (void)  
*Gets a password from the terminal.*
- int `main` (int argc, char \*argv[])  
*Main function.*

## Variables

- static const char \* `progrname` = "gl\_report"  
*Static variable for program name.*

### 8.26.1 Detailed Description

Main functionality for gl\_report program.

#### Author

Paul Griffiths

#### Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

# Index

- ~Config
  - genleg::Config, [21](#)
- ~DBConnDummy
  - gldb::DBConnDummy, [34](#)
- ~DBConnImp
  - gldb::DBConnImp, [36](#)
- ~DBConnMySQL
  - gldb::DBConnMySQL, [38](#)
- ~DBSQLStatements
  - genleg::DBSQLStatements, [41](#)
- ~Table
  - gldb::Table, [44](#)
- ~TableField
  - gldb::TableField, [50](#)
- ~TableRow
  - gldb::TableRow, [53](#)
- \_XOPEN\_SOURCE
  - config\_getopt.cpp, [60](#)
- add\_cmdline\_option
  - genleg::Config, [22](#)
- append\_field
  - gldb::TableRow, [53](#)
- append\_record
  - gldb::Table, [44](#)
- Config
  - genleg::Config, [21](#)
- config\_getopt.cpp
  - \_XOPEN\_SOURCE, [60](#)
- create\_from\_file
  - gldb::Table, [44](#)
- create\_table
  - genleg::DBSQLStatements, [41](#)
- create\_view
  - genleg::DBSQLStatements, [42](#)
- DBConn
  - gldb::DBConn, [29](#)
- DBConnCouldNotConnect
  - gldb::DBConnCouldNotConnect, [31](#)
- DBConnCouldNotQuery
  - gldb::DBConnCouldNotQuery, [32](#)
- DBConnDummy
  - gldb::DBConnDummy, [33](#), [34](#)
- DBConnException
  - gldb::DBConnException, [35](#)
- DBConnImp
  - gldb::DBConnImp, [36](#)
- DBConnMySQL
  - gldb::DBConnMySQL, [38](#)
- DBSQLStatements
  - genleg::DBSQLStatements, [41](#)
- Database interaction module, [11](#)
- get\_connection, [12](#)
- get\_database\_type, [12](#)
- Database program., [19](#)
- login, [19](#)
- main, [19](#)
- set\_configuration, [20](#)
- drop\_table
  - genleg::DBSQLStatements, [42](#)
- drop\_view
  - genleg::DBSQLStatements, [42](#)
- General purpose helpers., [15](#)
- split, [15](#)
- trim, [15](#)
- trim\_back, [16](#)
- trim\_front, [16](#)
- genleg::Config, [21](#)
- ~Config, [21](#)
- add\_cmdline\_option, [22](#)
- Config, [21](#)
- is\_set, [22](#)
- m\_opts\_set, [23](#)
- m\_opts\_supp, [23](#)
- populate\_from\_cmdline, [22](#)
- populate\_from\_file, [23](#)
- genleg::ConfigBadConfigFile, [23](#)
- genleg::ConfigBadOption, [24](#)
- genleg::ConfigCouldNotOpenFile, [25](#)
- genleg::ConfigException, [26](#)
- genleg::ConfigOptionNotSet, [27](#)
- genleg::DBSQLMySQL, [39](#)
- genleg::DBSQLStatements, [40](#)
- ~DBSQLStatements, [41](#)
- create\_table, [41](#)
- create\_view, [42](#)
- DBSQLStatements, [41](#)
- drop\_table, [42](#)
- drop\_view, [42](#)
- get\_connection
  - Database interaction module, [12](#)
- get\_database\_type
  - Database interaction module, [12](#)
- get\_headers
  - gldb::Table, [44](#)
- gldb::DBConn, [28](#)
- DBConn, [29](#)

- m\_imp, 30
  - operator=, 29
  - query, 29
  - select, 29
- gldb::DBConnCouldNotConnect, 30
  - DBConnCouldNotConnect, 31
- gldb::DBConnCouldNotQuery, 31
  - DBConnCouldNotQuery, 32
- gldb::DBConnDummy, 32
  - ~DBConnDummy, 34
  - DBConnDummy, 33, 34
  - operator=, 34
  - select, 34
- gldb::DBConnException, 34
  - DBConnException, 35
- gldb::DBConnImp, 35
  - ~DBConnImp, 36
  - DBConnImp, 36
  - query, 36
  - select, 36
- gldb::DBConnMySQL, 37
  - ~DBConnMySQL, 38
  - DBConnMySQL, 38
  - m\_conn, 39
  - operator=, 38
  - query, 39
  - select, 39
- gldb::Table, 42
  - ~Table, 44
  - append\_record, 44
  - create\_from\_file, 44
  - get\_headers, 44
  - insert\_query, 44
  - m\_headers, 45
  - m\_quoted, 46
  - m\_records, 46
  - num\_fields, 45
  - num\_records, 45
  - set\_quoted, 45
  - Table, 44
- gldb::TableBadInputFile, 46
- gldb::TableCouldNotOpenInputFile, 47
- gldb::TableException, 48
- gldb::TableField, 48
  - ~TableField, 50
  - length, 50
  - m\_data, 52
  - operator std::string, 50
  - operator<<, 51
  - operator+=, 50
  - operator=, 50, 51
  - TableField, 49
- gldb::TableRow, 52
  - ~TableRow, 53
  - append\_field, 53
  - m\_fields, 55
  - print, 54
  - record\_string, 54
  - size, 55
  - TableRow, 53
- insert\_query
  - gldb::Table, 44
- is\_set
  - genleg::Config, 22
- length
  - gldb::TableField, 50
- lib/config/config.cpp, 57
- lib/config/config.h, 58
- lib/config/config\_getopt.cpp, 59
- lib/database/data\_structures.h, 60
- lib/database/database.h, 61
- lib/database/dbconn.cpp, 63
- lib/database/dbconn.h, 64
- lib/database/dbconnimp.h, 66
- lib/database/table.cpp, 68
- lib/database/table.h, 69
- lib/database/tablefield.cpp, 70
- lib/database/tablefield.h, 71
- lib/database/tablerow.cpp, 73
- lib/database/tablerow.h, 74
- lib/database\_imp/database\_imp.h, 75
- lib/database\_imp/dummy/dbconn\_dummy\_imp.cpp, 77
- lib/database\_imp/dummy/dbconn\_dummy\_imp.h, 79
- lib/database\_imp/mysql/dbconn\_mysql\_imp.cpp, 81
- lib/database\_imp/mysql/dbconn\_mysql\_imp.h, 82
- lib/dbsql/dbsql\_mysql.h, 83
- lib/dbsql/dbsqlstatements.cpp, 84
- lib/dbsql/dbsqlstatements.h, 85
- lib/stringhelp/stringhelp.cpp, 87
- lib/stringhelp/stringhelp.h, 87
- login
  - Database program., 19
  - Reporting program., 17
- m\_conn
  - gldb::DBConnMySQL, 39
- m\_data
  - gldb::TableField, 52
- m\_fields
  - gldb::TableRow, 55
- m\_headers
  - gldb::Table, 45
- m\_imp
  - gldb::DBConn, 30
- m\_opts\_set
  - genleg::Config, 23
- m\_opts\_supp
  - genleg::Config, 23
- m\_quoted
  - gldb::Table, 46
- m\_records
  - gldb::Table, 46
- main
  - Database program., 19
  - Reporting program., 17



- num\_fields
  - gldb::Table, [45](#)
- num\_records
  - gldb::Table, [45](#)
- operator std::string
  - gldb::TableField, [50](#)
- operator<<
  - gldb::TableField, [51](#)
- operator+=
  - gldb::TableField, [50](#)
- operator=
  - gldb::DBConn, [29](#)
  - gldb::DBConnDummy, [34](#)
  - gldb::DBConnMySQL, [38](#)
  - gldb::TableField, [50](#), [51](#)
- populate\_from\_cmdline
  - genleg::Config, [22](#)
- populate\_from\_file
  - genleg::Config, [23](#)
- print
  - gldb::TableRow, [54](#)
- Program configuration module, [14](#)
- progs/gl\_db/gl\_db\_main.cpp, [89](#)
- progs/gl\_report/gl\_report\_main.cpp, [90](#)
- query
  - gldb::DBConn, [29](#)
  - gldb::DBConnImp, [36](#)
  - gldb::DBConnMySQL, [39](#)
- record\_string
  - gldb::TableRow, [54](#)
- Reporting program., [17](#)
  - login, [17](#)
  - main, [17](#)
  - set\_configuration, [18](#)
- SQL statements module, [13](#)
- select
  - gldb::DBConn, [29](#)
  - gldb::DBConnDummy, [34](#)
  - gldb::DBConnImp, [36](#)
  - gldb::DBConnMySQL, [39](#)
- set\_configuration
  - Database program., [20](#)
  - Reporting program., [18](#)
- set\_quoted
  - gldb::Table, [45](#)
- size
  - gldb::TableRow, [55](#)
- split
  - General purpose helpers., [15](#)
- Table
  - gldb::Table, [44](#)
- TableField
  - gldb::TableField, [49](#)
- TableRow
  - gldb::TableRow, [53](#)
- trim
  - General purpose helpers., [15](#)
- trim\_back
  - General purpose helpers., [16](#)
- trim\_front
  - General purpose helpers., [16](#)